Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Institut für Verkehrsplanung und Transportsysteme
Institute for Transport Planning and Systems

(Source: SBB)

Project report

Innovative measures to maximize capacity utilization of the transport network considering the actual demand on the trains

A framework to study rail networks considering the actual demand on the trains coupling passenger demand, operation simulations, and optimization

Authors: Dr. Xiaojie Luan[a]
Dr. Nuannuan Leng[a]
Prof. Dr. Francesco Corman[a]
MSc. ETH Lucas Meyer de Freitas[a,b]
Dipl. Ing. ETH Salem Blum[b]

Affiliations: a. Institute for Transport Planning and Systems, ETH Zurich; b. EBP Schweiz AG

Data Tuesday, 30 November 2021

Contents

1. Introduction

While the social benefits of increased demand for rail services are undeniable, demand increases pose several challenges to rail traffic operations. Such challenges arise because more trains are required to run in order to absorb the rising demand, leading to bottlenecks in the railway infrastructure. This can lead to negative effects for the reliability and stability of the system. Such instability is also intensified by high occupancy rates of trains and crowded stations, which can lead to increased delays in the dwelling times with spill-over effects to the entire network. In Switzerland, which has one of the most intensively used railway networks in the world, the interplay of demand and railway traffic management are therefore of central importance. Historically though, the fields of railway traffic management, emerging from the field of operations research and the field of demand management, emerging from an economic perspective have mostly remained separated.

The current project aims to bridge this gap by establishing a common ground where data, input and output of transport models and detailed railway simulators can be interconnected, and further improved by including optimization approaches managing the railway traffic.

Rail traffic management research has been experiencing large attention in the last years, due the potential of highly automated approaches to control network traffic (Autonomous Train Operations; Traffic Management Systems TMS, such as Rail Control System – RCS at Swiss Federal Railways). This research focuses on producing optimization strategies which minimize overall train delay; with few exceptions, these usually do not consider the demand on trains, that is, the number of passengers on trains and how they are experiencing delays. Those approaches seek the optimal control solution by pure traffic control measures, e.g., changing departure and arrival times (re-timing), changing train orders, changing global routes (re-routing), changing local routes (e.g. a different platform at a station), adaptation of stop pattern and partial/complete cancellation in case of the largest delays. Most of these approaches only focus on minimizing the deviation from the planned schedules (i.e., delays), with microscopic infrastructure details.

Only very few approaches calculate and minimize passenger delays. This is often simply done by considering the number of passengers on board fixed or allowing ideal passenger rerouting based on the shortest (generalized) travel time (e.g., Sels et al., 2016). Schöbel (2007) starts a stream of studies on delay management, which consists of deciding if connecting trains should wait in a station for the passengers of a delayed feeder trains or if they should depart on time. Cadarso et al. (2015) consider passenger flows as dynamic, also including the relevant case of passengers updating their route in a railway network in reaction to a disruption. Corman et al. (2017) introduce an iterative model composed of train scheduling and passenger routing problems and apply one heuristic solution approach to generate the disposition timetable in which train retiming, reordering, rerouting and train connections are considered.

For passengers on trains, the minimization of delay minutes and or effective travel time is the most noticeable effect of the rail traffic management and has impacts on their mode and route choice decisions. A more comprehensive view on passenger reactions to delays and network bottlenecks, considering impact of control decisions and information about schedule changes is rarely utilized for planning or offline capacity analyses (neglecting delays; and neglecting control actions), and to date has been only exceptionally used in the traffic control process.

Combining simulator and optimization approaches for pure railway traffic management has been tackled for instance in the recent ONTIME project (Quaglietta et al, 2016) and allows detailed analysis of many operational details (see Corman et al, 2017). Combining simulators of both railway operations and demand is also in the initial stage of research. For instance, Franke et al. (2018) combine the tools Ontime and Visum, which are well-established and used by several railways, to evaluate operational quality and service quality. Through a combination of microscopic rail simulations (e.g. Medeossi, et al., 2011) and agent-based modelling, a joint detailed simulation of both railway operations and demand is possible nowadays though. Through this combination, it is possible to develop demand-oriented rail traffic management strategies, within existing tools, which have showed already value in research as well as in practice. Borndörfer, et al. (2014) propose a micro-macro aggregation-disaggregation approach for timetable optimization of a very dense railway corridor, with detailed microscopic representation in railway simulation and a less detailed macroscopic representation in optimization methods. Högdahl, et al. (2019) propose an approach combining microscopic simulation and macroscopic timetable optimization to minimize the weighted sum of scheduled travel time and expected delay. At the same time, it is also possible to include demand information to simulate delays arising from larger demand in intensively used sections of the network (e.g. Leng et al., 2020). Conversely, the effects of demand management strategies on railway traffic can also be studied. The long term value of such an interconnected framework includes a better description of reliability of operations in transport models; and actions reactively performed when delays occur. This enables detailed modelling of synchronization of complex transport chains, evaluating actual reliability, reactive operations of public transport systems, evaluating of interrelation between demand and supply in case of non-performance.

One major limitation of those approaches is the generic usability, i.e. availability of data in suitable format and of software tools; interconnectivity between different programs; extendibility of the tools with regards to specific aspects; and finally, their complex interaction when put in control loops with optimizers.

2. Goals & approach

In order to examine innovative measures to improve the use of rail infrastructure considering passenger flows (demand, passenger assignment), it is first necessary to develop tools that can link demand and railway operations. We define the requirements for a suitable tool to achieve this goal, as follows:

(1) Simulate real-world railway operations including microscopic constraints from the infrastructure (geometry, layout, signalling, speeds), rolling-stock and operational rules (dispatching rules and traffic management).

(2) Include disaggregate demand information, that is, time-dependent passenger numbers and flows on trains and stations, computed by means of an assignment procedure.

(3) Being able to influence suitable control variables to steer the system to a desired better level of service.

Item (1) is a typical microscopic modelling of railway operations, compatible with blocking time theory and practical rules of the signalling and safety system, which allow for safe and realistic movements of trains in their kinematic aspect (speed, distance) as well as usage of infrastructure (block occupation and release).

With the information from (2), it is possible to estimate how many passengers are boarding each train at each station and therefore estimate how many passengers are on a particular train and where these passengers are heading to. This estimate can be made dynamically dependent on delays or schedule changes. This passenger flow information per vehicle can then be used to develop and test traffic management strategies to find a solution, which minimize passenger delay instead of train delays. This allows using railway resources (vehicles, but also infrastructure) in the most effective way. The ultimate goal is to increase passenger satisfaction and minimize operational costs.

Finally, simulation is only a first step to be able to prescriptively change operation to reach a maximum transport performance. When passengers plan a journey, their perception of quantifiable attributes (e.g., traveling time, waiting time, and transfers), non-quantifiable attributes (e.g., comfort, seat availability, and aversion towards being delayed), and whether they are frequent or occasional travellers have a significant impact on their behaviour. In the presence of disruptions, passengers receive operation information about expected delays and might reconsider their choices. Moreover, they could experience different delays than actually communicated ones, and in a longer term reconsider their habitual choices. The chances to implicitly include such complexity in pure optimization models is very small, therefore much of past research investigated simulation-optimization techniques to include the most relevant interactions of the passenger flow and train operations (for instance, dwell time extensions at platform; passenger delay; transfers and route choice). In the specific case, we want to have a simulator for both railway operations and passenger dynamics; coupled by an overarching optimization framework (3) into a generic framework or toolkit.

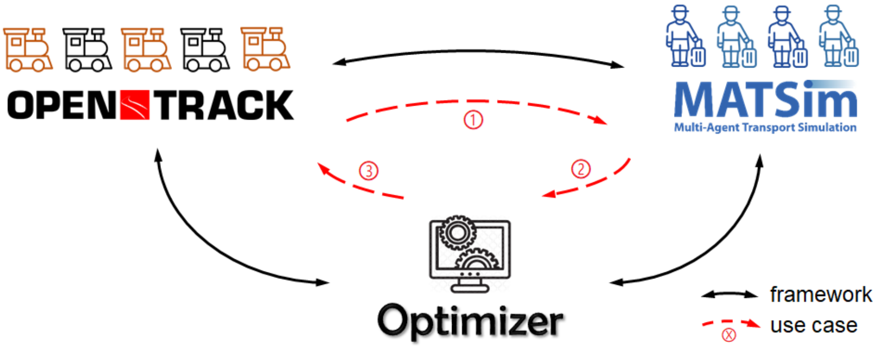


Figure 1 Framework and use case

The method we propose is to combine rail simulation, large-scale passenger simulation, and traffic management strategies, see the framework illustrated in Figure 1. The three aspects correspond to OpenTrack, MATSim, and an optimizer respectively. We present our framework in three parts:

(1) The description of the development of a comprehensive framework interconnecting the various simulators and the optimization, namely, combining the simulation of railway operations (i.e., OpenTrack) and disaggregated demand (i.e., MATSim), here reported in Section 3.

(2) The discussion of strategies for the explicitly integration of optimization, and possible evaluation of different strategies for passenger-oriented rail traffic and demand management, reported here in Section 4.

(3) The testing of the developed strategies on a representative section of a realistic network to evaluate the benefits of the whole sets of method, reported here in Section 5. The use case of the proposed framework is highlighted by the red dashed lines in Figure 1 and detailed in Figure 3. The other possibilities are discussed in Section 6, for the future studies.

3. Integrated framework of Opentrack-MATSim-Optimizer

The toolkit which we develop is based on an integration of two existing transport simulation tools, already available and used in academic and industrial settings, such as OpenTrack for railway operations and MATSim for transport demand / passenger assignment.

MATSim is an open source, activity-based, multi-agent simulator of travel demand implemented in Java, co-developed at ETH Zürich and TU-Berlin (Horni et al., 2016). It is modular and designed to handle large-scale scenarios. In MATSim, each agent has a daily activity chain (a plan). In an iterative process, based on a co-evolutionary algorithm, each agent tries to maximize its daily plan's score by changing routes, modes, end times, and locations of leisure and shopping activities (e.g. Rieser, et al., 2018).

OpenTrack is a microscopic rail simulation software and a benchmark for the simulation of railway operations (Nash and Huerlimann, 2004). In a simulation, predefined trains run according to a timetable on a railway network. During the simulation, OpenTrack calculates train movements under the constraints of the infrastructure, signalling system and timetable. OpenTrack can be extended by other tools and toolkits, see e.g. Stephan, (2008). The tool also allows for the simulation of randomly seeded delays and dispatching strategies to solve conflicts, therefore incorporating the dynamics and stochastics of rail operations.

The combination of both models into a toolkit will thus allow to:

- Add information on demand levels for trains and stations and use this information to simulate delays and therefore system instability issues related to high demand as well as to use demand information to test and evaluate demand-oriented dispatching strategies.
- Use delay information representing real-world dispatching rules from OpenTrack which affect the mode choice for rail to simulate the adaptive behaviour of agents in MATSim to such strategies into equilibrium or non-equilibrium solution.

From the point in which this tool is working, time-dependent and course specific train dispatching and passenger demand management strategies can be simulated, and their outcomes evaluated.

The architecture of the toolkit is schematically described in the Figure 2. Grey boxes are general inputs, green boxes are processed inputs, orange boxes are produced outputs from the simulation, yellow diamonds define the boundary of the TMS (traffic management system) and finally. The blue diamonds are the newly developed processing algorithms for combining each part, which are described more in detail, individually, in the remainder of this Section.

The general inputs (grey boxes) mainly consist of two parts:

(1) "Static OT Simulation Inputs" are the standard OpenTrack files for the network section on which the developed algorithms will be tested on. It consists of network infrastructure, train courses including their itineraries as well as the rolling stock for each course.

(2) "MATSim Input" are standard MATSim files about passenger demand. It consists of passengers' daily plans, passengers' attributes, physical network, facilities about passengers' activities, transit schedules about train timetable, as well as transit vehicles about detailed features of train types. After the MATSim runs till equilibrium, Events file from a simulation can show passenger demand. This also determined a pre-processed

input file, “MATSim Realized demand”. The “timetable” is a further possible input, as it can be included into a MATSim input file and thus influence all the downstream calculations.

For outputs of this architecture, there is a list of deliverables, including:

- A matching table of OpenTrack and MATSim (green box named “Correspondence table”): with matched train IDs and station IDs.
- Evaluating aggregated passenger numbers on trains, stations, or route sections (green box named “Demand dataset”): boarding per train or per station, alighting per train or per station, the load of each train at each route section, cross-sectional load on trains, and so on.
- Evaluating train delays in railway network (orange box named “Train delays”): delay of each train, system-level train delays, etc.
- Evaluating passengers’ delays due to corresponding train delays (orange box named “Passenger weighted delays”): cross-sectional delays, arrival passenger delays, etc.
- Evaluating optimized passengers’ delays (orange box named “Expected optimized weighted delays”): the decreased passengers’ delays thanks to the optimized dispatching strategies.

The present report considers a specific use case to examine the value of the developed framework. Besides the above, the toolkit has great potential; the possibilities that could be done with the toolkit in the future studies are discussed in Section 6.

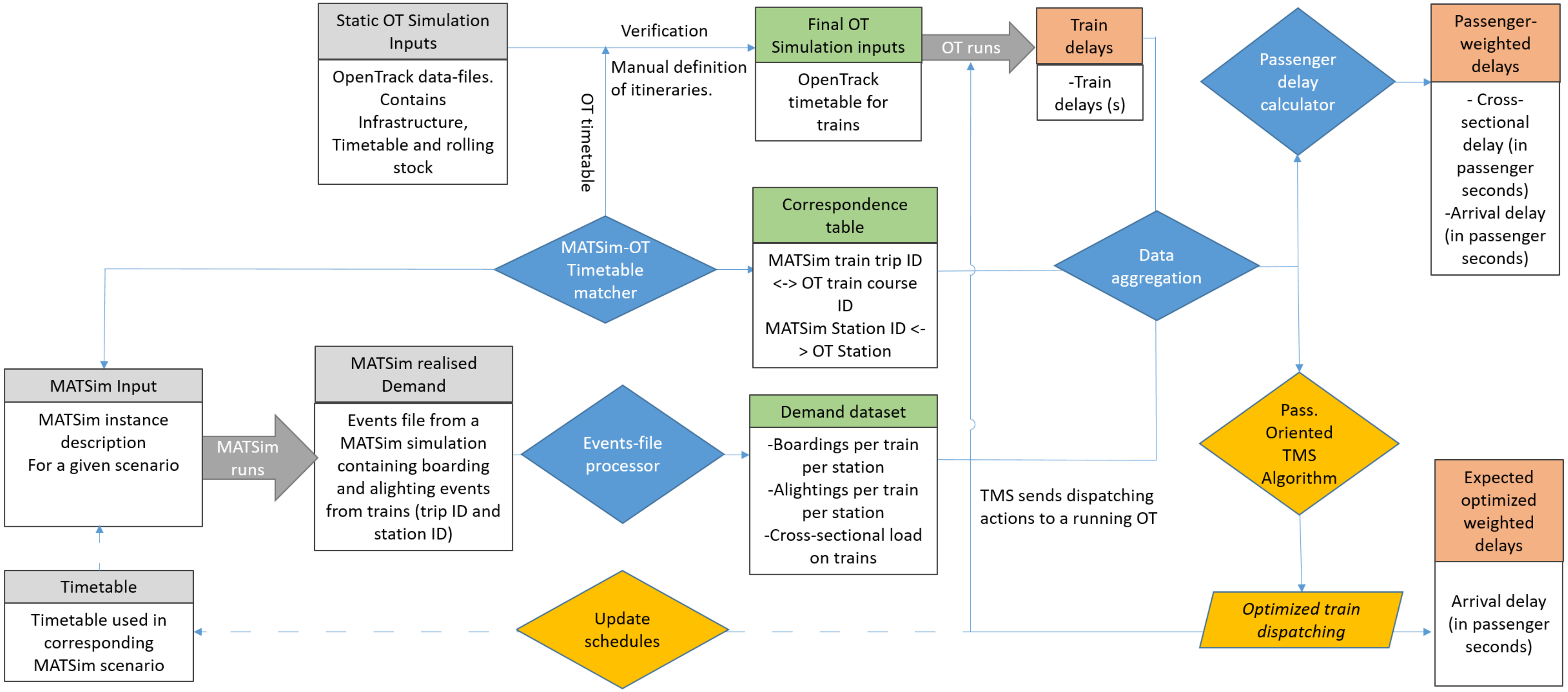
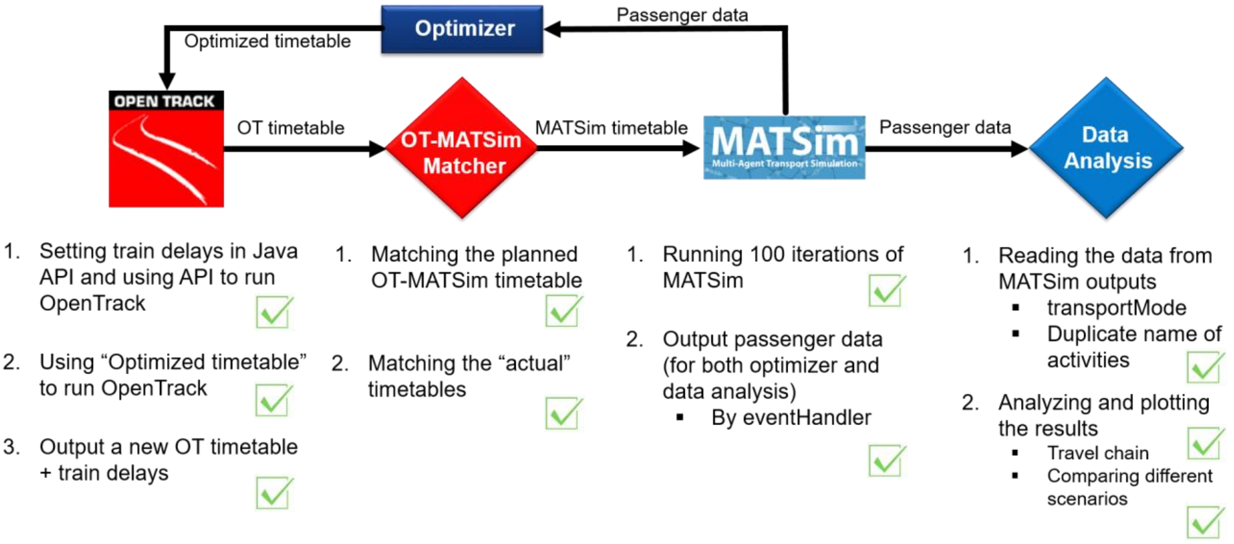


Figure 2 Toolkit scheme

In a nutshell, Figure 3 shows a toolchain connecting various models, which are already functional for different test cases. This feasible loop is capable to link passenger demand simulation and railway operation simulation together by a so-called OT-MATSim Matcher. This matcher translates the updates of railway timetable from OpenTrack to MATSim, including time or route changes. With an updated timetable, MATSim simulates the passenger behaviors in a multi-modal network (a more realistic multimodal transport network instead of a pure railway network) to understand the demand in real world. Combining both the demand from MATSim and timetable (incl. Infrastructure) from OpenTrack, the optimizer optimizes the total passenger delay (or train delay). Furthermore, the optimized timetable is capable to re-write back to OpenTrack to check the operating feasibility in an ultra-microscopic level of railway infrastructure. Theoretically, this toolchain can be used for multiple iterations to improve the solutions in a given train delay test case, taking advantages from all the three main models.



1. Setting train delays in Java API and using API to run OpenTrack ✓
2. Using "Optimized timetable" to run OpenTrack ✓
3. Output a new OT timetable + train delays ✓

1. Matching the planned OT-MATSim timetable ✓
2. Matching the "actual" timetables ✓

1. Running 100 iterations of MATSim ✓
2. Output passenger data (for both optimizer and data analysis)
   - By eventHandler ✓

1. Reading the data from MATSim outputs
   - transportMode
   - Duplicate name of activities ✓
2. Analyzing and plotting the results ✓
   - Travel chain
   - Comparing different scenarios ✓

Figure 3 A toolchain to connect various models

Next, we will introduce more details about processing algorithms (blue and yellow diamonds in Figure 2). Sections 0-3.4 correspond to the 4 blue/yellow diamonds with the name same to the section title. Sections 3.5-3.6 present the "Data Aggregation", namely the processing of the OpenTrack data and the MATSim data respectively. In Section 3.7, we describe how to apply the optimized timetable in OpenTrack, i.e., the yellow diamond of "Update schedules".

3.1. MATSim-OpenTrack timetable matcher

This algorithm has two goals:

(1) Produce a correspondence table matching each train vehicleID in MATSim to each OpenTrack train course ID as well as a correspondence table matching the station ID's from MATSim, which are included in the events file with the OpenTrack stations.

(2) Update the MATSim timetable based on the optimized TMS-timetable.

Goal (1) is in essence a translator between the infrastructure and services in MATSim and OpenTrack. Based on this translation, the "Passenger delay calculator" is able to match the train delay to MATSim demand, within the level of precision and time resolution allowed by the various platforms.

Technically, the current timetables used in MATSim and OpenTrack show inconsistencies although covering the same time period. Both are created by different sources and there is no assurance of coherence between both. Specifically, the timetable in MATSim is converted from PTV Visum (a commercial software for transport planning) while that in OpenTrack is from HAFAS (a software for the timetable information of the company HaCon, Hannover Consulting). In other terms, there is no one direct attribute or key (e.g. trainID or other) to match easily the timetables in MATSim and OpenTrack.

To solve this issue of data inconsistency, we match the trains in MATSim and OpenTrack to be performed in two steps. The first step is to match courses at a train line level, matching courses with the same stop sequences. The second step is to match at course level, finding a unique course in both MATSim and OpenTrack with the same departure time. The great majority of courses could be found in this way. Some unique courses might not always be found depending on various reasons. Extra-courses included in the OpenTrack timetable for testing capacity levels will not be included in MATSim.

Goal (2) is needed to ensure consistency between the matched timetables and the optimized timetable from the TMS. The resulting TMS-timetable is a product of different traffic management strategies, such as retiming, reordering, rerouting, newly added or deleted trains. To enable the use of the MATSim-OpenTrack timetable matcher, we standardize the TMS-timetable to the same xml-File format used for OpenTrack timetables. The MATSim timetable (called transitSchedule in MATSim) is updated with the new departure times at each station. Rerouted trains are implemented by deleting the corresponding course in MATSim and implementing a new one with updated stops.

3.1.1. Discussion about data consistency

In this subsection, we discuss about data consistency of two simulation tools (MATSim and OpenTrack).

From a network level, Figure 4 shows the data scope collected from SBB. The data includes 487 stations (195 with stops) and 528 courses in total. The train schedule mainly covers the time range from 5:00 to 18:00, plus few midnight train from 23:00 to 2:00.

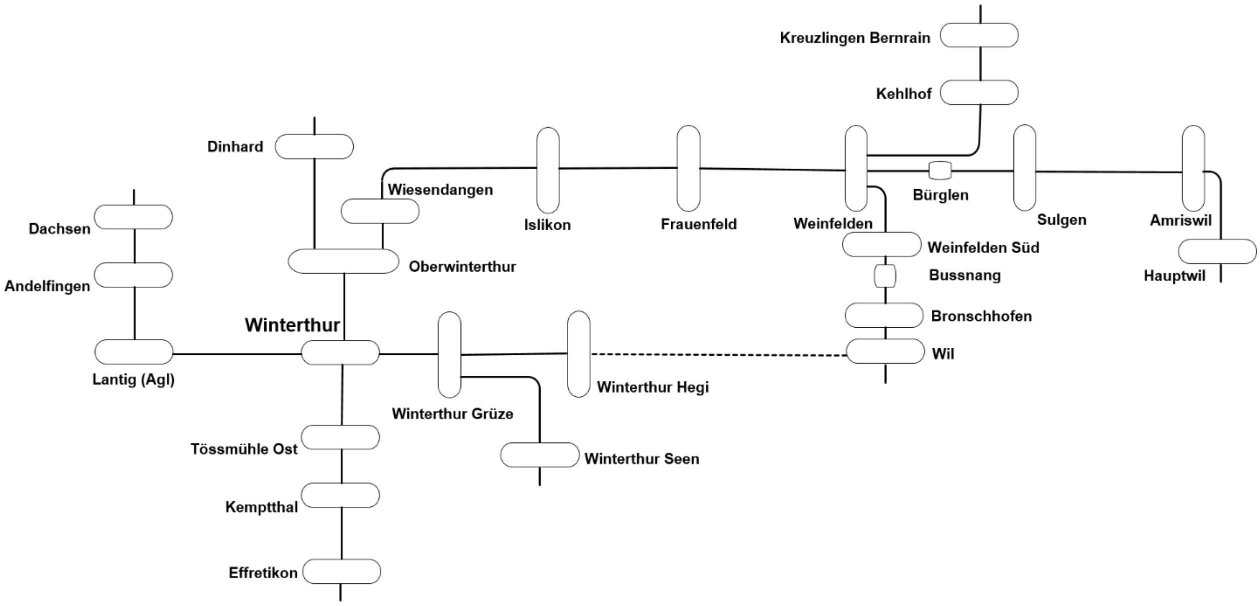


Figure 4 Data scope about OpenTrack

A comparable MATSim data collected from SBB covers either a “small area”, which is more centralized around Winterthur (see yellow colour in Figure 5), or a “full area” from Winterthur region to east border of Switzerland (see orange colour in Figure 6). Each of the case has either a 10% (one agent presents 10 passengers in reality) or 100% (one agent presents one passenger in reality) in the given simulation.

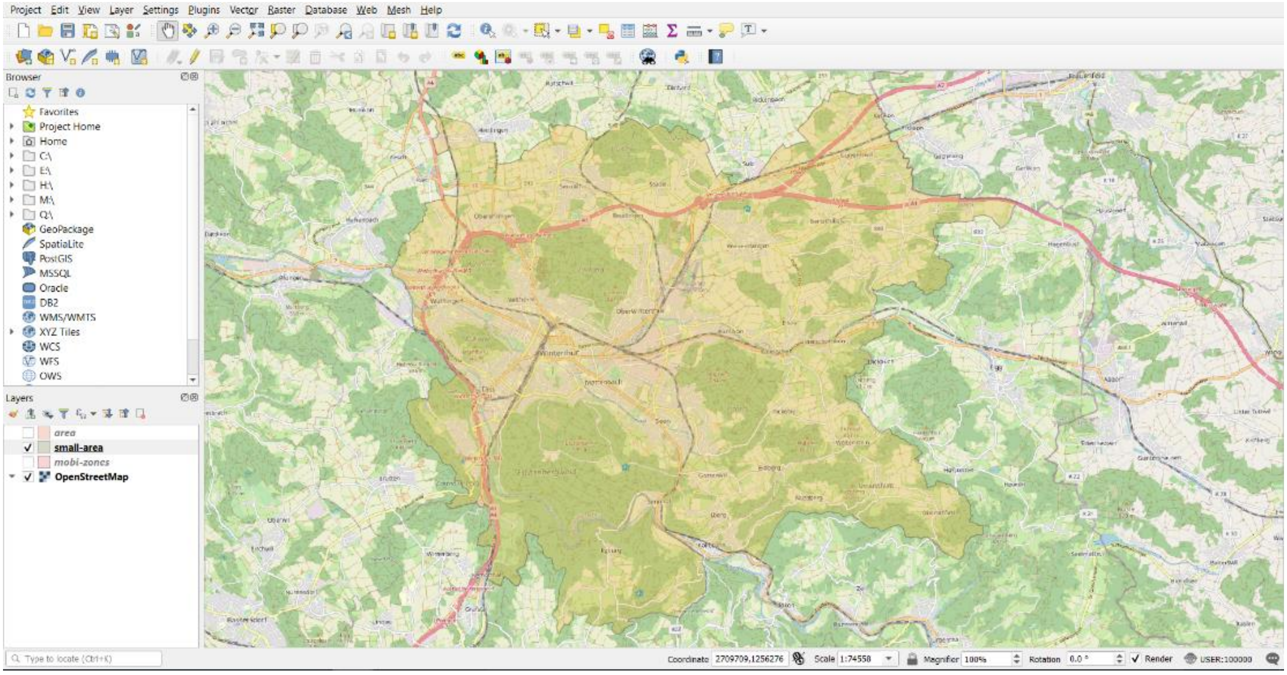


Figure 5 “Small area” data about MATSim

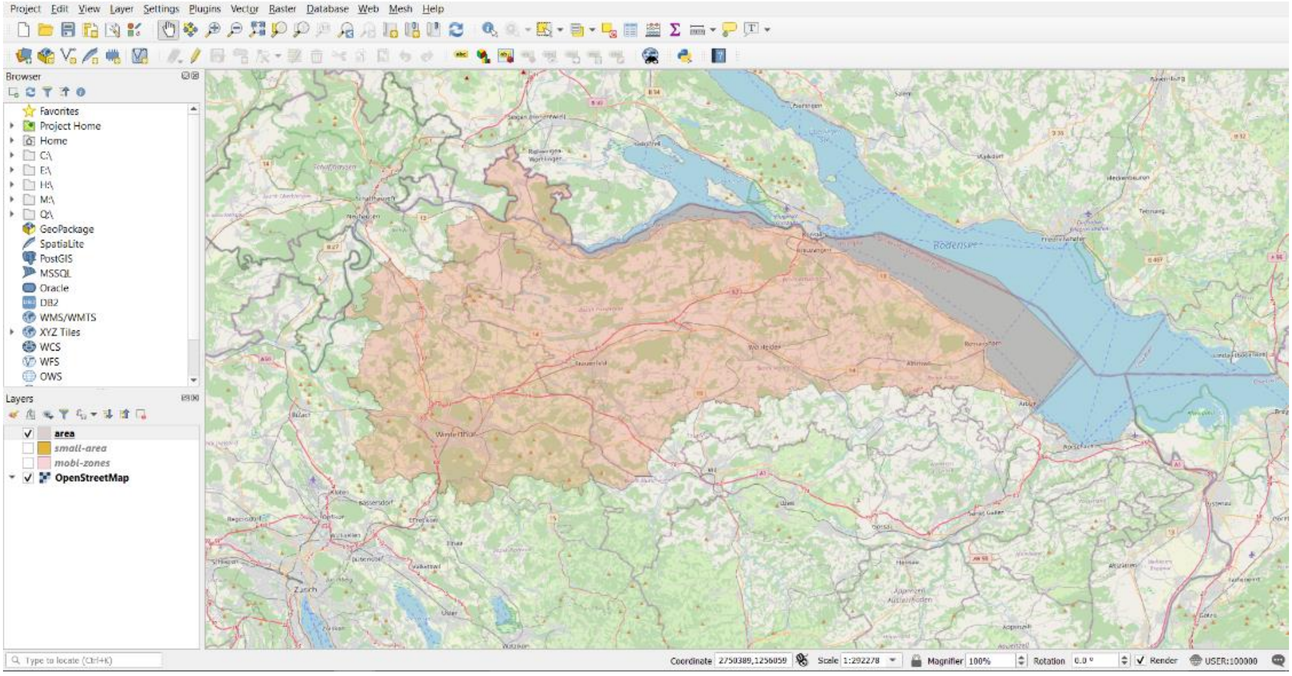


Figure 6 “Full area” data about OpenTrack

We compare the details of MATSim data of these 4 cases in Table 1, checking the differences about the number of agents, facilities, nodes, links, stops, transit lines and transit vehicles. Considering the running time of MATSim, the matches with OpenTrack in a small area that Optimizer capable to solve, we focus on

Table 1 Data scope about MATSim

	area	pct	Number of agents	Number of facilities	network	transitSchedule	transitvehicles
1	small	10%	17,378	28,978	Node: 745,881 Link: 1,600,611	Stops: 25,873 transitLine: 2,011	191,893
2	small	100%	173,784	127,184			
3	full	10%	40,883	64,235			
4	full	100%	408,834	235, 950			

The data consistency of the two simulation tools is compared spatially and temporally.

Spatially, OpenTrack presents an ultra-microscopic network of railway infrastructure, which includes much more details about railway system than a macroscopic representation in MATSim. However, MATSIM is spatially larger than OpenTrack in the aspects that MATSim includes other transport modes, e.g. bus/ tram/ bike/ walk.

Temporally, MATSim runs a transitSchedule for 32 hours of one iteration, which is due to its own features to allow all the agents return home at the end of day. In contrast, OpenTrack mainly includes the schedules during the daytime (from the data we collected). Moreover, a typical optimizer is capable to run around 2 hours of train schedules based on literature review.

To solve this data inconsistency between MATSim and OpenTrack, we proposed the following five different approach:

1) **Cut**: MATSim works only for the time horizon of OpenTrack. Instance files; and especially day-to-day re-planning need to be heavily adjusted.
2) **Ignore**: MATSim works for the entire 32 hours period, but the 3 hours of OpenTrack are what we use in our analysis. What happens outside is only useful for the generation of agents.
3) **Expand**: OpenTrack instances are delivered , which cover the entire 32 hours (or 24 hours) period
4) **Extend**: the OpenTrack instances currently available are artificially extended to cover the remaining hours, by assuming services keep running at the same frequency and times
5) **Merge/replace**: MATSim looks at 32 hours; OpenTrack focuses on a subset of 3-8 hours of those. The data from OpenTrack overwrites what MATSim originally thought. At the end of the OpenTrack focus, there might be some discrepancy

We apply the fifth approach “Merge/replace” in this present project. The benefits of this approach is to keep MATSim running as its designed structure without extra troubles, as well as the two simulations could match for the targeted train schedules in a defined test case.

Checking closer to the detailed data of the two simulation tools, we surprisingly discover that the differences of the two coding systems are significant. More specifically, the trainIDs are coded in different ways, for instance, “061_007031” in MATSim is the same train as “11526” in OpenTrack. In addition, the periodic train schedules are grouped in MATSim transitScheudle file, while OpenTrack represents each train in a separated way.

3.1.2. Design of OT-MATSim matcher

Here we use two examples to show the timetable data structure of the two simulation tools.

```xml
<timetable title="OpenTrack timetable" application="OpenTrack" date="Thu Aug 19 17:34:09 2021">
    <course>
        <courseID>11526</courseID>
        <timetableEntry stopInformation="yes">
            <stationID>BGL</stationID>
            <arrivalTime format="hh:mm:ss" type="planned" valid="yes">08:51:36</arrivalTime>
            <departureTime format="hh:mm:ss" type="planned" useDepartureTime="no" valid="yes">08:51:36</departureTime>
            <waitTime format="s">0</waitTime>
            <delayTime format="s">0</delayTime>
        </timetableEntry>
        <timetableEntry stopInformation="no">
            <stationID>WFKV</stationID>
            <arrivalTime format="hh:mm:ss" type="planned" valid="yes">08:52:52</arrivalTime>
            <departureTime format="hh:mm:ss" type="planned" useDepartureTime="yes" valid="yes">08:52:52</departureTime>
            <waitTime format="s">0</waitTime>
            <delayTime format="s">0</delayTime>
        </timetableEntry>
        <timetableEntry stopInformation="yes">
            <stationID>WF</stationID>
            <arrivalTime format="hh:mm:ss" type="planned" valid="yes">08:55:36</arrivalTime>
            <departureTime format="hh:mm:ss" type="planned" useDepartureTime="yes" valid="yes">08:56:36</departureTime>
            <waitTime format="s">60</waitTime>
            <delayTime format="s">0</delayTime>
        </timetableEntry>
    </course>
```

Data 1 Example of OpenTrack timetable file

```xml
<transitLine id="SBB_2020_005-D-15172">
    <transitRoute id="18742_1_9">
        <attributes>
            <attribute name="02_TransitLine" class="java.lang.String">SBB_2020_005-D-15172</attribute>
            <attribute name="03_LineRouteName" class="java.lang.String">BN-JEG</attribute>
            <attribute name="04_DirectionCode" class="java.lang.String">H</attribute>
            <attribute name="05_Name" class="java.lang.String">Bas301_Takt1</attribute>
            <attribute name="06_OperatorName" class="java.lang.String">Regionalverkehr Bern-Solothurn</attribute>
            <attribute name="07_OperatorCode" class="java.lang.String">RBS</attribute>
            <attribute name="08_TSysName" class="java.lang.String">RV - ProduktD</attribute>
            <attribute name="09_TSysCode" class="java.lang.String">D</attribute>
        </attributes>

        <transportMode>rail</transportMode>
        <routeProfile>
            <stop refId="3667" departureOffset="00:00:00" awaitDeparture="true"/>
            <stop refId="3167" arrivalOffset="00:03:30" departureOffset="00:04:00" awaitDeparture="true"/>
            <stop refId="2394" arrivalOffset="00:06:30" departureOffset="00:07:00" awaitDeparture="true"/>
            <stop refId="19045984" arrivalOffset="00:07:30" departureOffset="00:08:00" awaitDeparture="true"/>
            <stop refId="2287" arrivalOffset="00:10:30" departureOffset="00:11:00" awaitDeparture="true"/>
            <stop refId="2723" arrivalOffset="00:11:30" departureOffset="00:12:00" awaitDeparture="true"/>
            <stop refId="2696" arrivalOffset="00:12:30" departureOffset="00:13:00" awaitDeparture="true"/>
            <stop refId="2997" arrivalOffset="00:14:30" departureOffset="00:15:00" awaitDeparture="true"/>
            <stop refId="2002" arrivalOffset="00:18:00" departureOffset="00:18:00" awaitDeparture="true"/>
        </routeProfile>
        <route>
            <link refId="pt_3667"/>
            <link refId="pt_3667-pt_3167"/>
            <link refId="pt_3167"/>
            <link refId="pt_3167-pt_2394"/>
            <link refId="pt_2394"/>
            <link refId="pt_2394-pt_19045984"/>
            <link refId="pt_19045984"/>
            <link refId="pt_19045984-pt_2287"/>
            <link refId="pt_2287"/>
            <link refId="pt_2287-pt_2723"/>
            <link refId="pt_2723"/>
            <link refId="pt_2723-pt_2696"/>
            <link refId="pt_2696"/>
            <link refId="pt_2696-pt_2997"/>
            <link refId="pt_2997"/>
            <link refId="pt_2997-pt_2002"/>
            <link refId="pt_2002"/>
        </route>
        <departures>
            <departure id="178234" departureTime="06:22:00" vehicleRefId="178234"/>
            <departure id="178235" departureTime="06:52:00" vehicleRefId="178235"/>
            <departure id="178236" departureTime="07:22:00" vehicleRefId="178236"/>
            <departure id="178237" departureTime="07:52:00" vehicleRefId="178237"/>
            <departure id="178238" departureTime="08:22:00" vehicleRefId="178238"/>
            <departure id="178239" departureTime="08:52:00" vehicleRefId="178239"/>
            <departure id="178240" departureTime="09:22:00" vehicleRefId="178240"/>
            <departure id="178241" departureTime="09:52:00" vehicleRefId="178241"/>
```

Data 2 Example of MATSim timetable file

OpenTrack structures the timetable data in a very different way. Each train is recorded separately as a course, including the details about trainID (i.e. courseID) and the detailed stop patterns with the planned/updated departure and arrival time.

In contrast, MATSim structures the timetable in a periodic way. That means all the schedules that have the same stop patterns, planned routes and offsets from the departure stop are written in one “transitRoute”, with different departures separately recorded in “<departures>”. Under “<departures>”, each specific train is recorded by “vehicleRefId”.

The designed OT-MATSim matcher not only solves the mentioned data inconsistent issue between MATSim and OpenTrack, but also translates the changes of timetable from “Opentrack timetable” to “MATSim transitSchedule”. More specifically, the three cases of timetable changes are considered in the matcher:

First case pertains the trains that are delayed or re-ordered in OpenTrack. In the MATSim transitSchedule file, the periodic “TransitRoute” and “departures” need to be separated, and then the “departure Time” and “offset” are copied for the entire train with changes only on the stops that are delayed.

Second type is the trains that are re-routed or newly added train in OpenTrack. In the MATSim transitSchedule file, if a route exists already, copy “TransitRoute” in current “transitSchedule.xml”, and make specific changes; if a route does not exist, create a new “TransitLine”, be careful about the links with the physical network.

Third is the trains that are deleted in OpenTrack. In the MATSim transitSchedule file, the level of “departures” are deleted. For more details, if only few stops are deleted from the schedules, then the MATSim timetable needs to rewrite the entire itinerary with deleting the stops and departure/ arrival time.

3.1.3. Code availability

The developed MATSim-OpenTrack timetable matcher is accessible in the following Github repository: https://github.com/ebp-group/matsimOTmatcher

The available has two main functionalities:

-Translation of OpenTrack-Timetables to MATSim timetables

-Matching of timetables, in order to find corresponding courses between both timetables.

Since the MATSim and OpenTrack timetables are not equivalent in their terminology, a matching algorithm had to be developed to find equivalent routes and courses. OpenTrack consistently applies the official train designations, used in the published timetables[1], while MATSim uses designations used for planning but not for displaying the timetables. Therefore, a good practice for the future would be to standardize the designation of trains across all applications. While this is not the case, the matching algorithm was designed to perform the following tasks:

1. Match at the route level: Based on the matching of served stops.
2. Match at course level: Based on a search of departure times from first matched stop.

The algorithm provided a match of ca. 90% of courses, which is a satisfactory rate, considering previous experience of matching timetables across different platforms. Good practice in the naming of the routes and trips, such as consistently writing the transit Mode as “Rail” in MATSim instead of the routeID and writing correct departure times (errors are available), would improve the quality of the matching.

3.2. Events-file processor

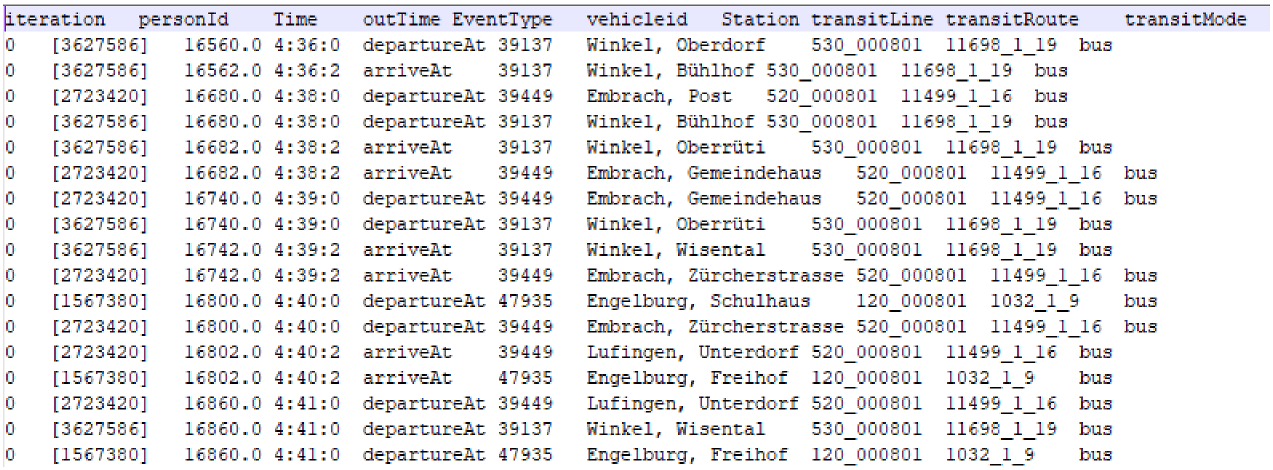
“Events” is a MATSim simulation output, where every action of every agent in a MATSim simulation run is recorded. Each boarding, alighting, as well as origins and destinations of trips are recorded

---

[1] As an example, the 2020 timetable: https://company.sbb.ch/content/dam/internet/corporate/de/sbb-als-geschaeftspartner/flotte-unterhalt/onestopshop/bezeichnung_der_zuege_2020.pdf

for every agent. Utilities in MATSim are measured not only by the negative utilities of travel but also by the positive utilities of undertaking activities during the day. In the MATSim simulation framework agents try different daily schedules (variation of activities as well as modes and route-choices) to get to these activities. A shorter or longer travel time (e.g. due to delays) will therefore lead to a corresponding increase or decrease in the daily score of an agent. For our study, aggregation needs to be performed so to identify the number of individuals on each train on each cross-section. In more details, the following aggregated results can be quantified: passengers boarding per train and station, passengers per train and station, cross-sectional loads on trains.

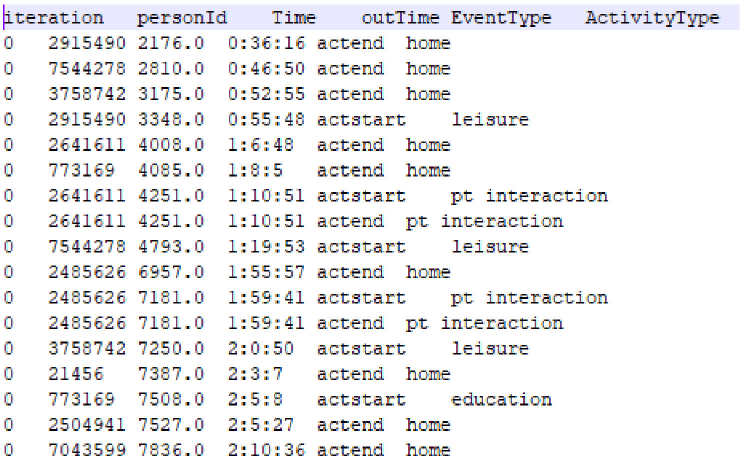
Instead of reading “events.xml” directly, the “eventHandler” is used to calculate agents’ entire journey. More specifically, one agent’s whole-day journey consists of sequential activities and trips. For the agents who use public transport trips, they need to board and alight vehicles. Therefore, four types of “eventHandler” are specifically used in this project to collect passenger data from MATSim, namely “VehicleArrivesAtFacilityEvent”, “VehicleDepartsAtFacilityEvent”, “ActivityStartEvent” and “ActivityEndEvent”.



```
iteration   personId    Time    outTime EventType   vehicleid   Station transitLine transitRoute    transitMode
0   [3627586]   16560.0 4:36:0  departureAt 39137   Winkel, Oberdorf    530_000801  11698_1_19  bus
0   [3627586]   16562.0 4:36:2  arriveAt    39137   Winkel, Bühlhof 530_000801  11698_1_19  bus
0   [2723420]   16680.0 4:38:0  departureAt 39449   Embrach, Post   520_000801  11499_1_16  bus
0   [3627586]   16680.0 4:38:0  departureAt 39137   Winkel, Bühlhof 530_000801  11698_1_19  bus
0   [3627586]   16682.0 4:38:2  arriveAt    39137   Winkel, Oberrüti    530_000801  11698_1_19  bus
0   [2723420]   16682.0 4:38:2  arriveAt    39449   Embrach, Gemeindehaus   520_000801  11499_1_16  bus
0   [2723420]   16740.0 4:39:0  departureAt 39449   Embrach, Gemeindehaus   520_000801  11499_1_16  bus
0   [3627586]   16740.0 4:39:0  departureAt 39137   Winkel, Oberrüti    530_000801  11698_1_19  bus
0   [3627586]   16742.0 4:39:2  arriveAt    39137   Winkel, Wisental    530_000801  11698_1_19  bus
0   [2723420]   16742.0 4:39:2  arriveAt    39449   Embrach, Zürcherstrasse 520_000801  11499_1_16  bus
0   [1567380]   16800.0 4:40:0  departureAt 47935   Engelburg, Schulhaus    120_000801  1032_1_9    bus
0   [2723420]   16800.0 4:40:0  departureAt 39449   Embrach, Zürcherstrasse 520_000801  11499_1_16  bus
0   [2723420]   16802.0 4:40:2  arriveAt    39449   Lufingen, Unterdorf 520_000801  11499_1_16  bus
0   [1567380]   16802.0 4:40:2  arriveAt    47935   Engelburg, Freihof  120_000801  1032_1_9    bus
0   [2723420]   16860.0 4:41:0  departureAt 39449   Lufingen, Unterdorf 520_000801  11499_1_16  bus
0   [3627586]   16860.0 4:41:0  departureAt 39137   Winkel, Wisental    530_000801  11698_1_19  bus
0   [1567380]   16860.0 4:41:0  departureAt 47935   Engelburg, Freihof  120_000801  1032_1_9    bus
```

Data 3 Example of eventHandler about vehicles

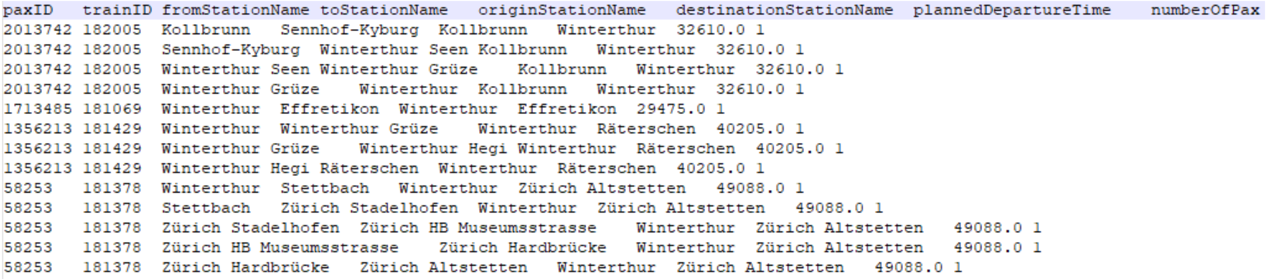
Based on the eventHandler about vehicles, the output from MATSim is capable to record each agent’s boarding and alighting time of specific vehicles at specific stations. The other features like tranLine ID, trainsRoute ID and tranistMode are recorded for more detailed analysis.



```
iteration   personId    Time    outTime EventType   ActivityType
0   2915490 2176.0  0:36:16 actend  home
0   7544278 2810.0  0:46:50 actend  home
0   3758742 3175.0  0:52:55 actend  home
0   2915490 3348.0  0:55:48 actstart    leisure
0   2641611 4008.0  1:6:48  actend  home
0   773169  4085.0  1:8:5   actend  home
0   2641611 4251.0  1:10:51 actstart    pt interaction
0   2641611 4251.0  1:10:51 actend  pt interaction
0   7544278 4793.0  1:19:53 actstart    leisure
0   2485626 6957.0  1:55:57 actend  home
0   2485626 7181.0  1:59:41 actstart    pt interaction
0   2485626 7181.0  1:59:41 actend  pt interaction
0   3758742 7250.0  2:0:50  actstart    leisure
0   21456   7387.0  2:3:7   actend  home
0   773169  7508.0  2:5:8   actstart    education
0   2504941 7527.0  2:5:27  actend  home
0   7043599 7836.0  2:10:36 actend  home
```

Data 4 Example of eventHandler about activities

The eventHandler about activites records the details about each agent’s activities, including activity type, start time and end time.



```
paxID   trainID fromStationName toStationName   originStationName   destinationStationName  plannedDepartureTime    numberOfPax
2013742 182005  Kollbrunn   Sennhof-Kyburg  Kollbrunn   Winterthur  32610.0 1
2013742 182005  Sennhof-Kyburg  Winterthur Seen Kollbrunn   Winterthur  32610.0 1
2013742 182005  Winterthur Seen Winterthur Grüze    Kollbrunn   Winterthur  32610.0 1
2013742 182005  Winterthur Grüze    Winterthur  Kollbrunn   Winterthur  32610.0 1
1713485 181069  Winterthur  Effretikon  Winterthur  Effretikon  29475.0 1
1356213 181429  Winterthur  Winterthur Grüze    Winterthur  Räterschen  40205.0 1
1356213 181429  Winterthur Grüze    Winterthur Hegi Winterthur  Räterschen  40205.0 1
1356213 181429  Winterthur Hegi Räterschen  Winterthur  Räterschen  40205.0 1
58253   181378  Winterthur  Stettbach   Winterthur  Zürich Altstetten   49088.0 1
58253   181378  Stettbach   Zürich Stadelhofen  Winterthur  Zürich Altstetten   49088.0 1
58253   181378  Zürich Stadelhofen  Zürich HB Museumsstrasse    Winterthur  Zürich Altstetten   49088.0 1
58253   181378  Zürich HB Museumsstrasse    Zürich Hardbrücke   Winterthur  Zürich Altstetten   49088.0 1
58253   181378  Zürich Hardbrücke   Zürich Altstetten   Winterthur  Zürich Altstetten   49088.0 1
```

Data 5 Example of MATSim outputs to TMS from events-file processor

In the process of running MATSim scenarios, passenger data is recorded by these eventHandler. The results of the 100th iteration of MATSim run are re-written in the format that Optimizer can read as inputs. The data format of output from Events-file processor records the passengerID, trainID, original Station, destination Station, all the pass stations, the planned departure time from the origin station. Moreover, the selected data are based on the matched trains generated by OpenTrack-MATSim matcher.

3.3. Passenger delay calculator

The “Passenger delay calculator” processes the data described above to derive two delay key figures. The first is cross-sectional passenger delay, namely the delay of a train upon departing a station and passing every infrastructure element, multiplied by the number of passengers within the train. The second is passenger arrival delay. This is the delay of a train upon arrival at a station multiplied by the number of passengers alighting the train. Moreover, the daily score of passengers can be also used as an indicator to evaluate the impact of train delays for their daily overview satisfaction.

Figure 7 presents a detailed daily travel chain of the agents (y-axis) across their entire day. The diagram shows each transport mode, in the course of a day (x-axis). We focus only on agents, which are onboard the same service, departing the station Winterthur shortly after 8 o‘clock, see the time axis corresponding on the orange lines. In other terms, it is possible to zoom in the specific agents on a service, and understand where they come from, where they go to, and what is the impact of a delay on the specific link, to further downstream activities, or transfers to other public transport services. We select three classifications based on alternative mode/route choices (i.e. rail, bus/tram, car/bike, respectively yellow, green and pink lines), as well as typical agents’ activities (e.g. home, work or others like shopping). Based on this, different level of passenger delays (stage-based or trip-based, etc.) as well as aggregated passenger delays per train/ station or route section can be quantified later.

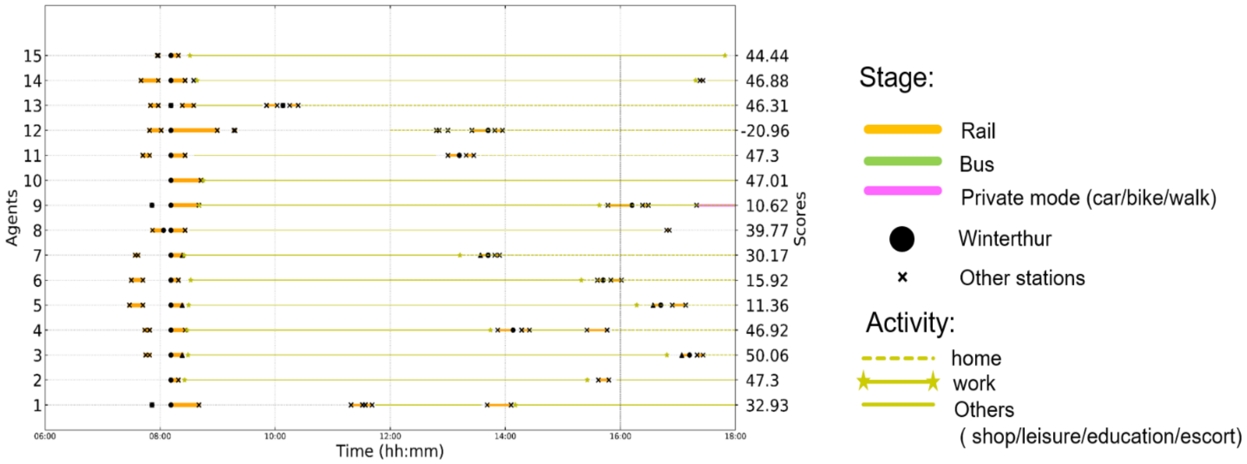


Figure 7 Passengers' daily travel chain from MATSim

The indicators for passenger delay can be examined in different levels based on this showed train chain, which can provide different evaluation for other specific projects. For instance,

- Stage-based delay. In the classical MATSim definition, a stage means one agent travels on one continuous vehicle. That means the passengers who are identified as "users of the railway system" may only be taking part of their journeys on a train, and/or they may try to use a combination of railway and bus together to go back home from work. This stage-based delay is very important to understand the quality of an operation system.

- Trip-based delay. This is the delay between two activities of a passenger, which cares more about the quality of direct trips of passengers. Different from stage-based delay, this is more about passengers' purpose of travel. In other words, if only one stage is delayed, but the entire trip is not delayed too much, perhaps passengers are possibly still satisfied.

- Whole-day delay. This could be also an option, because passengers may ignore their unpleasant feeling from a few-second delay of one trip in a 24-hour period.

- Score. This is a general but comprehensive index for one passengers' journey during the entire day, including the satisfaction about trips and activities.

All this can be used as evaluation of the performance of railway system, taking the advantage of combining railway simulation and passenger simulation. It can also be integrated to the traffic management system (TMS) algorithm to optimize dispatching strategies with the objective of minimizing passenger delays.

3.4. Passenger-oriented TMS algorithm

The optimizer is interconnected with the framework by means of input and output files, which have specific variables and parameters. We wrap the optimization approaches (the yellow diamonds in Figure 2) behind standard input and output files, and some generic specifications. In this case, the optimization method reads out the infrastructure and the planned timetable from OpenTrack (see Section 3.5), as well as passenger data from MATSim (see Section 3.6). This algorithm uses passenger route choices (i.e., a sequence of train services) as an input to perform passenger-oriented dispatching calculations and sends commands back to OpenTrack. When constructing mathematical formulations, the impact of control actions (change of schedules) on the attributes and the effect of these alternative passenger choices must be identified and represented, as well as the impact of passenger responses on control actions, such as retiming and reordering. The details of the optimization showcased are presented in Section 4.

3.5. OpenTrack data converting for TMS

This section describes the converting process of the OpenTrack data for the TMS algorithm, including the network data, the itinerary-path-route data, and the timetable data. Besides, we also introduce the data structure, the reserve and release rules, and the estimation of the minimum running times of trains on microscopic links (which are needed in the TMS algorithm, but not really used in OpenTrack).

3.5.1. Network data

OpenTrack describes a railway network in a special graph called double-vertex graph, while the TMS algorithm uses a single-vertex graph, see Figure 8. Vertices mark the points in the railway network where at least one route attribute (gradient, radius, speed, etc.) changes or where there is a signal or a station. An edge connects two adjacent vertices, corresponding to a track unit. The vertices and edges describe the railway infrastructure microscopically. At the macroscopic level, every station is described as a reference point, i.e., a station vertex, connected with other ordinary vertices by edges, see Figure 9. In such a way, the microscopic and macroscopic levels are interconnected.

Figure 8 illustrates the difference between the double-vertex graph and the single-vertex graph, using an example of a switch. To distinguish these two kinds of graphs and also to avoid misleading, we denote the elements in the single-vertex graph by *nodes* and *links*. More specifically, a node represents a double-vertex (two paired vertices), and a link corresponds to an edge, connecting two nodes in pair, see the illustration in Figure 8. At the macroscopic level, the same, we define station nodes/links, see Figure 9(b).

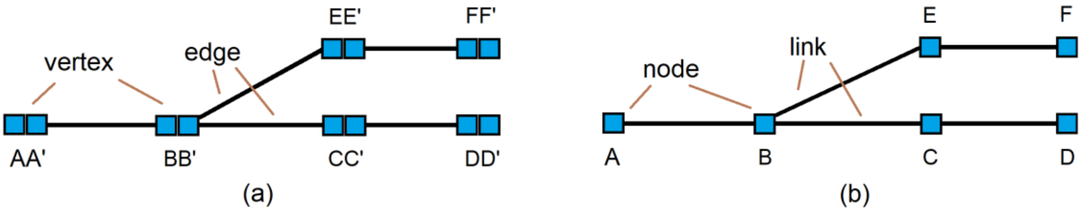


Figure 8 Vertex graph: (a) double-vertex in Opentrack; (b) single-vertex in TMS

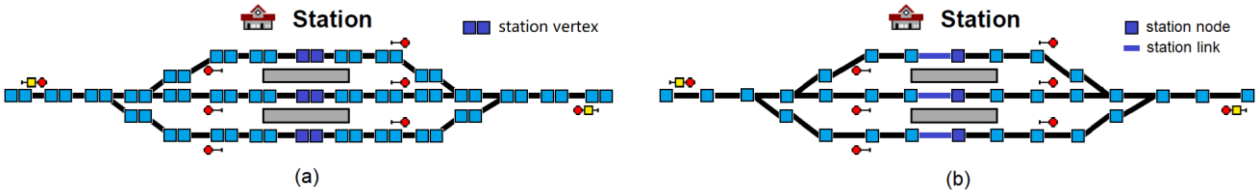


Figure 9 Station vertex: (a) double-vertex in Opentrack; (b) single-vertex in TMS

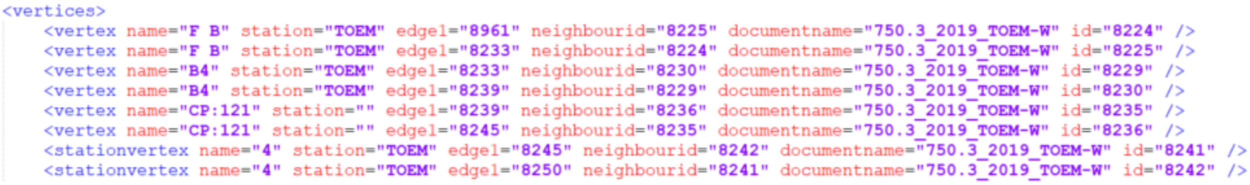
In OpenTrack, every element of the graph holds various attributes. The attributes used to identify the railway infrastructure network are as follows:

- name, station, edge1, neighbourid, documentname, and id, for a vertex;

- vertex1, vertex2, length, and id, for an edge.

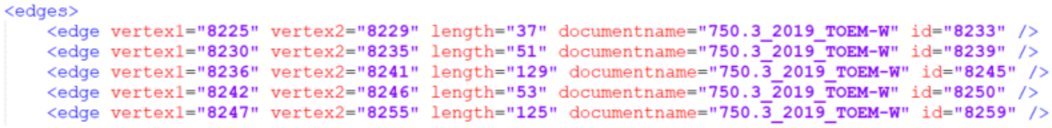
The attribute “neighbourid” indicates the “id” of the paired vertex (e.g., for a double-vertex AA', the “id” of vertex A' is given as the “neighbourid” of vertex A). The “edge1” gives the “id” of the edge connecting the vertex with another. The vertex representing a station is labeled as a “stationvertex”. For an edge, the “vertex1” and “vertex2” give the “id” of the two vertices that the edge connects, e.g., the “id” of vertex A’ and vertex B for edge A’B in Figure 8(a).

All this information is available in the OpenTrack infrastructure file, can be exported directly as an XML file. The data format can be seen as follows:



```
<vertices>
    <vertex name="F B" station="TOEM" edge1="8961" neighbourid="8225" documentname="750.3_2019_TOEM-W" id="8224" />
    <vertex name="F B" station="TOEM" edge1="8233" neighbourid="8224" documentname="750.3_2019_TOEM-W" id="8225" />
    <vertex name="B4" station="TOEM" edge1="8233" neighbourid="8230" documentname="750.3_2019_TOEM-W" id="8229" />
    <vertex name="B4" station="TOEM" edge1="8239" neighbourid="8229" documentname="750.3_2019_TOEM-W" id="8230" />
    <vertex name="CP:121" station="" edge1="8239" neighbourid="8236" documentname="750.3_2019_TOEM-W" id="8235" />
    <vertex name="CP:121" station="" edge1="8245" neighbourid="8235" documentname="750.3_2019_TOEM-W" id="8236" />
    <stationvertex name="4" station="TOEM" edge1="8245" neighbourid="8242" documentname="750.3_2019_TOEM-W" id="8241" />
    <stationvertex name="4" station="TOEM" edge1="8250" neighbourid="8241" documentname="750.3_2019_TOEM-W" id="8242" />
```

Data 6 Example of vertex data file



```
<edges>
    <edge vertex1="8225" vertex2="8229" length="37" documentname="750.3_2019_TOEM-W" id="8233" />
    <edge vertex1="8230" vertex2="8235" length="51" documentname="750.3_2019_TOEM-W" id="8239" />
    <edge vertex1="8236" vertex2="8241" length="129" documentname="750.3_2019_TOEM-W" id="8245" />
    <edge vertex1="8242" vertex2="8246" length="53" documentname="750.3_2019_TOEM-W" id="8250" />
    <edge vertex1="8247" vertex2="8255" length="125" documentname="750.3_2019_TOEM-W" id="8259" />
```

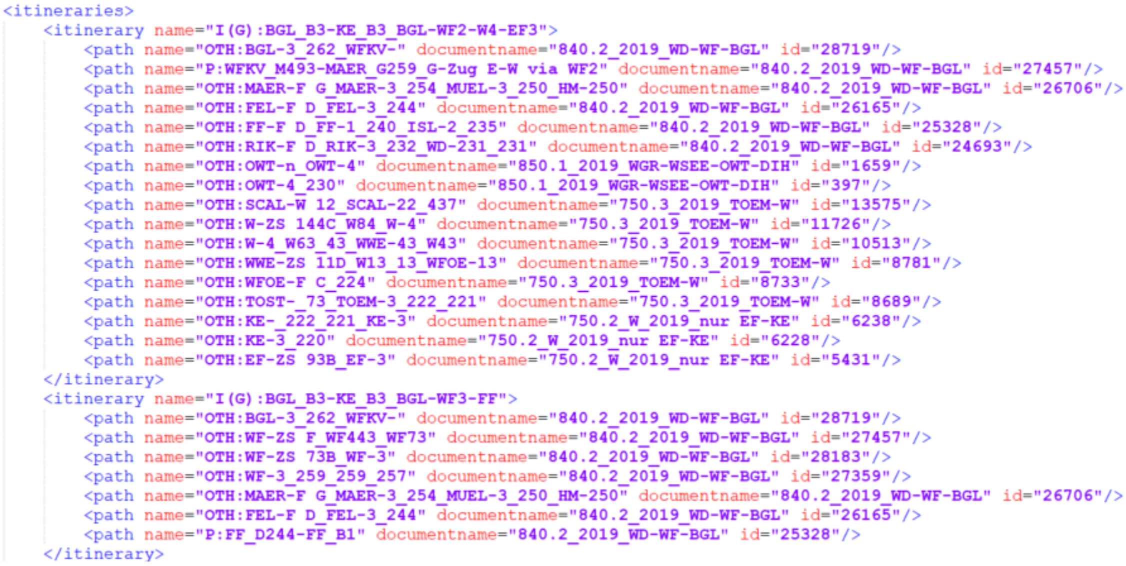
Data 7 Example of edge data file

It worth noting that there can be multiple OpenTrack documents, containing different parts of the railway network. One should open all relevant documents at the same time, in order to export the infrastructure xml file with the full information. Moreover, one may find repeated “id” of vertices and edges in different documents. Therefore, we recommend and use a combination of “id” and “documentname” to identify the physical vertices and edges.

In such a way, we can read out the railway network data from OpenTrack, and we translate the railway network into a node-link representation, based on which the TMS algorithm is developed.

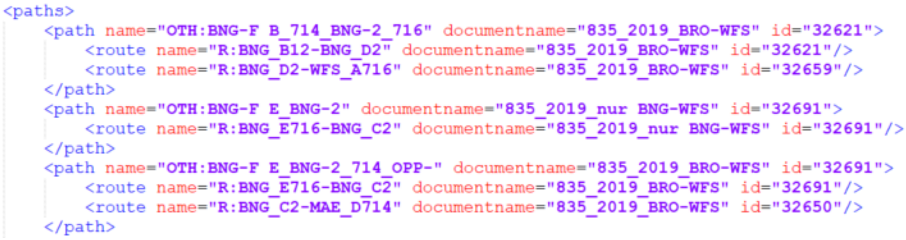
3.5.2. Routes, paths, and itineraries

In OpenTrack, a structure of “itinerary-path-route-vertex” is used to describe the tracks traversed by a train in its operation. A route consists of an order of vertices of one direction of travel. A path comprises one or multiple connected routes, and a (linked) list of paths is concatenated to form an itinerary. The information about the routes, paths, and itineraries can be exported as XML files from OpenTrack, with the following data format:



```xml
<itineraries>
    <itinerary name="I(G):BGL_B3-KE_B3_BGL-WF2-W4-EF3">
        <path name="OTH:BGL-3_262_WFKV-" documentname="840.2_2019_WD-WF-BGL" id="28719"/>
        <path name="P:WFKV_M493-MAER_G259_G-Zug E-W via WF2" documentname="840.2_2019_WD-WF-BGL" id="27457"/>
        <path name="OTH:MAER-F G_MAER-3_254_MUEL-3_250_HM-250" documentname="840.2_2019_WD-WF-BGL" id="26706"/>
        <path name="OTH:FEL-F D_FEL-3_244" documentname="840.2_2019_WD-WF-BGL" id="26165"/>
        <path name="OTH:FF-F D_FF-1_240_ISL-2_235" documentname="840.2_2019_WD-WF-BGL" id="25328"/>
        <path name="OTH:RIK-F D_RIK-3_232_WD-231_231" documentname="840.2_2019_WD-WF-BGL" id="24693"/>
        <path name="OTH:OWT-n_OWT-4" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="1659"/>
        <path name="OTH:OWT-4_230" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="397"/>
        <path name="OTH:SCAL-W 12_SCAL-22_437" documentname="750.3_2019_TOEM-W" id="13575"/>
        <path name="OTH:W-ZS 144C_W84_W-4" documentname="750.3_2019_TOEM-W" id="11726"/>
        <path name="OTH:W-4_W63_43_WWE-43_W43" documentname="750.3_2019_TOEM-W" id="10513"/>
        <path name="OTH:WWE-ZS 11D_W13_13_WFOE-13" documentname="750.3_2019_TOEM-W" id="8781"/>
        <path name="OTH:WFOE-F C_224" documentname="750.3_2019_TOEM-W" id="8733"/>
        <path name="OTH:TOST-_73_TOEM-3_222_221" documentname="750.3_2019_TOEM-W" id="8689"/>
        <path name="OTH:KE-_222_221_KE-3" documentname="750.2_W_2019_nur EF-KE" id="6238"/>
        <path name="OTH:KE-3_220" documentname="750.2_W_2019_nur EF-KE" id="6228"/>
        <path name="OTH:EF-ZS 93B_EF-3" documentname="750.2_W_2019_nur EF-KE" id="5431"/>
    </itinerary>
    <itinerary name="I(G):BGL_B3-KE_B3_BGL-WF3-FF">
        <path name="OTH:BGL-3_262_WFKV-" documentname="840.2_2019_WD-WF-BGL" id="28719"/>
        <path name="OTH:WF-ZS F_WF443_WF73" documentname="840.2_2019_WD-WF-BGL" id="27457"/>
        <path name="OTH:WF-ZS 73B_WF-3" documentname="840.2_2019_WD-WF-BGL" id="28183"/>
        <path name="OTH:WF-3_259_259_257" documentname="840.2_2019_WD-WF-BGL" id="27359"/>
        <path name="OTH:MAER-F G_MAER-3_254_MUEL-3_250_HM-250" documentname="840.2_2019_WD-WF-BGL" id="26706"/>
        <path name="OTH:FEL-F D_FEL-3_244" documentname="840.2_2019_WD-WF-BGL" id="26165"/>
        <path name="P:FF_D244-FF_B1" documentname="840.2_2019_WD-WF-BGL" id="25328"/>
    </itinerary>
```

Data 8 Example of itinerary data file



```xml
<paths>
    <path name="OTH:BNG-F B_714_BNG-2_716" documentname="835_2019_BRO-WFS" id="32621">
        <route name="R:BNG_B12-BNG_D2" documentname="835_2019_BRO-WFS" id="32621"/>
        <route name="R:BNG_D2-WFS_A716" documentname="835_2019_BRO-WFS" id="32659"/>
    </path>
    <path name="OTH:BNG-F E_BNG-2" documentname="835_2019_nur BNG-WFS" id="32691">
        <route name="R:BNG_E716-BNG_C2" documentname="835_2019_nur BNG-WFS" id="32691"/>
    </path>
    <path name="OTH:BNG-F E_BNG-2_714_OPP-" documentname="835_2019_BRO-WFS" id="32691">
        <route name="R:BNG_E716-BNG_C2" documentname="835_2019_BRO-WFS" id="32691"/>
        <route name="R:BNG_C2-MAE_D714" documentname="835_2019_BRO-WFS" id="32650"/>
    </path>
```

Data 9 Example of path data file

```xml
<routes>
    <route name="R:HET_H P732- " documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10575">
        <aspects>
            <aspect code="TS"/>
        </aspects>
        <vertex name="H P732" station="HET" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10575"/>
        <vertex name="" station="HET" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10580"/>
        <vertex name="MT vStrecke" station="HET" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10667"/>
        <vertex name="" station="HET" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10691"/>
        <vertex name="W11" station="HET" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10613"/>
        <vertex name="" station="HET" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10584"/>
        <vertex name="CP:933" station="" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10588"/>
        <vertex name="MT vStrecke" station="HET" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10679"/>
        <vertex name="933" station="HET" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10697"/>
        <vertex name="H S933" station="HET" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10592"/>
        <vertex name="" station="HET" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10597"/>
        <vertex name="" station="HET" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10654"/>
        <stationvertex name="933" station="HET" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10601"/>
        <vertex name=" " station="" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10605"/>
    </route>
    <route name="R:HET_H P732- .1" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10575">
        <aspects>
            <aspect code="60"/>
        </aspects>
        <vertex name="H P732" station="HET" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10575"/>
        <vertex name="" station="HET" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10580"/>
        <vertex name="MT vStrecke" station="HET" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10667"/>
        <vertex name="" station="HET" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10691"/>
        <vertex name="W11" station="HET" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10613"/>
        <vertex name="" station="HET" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10617"/>
        <vertex name="CP:733" station="" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10621"/>
        <vertex name="MT vStrecke" station="HET" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10675"/>
        <vertex name="733" station="HET" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10701"/>
        <vertex name="S733" station="HET" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10625"/>
        <vertex name="" station="HET" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10630"/>
        <vertex name="" station="HET" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10650"/>
        <stationvertex name="733" station="HET" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10634"/>
        <vertex name=" " station="" documentname="850.1_2019_WGR-WSEE-OWT-DIH" id="10640"/>
    </route>
```

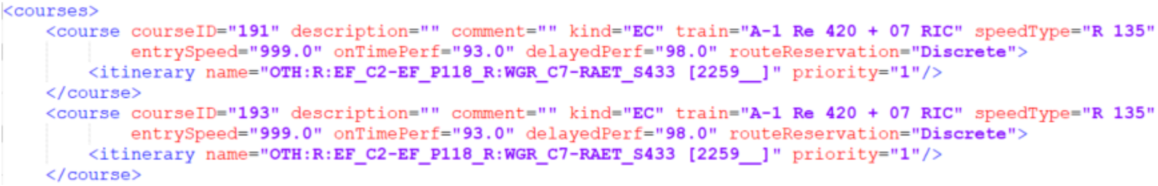
Data 10 Example of route data file

As shown, the attributes, associated with each itinerary, path, and route, include name, id, and documentname. As before, the “id” of those objects is not unique. We suggest and use a combination of “name”, “documentname”, and “id” to identify them, in order to avoid mistakes.

Based on the vertex-node translator of the railway network (see Section 3.5.1), we can then describe each route/path/itinerary by a sequence of links, to be used in the TMS model.

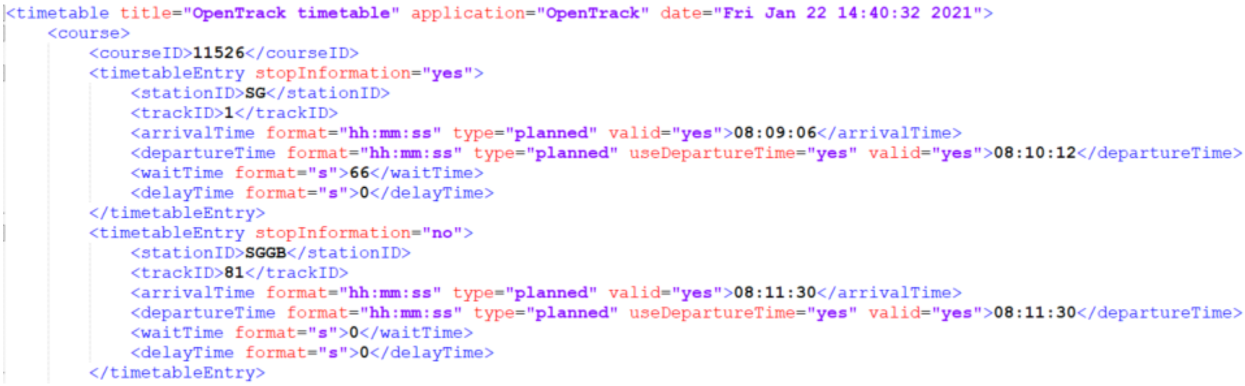
3.5.3. Timetable data

OpenTrack uses the term “course” to define a service operated by a train vehicle over a period of time. Each train course is associated with one itinerary or a set of itineraries (see Section 3.5.2), with an associated set of timetable entries, which specify the departure time, arrival time, dwell time, connections, and other key data for the train movements, at each passing station. The OpenTrack course and timetable can also be exported as XML files, in a following format:



```
<courses>
    <course courseID="191" description="" comment="" kind="EC" train="A-1 Re 420 + 07 RIC" speedType="R 135"
            entrySpeed="999.0" onTimePerf="93.0" delayedPerf="98.0" routeReservation="Discrete">
        <itinerary name="OTH:R:EF_C2-EF_P118_R:WGR_C7-RAET_S433 [2259__]" priority="1"/>
    </course>
    <course courseID="193" description="" comment="" kind="EC" train="A-1 Re 420 + 07 RIC" speedType="R 135"
            entrySpeed="999.0" onTimePerf="93.0" delayedPerf="98.0" routeReservation="Discrete">
        <itinerary name="OTH:R:EF_C2-EF_P118_R:WGR_C7-RAET_S433 [2259__]" priority="1"/>
    </course>
```

Data 11 Example of train course data file



```
<timetable title="OpenTrack timetable" application="OpenTrack" date="Fri Jan 22 14:40:32 2021">
    <course>
        <courseID>11526</courseID>
        <timetableEntry stopInformation="yes">
            <stationID>SG</stationID>
            <trackID>1</trackID>
            <arrivalTime format="hh:mm:ss" type="planned" valid="yes">08:09:06</arrivalTime>
            <departureTime format="hh:mm:ss" type="planned" useDepartureTime="yes" valid="yes">08:10:12</departureTime>
            <waitTime format="s">66</waitTime>
            <delayTime format="s">0</delayTime>
        </timetableEntry>
        <timetableEntry stopInformation="no">
            <stationID>SGGB</stationID>
            <trackID>81</trackID>
            <arrivalTime format="hh:mm:ss" type="planned" valid="yes">08:11:30</arrivalTime>
            <departureTime format="hh:mm:ss" type="planned" useDepartureTime="yes" valid="yes">08:11:30</departureTime>
            <waitTime format="s">0</waitTime>
            <delayTime format="s">0</delayTime>
        </timetableEntry>
```

Data 12 Example of OpenTrack timetable data file

Each train course holds various attributes, among which the “courseID” and the “itinerary name” are relevant to the TMS algorithm. With the timetable data file, we are able to get the information about the planned arrival and departure times, and the minimum dwell times of trains at passing stations, which are used as reference in the TMS algorithm, to calculate train delays and to restrict train dwell activities at stations.

3.5.4. Reserve and release rules

In OpenTrack, safety elements are sections of track that can generally be occupied by only one train at a time. A safety element comprises one or several edges (with their start and end vertices) of the network that can only be released or reserved together. OpenTrack automatically generates safety elements, and the safety elements are not visible on the worksheet but are generated behind the track layout. One can edit (e.g., merge or reset) the safety elements in OpenTrack based on the needs.

In addition, OpenTrack provides different setting options to change the reserve and release rules of track segments. For instance, one can select “Reserve with Previous” attribute to a route, which implements that “the route is reserved if the previous route of the train movement is reserved”.

In the TMS algorithm, we simply follow the default setting, i.e., letting all links of a route be reserved and released at the same time.

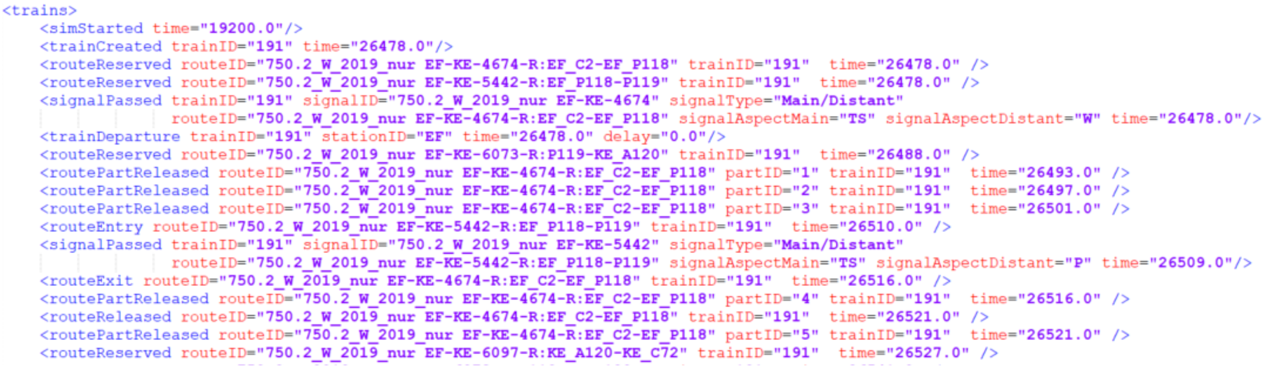
3.5.5. Estimation of the minimum train running time

OpenTrack simulates train movements using motion equations based on the characteristics of vehicle engine (e.g., tractive effort, acceleration, and braking rate), train (e.g., mass, running speed, speed limits), track (e.g., distance, gradient, curve radius, and speed limits), and so on. In the TMS algorithm, the minimum train running times are used to ensure that train movements respect the technical requirements. The minimum train running time cannot be generated directly by OpenTrack since it calculates the train motion with details, but it is needed in the TMS algorithm.

To estimate the minimum running time of trains on links (i.e., edges in OpenTrack, the minimal track units), we simulate each train course separately by using OpenTrack API. By doing this, each train can traverse the predefined routes, without being affected by the others, and we can get the train movement information, given as the OpenTrack responses (i.e., status messages). There are many different types of responses. Here, the most useful responses are “routeEntry” and “routeExit”, which indicate respectively the time that the train enters and exits the corresponding route. If no such response is given, we then use the responses of “signalPassed”, and “routeReleased”, and we assume that

- “routeEntry” time = “signalPassed” time + 1 s;
- “routeExit” time = “routeReleased” time - 5 s.

As a result, we can calculate the minimum running time of trains on each route, i.e., the difference between “routeExit” time and “routeEntry” time. An example of the OpenTrack responses is given below:



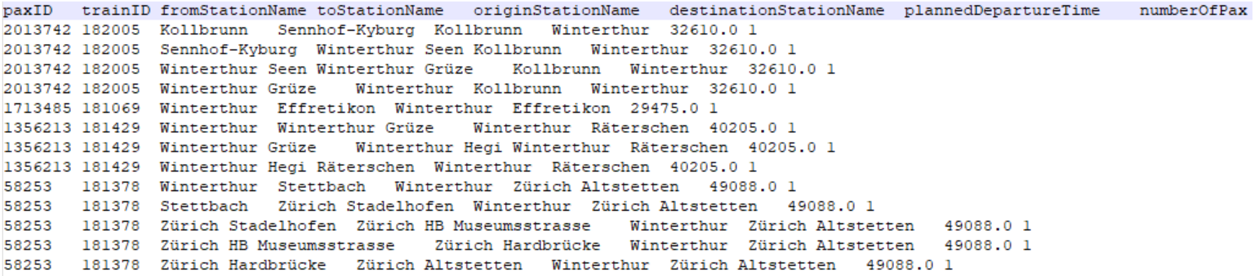
```xml
<trains>
    <simStarted time="19200.0"/>
    <trainCreated trainID="191" time="26478.0"/>
    <routeReserved routeID="750.2_W_2019_nur EF-KE-4674-R:EF_C2-EF_P118" trainID="191"  time="26478.0" />
    <routeReserved routeID="750.2_W_2019_nur EF-KE-5442-R:EF_P118-P119" trainID="191"  time="26478.0" />
    <signalPassed trainID="191" signalID="750.2_W_2019_nur EF-KE-4674" signalType="Main/Distant"
                  routeID="750.2_W_2019_nur EF-KE-4674-R:EF_C2-EF_P118" signalAspectMain="TS" signalAspectDistant="W" time="26478.0"/>
    <trainDeparture trainID="191" stationID="EF" time="26478.0" delay="0.0"/>
    <routeReserved routeID="750.2_W_2019_nur EF-KE-6073-R:P119-KE_A120" trainID="191"  time="26488.0" />
    <routePartReleased routeID="750.2_W_2019_nur EF-KE-4674-R:EF_C2-EF_P118" partID="1" trainID="191"  time="26493.0" />
    <routePartReleased routeID="750.2_W_2019_nur EF-KE-4674-R:EF_C2-EF_P118" partID="2" trainID="191"  time="26497.0" />
    <routePartReleased routeID="750.2_W_2019_nur EF-KE-4674-R:EF_C2-EF_P118" partID="3" trainID="191"  time="26501.0" />
    <routeEntry routeID="750.2_W_2019_nur EF-KE-5442-R:EF_P118-P119" trainID="191"  time="26510.0" />
    <signalPassed trainID="191" signalID="750.2_W_2019_nur EF-KE-5442" signalType="Main/Distant"
                  routeID="750.2_W_2019_nur EF-KE-5442-R:EF_P118-P119" signalAspectMain="TS" signalAspectDistant="P" time="26509.0"/>
    <routeExit routeID="750.2_W_2019_nur EF-KE-4674-R:EF_C2-EF_P118" trainID="191"  time="26516.0" />
    <routePartReleased routeID="750.2_W_2019_nur EF-KE-4674-R:EF_C2-EF_P118" partID="4" trainID="191"  time="26516.0" />
    <routeReleased routeID="750.2_W_2019_nur EF-KE-4674-R:EF_C2-EF_P118" trainID="191"  time="26521.0" />
    <routePartReleased routeID="750.2_W_2019_nur EF-KE-4674-R:EF_C2-EF_P118" partID="5" trainID="191"  time="26521.0" />
    <routeReserved routeID="750.2_W_2019_nur EF-KE-6097-R:KE_A120-KE_C72" trainID="191"  time="26527.0" />
```

Data 13 Example of OpenTrack response (status message) file

To further estimate the minimum running time on edges, we distribute the running time on a route to the edges that constitute the route, based on the edge length. Moreover, we round up the estimated value and assume a minimum running time of 1 s on each edge.

3.6. MATSim data preprocessing for TMS

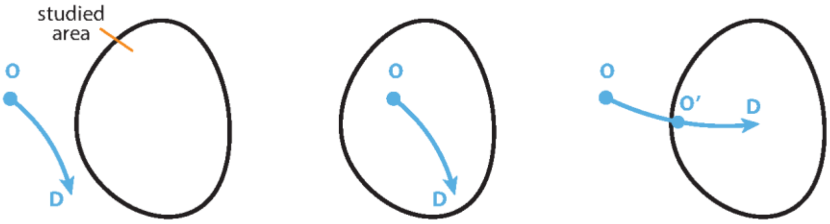
From MATSim, we get the passenger data, which indicates the travel information of each passenger/agent, more specifically, a sequence of train services that each passenger follows for reaching the destination. An example of the passenger data from MATSim is given below:



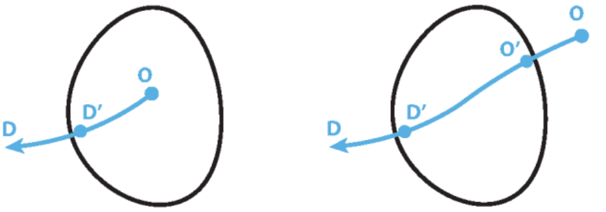
```
paxID   trainID fromStationName toStationName   originStationName   destinationStationName  plannedDepartureTime    numberOfPax
2013742 182005  Kollbrunn   Sennhof-Kyburg  Kollbrunn   Winterthur  32610.0 1
2013742 182005  Sennhof-Kyburg  Winterthur Seen Kollbrunn   Winterthur  32610.0 1
2013742 182005  Winterthur Seen Winterthur Grüze    Kollbrunn   Winterthur  32610.0 1
2013742 182005  Winterthur Grüze    Winterthur  Kollbrunn   Winterthur  32610.0 1
1713485 181069  Winterthur  Effretikon  Winterthur  Effretikon  29475.0 1
1356213 181429  Winterthur  Winterthur Grüze    Winterthur  Räterschen  40205.0 1
1356213 181429  Winterthur Grüze    Winterthur Hegi Winterthur  Räterschen  40205.0 1
1356213 181429  Winterthur Hegi Räterschen  Winterthur  Räterschen  40205.0 1
58253   181378  Winterthur  Stettbach   Winterthur  Zürich Altstetten   49088.0 1
58253   181378  Stettbach   Zürich Stadelhofen  Winterthur  Zürich Altstetten   49088.0 1
58253   181378  Zürich Stadelhofen  Zürich HB Museumsstrasse    Winterthur  Zürich Altstetten   49088.0 1
58253   181378  Zürich HB Museumsstrasse    Zürich Hardbrücke   Winterthur  Zürich Altstetten   49088.0 1
58253   181378  Zürich Hardbrücke   Zürich Altstetten   Winterthur  Zürich Altstetten   49088.0 1
```

Data 14 Example of passenger data file from MATSim

The area that MATSim focuses on is larger than that of OpenTrack (see Section 0); thus, we need to preprocess the passenger data before using it in the TMS algorithm. Figure 10 illustrates the 5 scenarios that appear in passengers' trips:



(a) discard (b) nothing (c) replace O by O'



(d) replace D by D' (e) replace O by O' and D by D'

Figure 10 Scenarios and actions taken in the preprocessing of passenger data

(1) The passenger never enters the studied area, as shown in Figure 10(a). Then, the passenger is considered to be irrelevant, and we discard the passenger.

(2) The passenger travels completely inside the studied area, as shown in Figure 10(b). Then, the passenger is regarded to be relevant, and we do nothing.

(3) The passenger departs from a station outside the studied area, but ends the trip in the studied area, see Figure 10(c). Then, we replace the origin of the passenger by the first station that the passenger visits in the studied area.

(4) The passenger departs from a station in the studied area, but the destination is not in the studied area, see Figure 10(d). Then, we replace the destination of the passenger by the last station that the passenger visits before leaving the studied area.

(5) Figure 10(e) shows that neither the origin nor the destination of the passenger are in the studied area. In such a scenario, we cut the legs outside the studied area, using respectively the first and the last station that the passenger visits in the studied area as the origin and the destination.

As a result, we actually focus only on those passenger trips that are related to the train services under consideration (spatially and temporally). Those irrelevant passengers and trip legs are discarded. Then, we gather the passengers who have the same OD and choose exactly the same train services into a group. By doing this, the number of passengers can be reduced, so does the problem scale. A group of passengers is then identified by the origin, destination, route choice (a sequence of train services), and the number of passengers in the group.

3.7. TMS timetable acceptance in OpenTrack

The result of the TMS algorithm, i.e., the rescheduled timetable, is written in an XML format of the OpenTrack timetable, see the timetable data format given in Section 3.5.3. The XML timetable file can be imported into OpenTrack, either fully overwriting the existing timetable or partially updating the timetable entries of the train courses. The departure and arrival times in the rescheduled timetable should be written as “planned”, rather than “actual”, in order to be correctly accepted by the OpenTrack.

Moreover, in the TMS algorithm, train connections are implicitly considered, by keeping passenger feasible transfers as many as possible. If a train connection is planned in the original timetable, but never used in any passenger trips, then the train connection is neglected and may not be kept in the rescheduled timetable. Also, the TMS algorithm does not generate any result about train connections; thus, no train connection information is provided in the generated TMS timetable. However, in OpenTrack simulation, some train connections may be enforced: the successive train course cannot depart if the preceding train course has not arrived. As a result, no simulation result can be obtained for the successive train, even if its operating time is fully covered by the simulated time window. Depending on the situation and the necessity, one may need to disable the train connections, in order to get the simulation results for those successive trains.

Another way of sending the TMS results to OpenTrack, is using OpenTrack API. The TMS results can be written as a list of dispatching commands sent to OpenTrack, in order to perform the dispatching actions and thus control the train traffic during the simulation process. We did not use this method in the present report but could be applied in the future study.

4. Optimization of traffic management: train-oriented and user-oriented

The TMS algorithm used in this project is based on the work presented in Luan et al. (2017, 2020). Its task is to reschedule the original timetable in order to resolve conflicts and to reduce (train or passenger) delays, which have been caused by various perturbations.

The TMS algorithm formulates the rescheduling process as a mixed-integer linear programming (MILP) model, based on the time-instance formulation method. The TMS model considers passenger demand data (i.e., volumes on board services) as input, which is exported from MATSim (see the description in Section 3.6). In addition to the passenger data, further inputs to the TMS model include the infrastructure information (position of signals and stations), the traffic characteristics (origin, destination, route, minimum travel time, etc.), and the original timetable, which are all converted or computed based on the OpenTrack data (see the details in Section 3.5).

In OpenTrack, the railway network is described by (double) vertices and edges, which are represented by nodes and links in the TMS algorithm correspondingly. Each train traverses the network from its origin, via some intermediate stations, and to its destination, identified by a sequence of routes (i.e., an itinerary) in OpenTrack. A route further comprises a sequence of vertices/edges, which are blocked and released at the same time. We follow the same rules in the TMS: 1) a link connects two adjacent nodes (i.e., vertices in OpenTrack); 2) the links composing each route are gathered into a set, blocked and released at the same time; and (3) each train traverses a sequence of routes/links connecting its origin and its destination. As such, the output of our TMS algorithm is a microscopic schedule, i.e., arrival and departure times for each link, with a guarantee of the solution feasibility at the microscopic level. Section 3.5 elaborates the converting of the OpenTrack data for the TMS algorithm. In addition, the algorithm is assumed to have perfect knowledge of all initial delays arising during the considered time horizon.

In summary, the TMS model optimizes train orders and arrival and departure times at all passing stations based on the current (initial) delay and traffic situation, as well as the estimated/planned route choices of passengers, considering two objectives:

(1) the *train-oriented* objective, i.e., the sum, over all trains, of the delay times (including early delays) at all visited stations, denoted by Z_{train};

(2) the *user-oriented* objective, i.e., the sum, over all passengers, of the delay time at their destination, while keeping the maximal number of the planned passenger routes feasible, denoted by Z_{pax},

subject to a number of constraints for ensuring the operational and safety requirements. The key constraints include: (1) Transition constraint to force the spatial and temporal transition of each train as it moves on the rail network; (2) Train travel/dwell time constraint to ensure the required minimum travel/dwell time; (3) Safety headway constraint to define the safety time interval between trains; (4) Capacity constraint to guarantee that any pair of trains using the same infrastructure (track/block) are conflict-free; and (5) Route feasibility constraint to examine whether the planned passenger routes are still feasible in the rescheduled timetable.

The formulations of the TMS algorithm are detailed in Sections 4.1-4.2. Note that the TMS algorithm can be solved by a standard MILP solver, e.g., CPLEX or Gurobi.

4.1. Notations

We next present the notations of the TMS model. Table 2 lists the sets, subscripts, input parameters, and decision variables used by the MILP model.

Table 2 Sets, subscripts, input parameters, and decision variables.

Symbol	**Description**
	Subscripts and Sets
F	set of trains, |*F*| is the number of trains
V	set of nodes, |*V*| is the number of nodes
E	set of links (i.e., block sections), $E \subseteq V \times V$, |*E*| is the number of links
S	set of stations (at macroscopic level), each station corresponds to a set of link(s)
s	station index, $s \in S$
G	set of segments, each segment is represented by a pair of stations $(s_1, s_2), s_1, s_2 \in S$
P	set of passenger groups, |*P*| is the number of passenger groups
f	train index, $f \in F$
R	set of train routes, each route consists of a sequence of links
i, j, k, l	node index, $i, j, k, l \in V$
e	link index, denoted by $e = (i, j) \in E$
p	passenger group index, $p \in P$
r	route index, $r \in R$
R_f	set of routes to be followed by train *f*
E_r	set of links constitute route *r*
E_f	set of links that train *f* may use, $E_f \subseteq E$
E_s	set of links that represent station *s*, $E_s \subseteq E$
S_f	set of stations that train *f* may visit, $S_f \subseteq S$
S_f^{stop}	set of stations, where train *f* is required to stop, $S_f^{\text{stop}} \subseteq S_f$
G_f	set of segments that train *f* may traverse, $G_f \subseteq G$
C_p	set of route choices that passenger p follows, $(f, s_1, s_2) \in R_p$, indicating that passenger *p* takes train *f* to travel through segment (s_1, s_2)
	Input Parameters
o_f	origin node of train *f*
q_f	destination node of train *f*
b_f	direction of train *f*
c_f	capacity of train *f* for carrying passengers
ξ_f	planned departure time of train *f* from its origin

γ_f^{pri}	primary delay time of train *f* at its origin node
$\varpi_{f,s}$	planned arrival time of train *f* at station s, $s \in S_f$
$\tau_{f,i,j}^{\min}$	free flow running time of train *f* to drive through link (i,j)
$w_{f,i,j}^{\min}$	minimum dwell time of train *f* on link (i,j), $(i,j) \in E_f$
$g_{f,i,j}$	safety time interval between occupancy of link (i,j) and arrival of train *f*, including setup time, sight and reaction time, and approach time
$h_{f,i,j}$	safety time interval between departure of train *f* and release of link (i,j), including clearing time and release time
α_p	origin station of passenger group *p*
β_p	destination station of passenger group *p*
n_p	number of passengers in group *p*
λ^{trf}	minimum passenger transfer time at stations from one train to another
M	a sufficiently large positive number
ε	a sufficiently small positive number
Decision Variables	
$a_{f,i,j}$	arrival time of train *f* at link (i,j), arrival time of train *f* at station s is indicated by $A_{f,s}$
$d_{f,i,j}$	departure time of train *f* from link (i,j), $D_{f,s}$ indicates departure time of train *f* at station *s*
$w_{f,i,j}$	dwell time of train *f* on link (i,j), $W_{f,s}$ indicates the actual dwell time of train *f* at station *s*
$\phi_{f,i,j}$	reserve time of link $(i,j)$for the use of train *f*
$\mu_{f,i,j}$	release time of link (i,j) for the use of train *f*
$\theta_{f_1,f_2,i,j}$	binary train ordering variables: $\theta_{f_1,f_2,i,j} = 1$ if train *f'* arrives at link (i,j) after train *f*, and otherwise $\theta_{f_1,f_2,i,j} = 0$
ρ_p	delay time of passengers in group *p*
η_p	travel time of passengers in group *p*
κ_p	binary variable, feasible route choice of passengers in group *p*, κ_p=1 if the route choice of $R_p$is still feasible in the adjusted schedule, otherwise κ_p=0

4.2. Model formulations

We formulate two objectives:

1) the sum of train arrival delays at all stations:

$$\min Z_{\text{train}} = \sum_{f\in F}\sum_{s\in S_f}\left(A_{f,s}-\varpi_{f,s}\right) \tag{1}$$

2) the sum of delays over all passengers, while maintaining as much as possible the feasible route choices in R_p for all passengers $p \in P$:

$$Z_{\text{pax}} = \sum_{p \in P} \rho_p + \sum_{p \in P} M \cdot \left(1 - \kappa_p\right) \tag{2}$$

The first term in (1), i.e., the delay of passengers in group p, can be calculated by

$$\rho_p = \min\left\{ \max\left\{ \sum_{f \in F} \sum_{i \in V:\left(f,i,\beta_p\right) \in R_p} \left(A_{f,\beta_p} - \varpi_{f,\beta_p}\right), 0 \right\}, M \cdot \kappa_p \right\}, \quad \forall p \in P \tag{3}$$

i.e., the positive arrival delay of train *f* that passengers in group p take for arriving at the destination β_p. Moreover, if the planned route in set R_p becomes infeasible, i.e., $\kappa_p = 0$, then the passengers in group *p* cannot follow their planned route and thus the delay is set to be 0. The second term in (1) penalizes the cancelled/infeasible route of passengers *p* planned in set R_p.

We consider two objective functions: one is to minimize the total train delay in (1), i.e., $\min Z_{\text{train}}$; and another one is to minimize the total passenger delay, the number of infeasible passenger routes, as well as the total train delay, i.e., $\min Z_{\text{pax}} + \omega \cdot Z_{\text{train}}$. We minimize also train delay when targeting at passengers' benefit, in order to avoid unnecessary time deviations from the original timetable. The weight ω is used to balance the importance of Z_{train} and Z_{pax}.

The following constraint ensures that trains do not leave their origins before the earliest departure time,

$$a_{f,o_f,j} \geq \xi_f + \gamma_f^{\text{pri}}, \quad \forall f \in F, (o_f, j) \in E_f \tag{4}$$

i.e., the sum of the planned departure time ξ_f and the primary delay time γ_f^{pri}.

To force the transition of a train within a link, i.e., the train departure time from a link is greater/later than its arrival time at the same link, the following constraint is proposed:

$$d_{f,i,j} \geq a_{f,i,j}, \quad \forall f \in F, (i,j) \in E_f \tag{5}$$

If two adjacent links *(i, j)* and *(j, k)* are consecutively used by train *f*, then we should ensure that the departure time of train *f* from link *(i, j)* equals its arrival time at link *(j, k)*, formulated as follows:

$$\sum_{i:(i,j) \in E_f} d_{f,i,j} \geq \sum_{k:(j,k) \in E_f} a_{f,j,k}, \quad \forall f \in F, j \in V \setminus \left\{o_f, q_f\right\} \tag{6}$$

The train dwell time constraint

$$w_{f,i,j} \geq w_{f,i,j}^{\min}, \quad \forall f \in F, (i,j) \in E_f \tag{7}$$

guarantees the required minimum dwell times at stations. The minimum dwell time is the time required to complete the processes of passengers boarding and alighting, goods loading and unloading, etc.

The train travel time constraint

$$d_{f,i,j} - w_{f,i,j} - a_{f,i,j} \geq \tau_{f,i,j}^{\min}, \quad \forall f \in F, (i,j) \in E_f \tag{8}$$

enforces the technically required minimum running time of train *f* on link (*i, j*).

The train order variables $\theta_{f_1,f_2,i,j}$ are forced by the following constraint:

$$\theta_{f_1,f_2,i,j} + \theta_{f_2,f_1,i,j} = 1, \quad \forall f_1, f_2 \in F, f_1 \neq f_2, (i,j) \in E_{f_1} \cap E_{f_2} \tag{9}$$

It ensures that either train f_2 arrives at link (*i, j*) after train f_1 or train f_1 arrives at link (*i, j*) after train f_2, for each pair of trains using the same link (where conflicts may occur).

We calculate the reserve and release time of each block by using the following constraints:

$$\phi_{f,i,j} = \min_{(k,l)\in E_f \cap E_r} \left(a_{f,i,j} - g_{f,i,j}\right), \quad \forall f \in F, r \in R_f, (i,j) \in E_f \cap E_r \tag{10}$$

$$\mu_{f,i,j} = \min_{(k,l)\in E_f \cap E_r} \left(d_{f,i,j} + h_{f,i,j}\right), \quad \forall f \in F, r \in R_f, (i,j) \in E_f \cap E_r \tag{11}$$

i.e., all the links of each route are reserved and released at the same time.

Then, with the following two constraints,

$$\phi_{f_2,i,j} + \left(1 - \theta_{f_1,f_2,i,j}\right) \cdot M \geq \mu_{f_1,i,j}, \quad \forall f_1, f_2 \in F, f_1 \neq f_2, b_{f_1} = b_{f_2}, (i,j) \in E_{f_1} \cap E_{f_2} \tag{12}$$

$$\phi_{f_2,j,i} + \left(1 - \theta_{f_1,f_2,i,j}\right) \cdot M \geq \mu_{f_1,i,j}, \quad \forall f_1, f_2 \in F, f_1 \neq f_2, b_{f_1} \neq b_{f_2}, (i,j) \in E_{f_1}, (j,i) \in E_{f_2} \tag{13}$$

we ensure that any pair of trains using one link in the same or different direction respectively are conflict-free at the microscopic level. If two trains are running on the same link, the successive train can only access to the link after the link is released for the proceeding train.

Based on the microscopic train schedule, we extract the schedule of trains at the macroscopic (station) level, by the following three constraints

$$D_{f,s} = d_{f,i,j}, \quad \forall f \in F, i, j \in V, s \in S_f, (i,j) \in E_f \cap E_s \tag{14}$$

$$A_{f,s} = d_{f,i,j} - w_{f,i,j}, \quad \forall f \in F, i, j \in V, s \in S_f, (i,j) \in E_f \cap E_s \tag{15}$$

$$W_{f,s} = D_{f,s} - A_{f,s}, \quad \forall f \in F, s \in S_f \tag{16}$$

for getting respectively the departure time, arrival time, and dwell time of train f at station *s*. The model aggregates the microscopic and macroscopic levels. The consideration of the microscopic details guarantees the feasibility of the generated train schedule. With the inclusion of the macroscopic level, we can reduce the model complexity of rerouting passengers. Because passengers' choices are at the station level: they are not allowed to board or alight from a train on any segments, only at stations.

With the adaptation of the train schedule, some route plans that were feasible in the original timetable may become infeasible, mostly because some train connections are dropped. Therefore, we use the following constraint

$$\kappa_p \leq 1 + \left(D_{f_2,s_2} - A_{f_1,s_1} - \lambda^{\min} \right) \cdot \varepsilon, \quad \forall p \in P, f_1, f_2 \in F, s_1 \in S_{f_1}, s_2 \in S_{f_1} \cap S_{f_2}, s_3 \in S_{f_2}, (f_1, s_1, s_2) \in C_p, (f_2, s_2, s_3) \in C_p \tag{17}$$

to examine whether the train connections in the planned route set C_p are still maintained. If the gap between the departure time of train f_2 from station s_2 and the arrival time of train f_1 at station s_2 is long enough, then passengers are able to transfer from train f_1 to train f_2. Otherwise, if the transfer time is not enough, i.e., $D_{f_2,s_2} - D_{f_1,s_2} - \lambda^{\mathrm{trf}} < 0$, then we enforce the binary variable κ_p to be zero, indicating that the planned route of passengers in group p is no longer possible in the rescheduled timetable.

The TMS algorithm used in this project is based on the work presented in Luan et al. (2017, 2020). We adopt the CPLEX, a commercial and standard MILP solver, to solve the TMS model. The TMS model and the data converting and preprocessing process introduced in Sections 3.5-3.6 are all implemented in Java, the same programming language to the OpenTrack API and MATSim. This means the entire framework is implemented under a Java development environment.

5. Case study

A set of experiments is performed in the Swiss Federal Railways network around the node of Winterthur. Winterthur is a medium-sized city located north of Zurich. The selection of the proposed network section is due to its intensive use and the high number of hourly conflicts at grade crossings including some of the main lines in the Swiss network. A schematic railway network line plan is presented in Figure 11. In total, there are 195 commercial stations for passenger stops (incl. the 36 stations shown in Figure 11 and other stations linked to this region) and 528 train courses operating from 5:00 to 18:00. The train frequency on most S-Bahn train services (i.e. all those beginning with S, black and green lines) is every half hour (double solid line), while that on most inter-region (IR, blue lines) and inter-city (IC, red lines) services is every hour, only a few IC services with a frequency of every two hours (dotted line).

Public transport integration is fully implemented in the Winterthur network; any multi-modal trip between origin and destination is available to public transport users without extra charge. We consider the passengers who have at least one activity (e.g. home, work, shop, etc.) slotted in the surrounding region of Winterthur (as shown by the yellow marks in Figure 11) during one day. The total number of agents is 17,378 in the MATSim simulation, representing 10% of total population in each area.

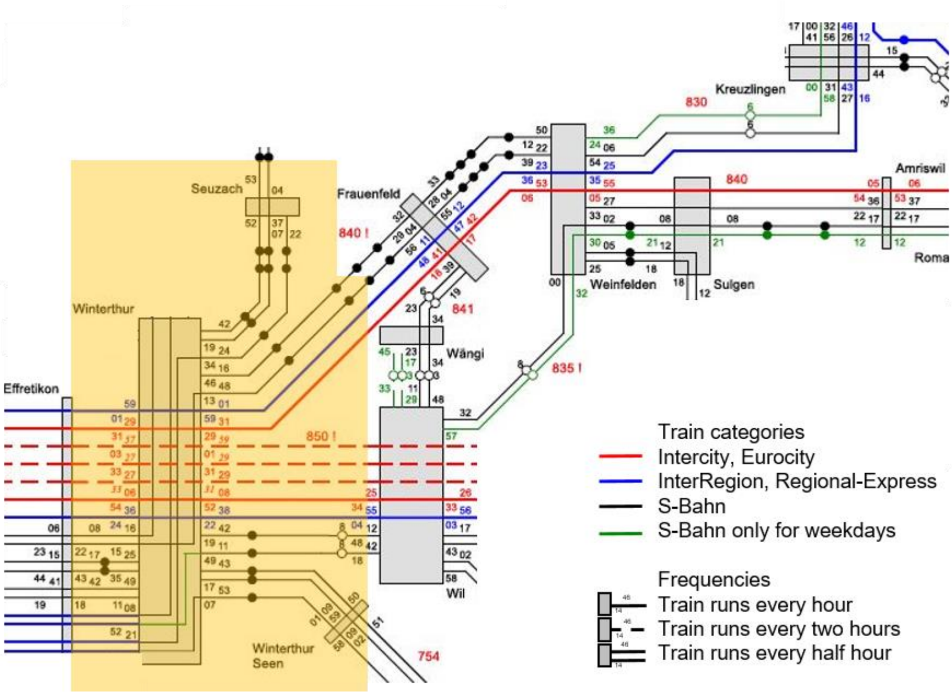


Figure 11 Winterthur railway network

Under the framework that we proposed, different test cases could be built and tested. First, the scenarios can report the benefits for combining OpenTrack and the TMS optimizer. The new timetable calculated in the TMS optimizer can update the OpenTrack timetable and further evaluate delays, delay propagation and the possible conflicts. For instance, a series of simulation runs will be conducted with and without optimization, for comparison purposes. This way, a multiplicity of conflict situations can be simulated, and the benefits of the optimization strategies evaluated in OpenTrack. Second, the scenarios can report the benefits for combining MATSim and OpenTrack. The single OpenTrack simulation run can calculate the train delays. As a comparison, MATSim can quantify the passenger delays corresponding to train delays. In this way, train and passenger delays can be quantified and evaluated at the scale of a large network. Third, combining the OpenTrack-MATSim-Optimizer architecture can evaluate the added value of passenger-oriented dispatching strategies to increased passenger satisfaction on densely used railway networks. This is accomplished by considering the sum of passenger delays within a timeframe in a rail network system. By simulating future timetables with future scenarios with this framework, real-world dispatching strategies (which in practice are to a large extent standardized beforehand) can be adapted to minimize overall passenger delays.

Details about the settings of the test cases:

We are trying to set up two comparable test cases with a comparison of morning peak and non-peak scenarios. By checking the data that we got from SBB, most of the schedules starts from 8:00 to 14:00. We selected two target time slots 8:00-10:00 (morning peak) and 12:00-14:00 (non-peak). However, due the following two reasons, the two directions are probably not exactly symmetric for the Morning data:

(1) Some trains depart earlier than 7:30 and are not given in the provided OpenTrack data. In this case, the corresponding trains later than 10:00 are selected. This happens typically for S-Bahn; to other trains, this may also be the case.

(2) The trainIDs that are not matched in the results of OpenTrack –MATSim matcher are excluded.

These also result in the two test cases not being one-to-one correspondent. After selecting the trains for each test case, we run OpenTrack with a “0.5 hour earlier + 1 hour later” setting. This is to ensure the long-distance train having sufficient time for reaching/leaving the studied Winterthur region. Actually, the time scope of the two test cases are 7:30-11:00 and 11:30-15:00. The two test cases are named “TestCase1” and “TestCase2” respectively.

In the present report, we focus on a single possible use case, and, we mainly focus on checking the efficiency of this integrated simulation framework and examining the proposed TMS model. We consider the two test cases, as introduced above. Each test case includes 64 trains. For each test case, we set 4 delay scenarios:

(1) for TestCase1, the S train 20428 has an initial departure delay of 600, 700, 800, and 900 seconds from Effretikon (EF), called scenario 101, 102, 103, and 104 respectively;

(2) for TestCase2, the S train 20444 incurs an initial departure delay of 600, 700, 800, and 900 seconds from Effretikon (EF), also called scenario 101, 102, 103, and 104 respectively.

Namely, we set an initial departure delay of the periodic train course 204XX (S24) from station EF. This means, the delay scenarios in TestCase1 and TestCase2 are set for the same periodic train, but for different operating time windows of a day.

Moreover, in the TMS model, we set the weight ω of train delay to be 1, 0.1, and 0.01, to be balanced with passenger delay in the objective function, as explained in Section 4. Therefore, we have $2 \times 4 \times (1+3) = 32$ test scenarios in total.

In Table 3, we report the TMS results of TestCase1. The results of min train delay and min passenger delay are comparatively presented; their differences are highlighted in the lower portion of Table 3. For delays, we consider the following measurements:

(1) For train delay,

- total delay (tot): total delay over all trains at all stations (stopped or non-stopped);
- average delay (avg): average train delay per station;
- maximal delay (max): maximal delay per train per station;
- total final delay (tot final): total delay over all trains at the final station (destination);
- maximal final delay (max final): maximal train delay at final station (destination).

(2) For passenger delay,

- total delay (tot): total delay over all passengers at their destination;
- average delay (avg): average passenger delay at their destination;
- maximal delay (max): maximal delay per passenger at their destination.

Table 3 Results of TestCase1

Objective min	Weight ω	Scenario ID	CPU time (s)	Train delay (s)					Passenger delay (s)			# of infea routes
				tot	avg	max	tot final	max final	tot	avg	max	
$Z_1 = Z_{\text{train}}$	1	101	47.30	10485	17	604	515	515	7723	32	586	0
		102	60.31	12637	20	715	616	616	9142	38	687	0
		103	76.89	16545	27	838	804	725	13105	54	796	0
		104	58.45	17283	28	965	825	774	12896	53	930	0
$Z_2 = Z_{\text{pax}} + \omega \cdot Z_{\text{train}}$	1	101	49.81	10485	17	604	515	515	7723	32	586	0
		102	56.16	12637	20	715	616	616	9142	38	687	0
		103	76.84	16548	27	817	861	704	12443	52	775	0
		104	48.28	17283	28	965	825	774	12896	53	930	0
	0.1	101	55.48	11339	18	587	621	513	7604	31	552	0
		102	57.42	12637	20	715	616	616	9142	38	687	0
		103	75.42	16548	27	817	861	704	12443	52	775	0
		104	47.08	17283	28	965	825	774	12896	53	930	0
	0.01	101	44.27	11339	18	587	621	513	7604	31	552	0
		102	64.41	14595	23	691	818	642	9022	37	652	0
		103	153.00	20898	34	787	1290	596	12335	51	752	0
		104	50.95	17283	28	965	825	774	12896	53	930	0
Compare i.e., $Z_2 - Z_1$	1	101	49.81	0	0	0	0	0	0	0	0	0
		102	56.16	0	0	0	0	0	0	0	0	0
		103	76.84	+3	0	-21	+57	-21	-662	-2	-21	0
		104	48.28	0	0	0	0	0	0	0	0	0
	0.1	101	55.48	+854	+1	-17	+106	-2	-119	-1	-34	0
		102	57.42	0	0	0	0	0	0	0	0	0
		103	75.42	+3	0	-21	+57	-21	-662	-2	-21	0
		104	47.08	0	0	0	0	0	0	0	0	0
	0.01	101	44.27	+854	+1	-17	+106	-2	-119	-1	-34	0
		102	64.41	+1958	+3	-24	+202	+26	-120	-1	-35	0
		103	73.64	+4353	+7	-51	+486	-129	-770	-3	-44	0
		104	50.95	0	0	0	0	0	0	0	0	0

As shown in Table 3, the average computation time for TestCase1 is 64 seconds. For all the test scenarios, all the planned passenger routes are still feasible in the rescheduled timetable. By switching the target from trains to passengers, the total passenger delay decreases at the expense of a larger total train delay. We observe fluctuations in the increment of the total train delay: sometimes, very small, only a few seconds; sometimes, very larger, up to one hour. Let us look at scenario 103 as an example. When setting weight ω = 1, we observe that a 3-second increment of total train delay can already cause a 662-second reduction of total passenger delay. However, when setting weight ω = 0.01, the total train delay largely increases, by 3567 seconds, while the reduction of the total passenger delay is not significant, by only 770 seconds. A large increase in train delay does not (always) lead to a large reduction in passenger delay.

Moreover, we see the diversity of the results of different delay measurements. Even if the total train delay increases, the maximal train delay per station and the maximal final delay can still decrease. A reduction of the maximal train delay reveals that the delays are more evenly/equitably distributed over trains. For the passenger delays, the total, average, and maximal delays all become smaller, reflecting an improved overall performance from the perspective of passengers.

Figure 12 illustrates the TMS results of TestCase1 in a time-space graph, for the representative delay scenario 103. The trains of interest are as follows: (1) the S24 train 20428 in turquoise; (2) the IR train 2111 in yellow; (3) the IC train 707 in dark blue; and (4) the S7 train 18731 in pink. The passenger demand on these 4 trains (reported by MATSim, considering 10% of population, i.e., one agent represents 10 passengers) are given in Table 4.

Table 4 Passenger demand on the 4 trains of interest, in TestCase1

Train course ID	Amount of agent	Approx. Amount of passengers (1:10)	Origin	Destination
20428	13	130	EF	W
	1	10	SCAL	OWT
	1	10	RIK	FF
2111	35	350	EF	W
707	3	30	EF	W
18731	5	50	W	EF

As illustrated in Figure 12(a), where we minimize the total train delay, the initial departure delay of train 20428 at EF causes delays to both train 707 and train 2111 between EF and SCAL. Train 2111 overtakes train 20428 at OWT. The delay of train 2111 is recovered by using travel time supplement: after leaving FF, train 2111 almost follows the planned schedule.

Figure 12(b) shows the TMS results, where we minimize the total passenger delay and the total train delay with a weight of 1 or 0.1. We see that the sequence of train 2111 and train 707 is swapped between EF and SCAL, comparing with Figure 12(a). With this change, train 2111 incurs smaller delays while train 707 undertakes larger delays. Overall, the total train delay increases by 3 seconds, but the total passenger delay decreases by 662 seconds (since train 2111 has more passengers on board than train 707, see Table 4).

In Figure 12(c), we present the TMS results with a small weight ω = 0.01 for train delay. In this case, train 20428 and train 2111 maintain their planned train orders along their routes; as a result, we observe a large deviation of train 2111 from the planned schedule, while train 20428 undertakes smaller delays at the stations between OWT and FEL, benefiting the passengers on board the train.

By comparing the results shown in Figure 12, with balanced importance of train delay and passenger delay, we observe different control strategies produced by the TMS algorithm.

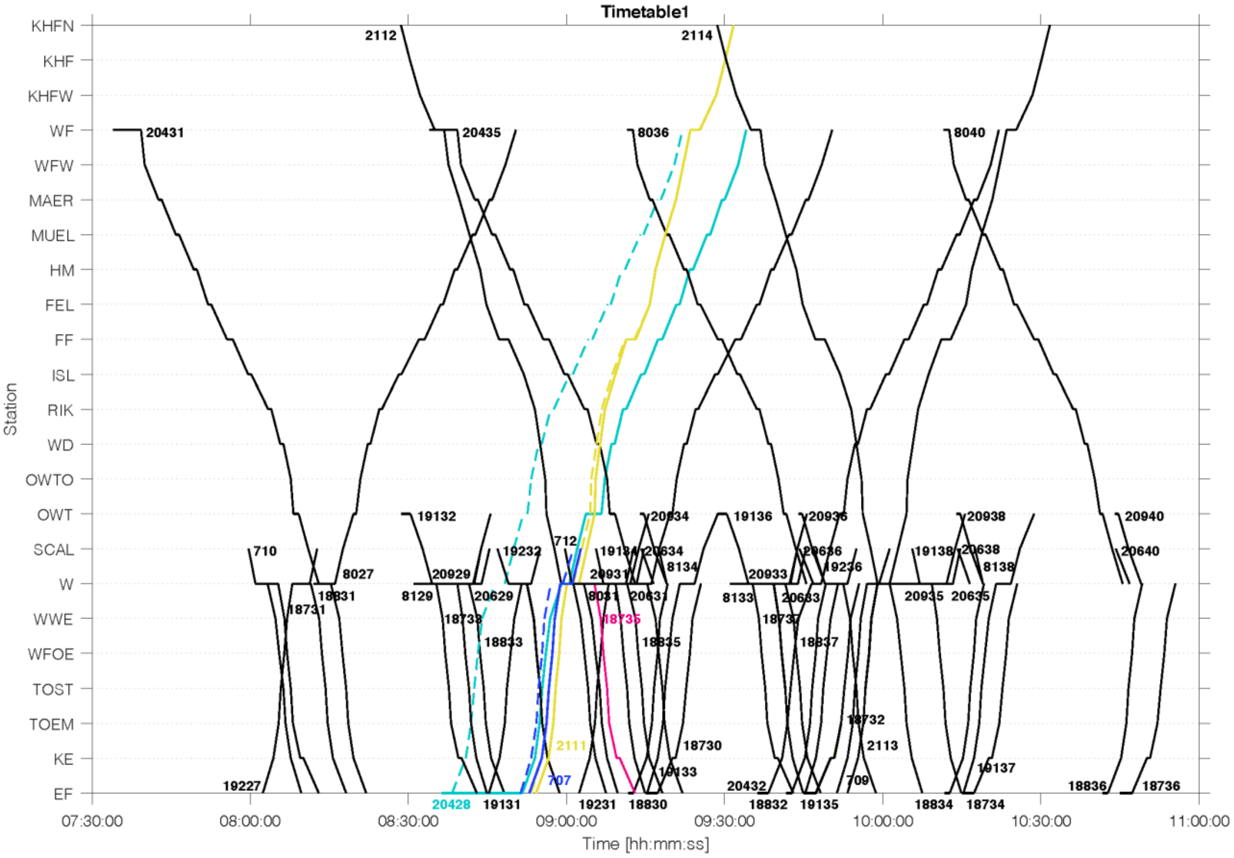


Figure 12(a) The TMS results of TestCase1, delay scenario 103, min train delay (Z_1)

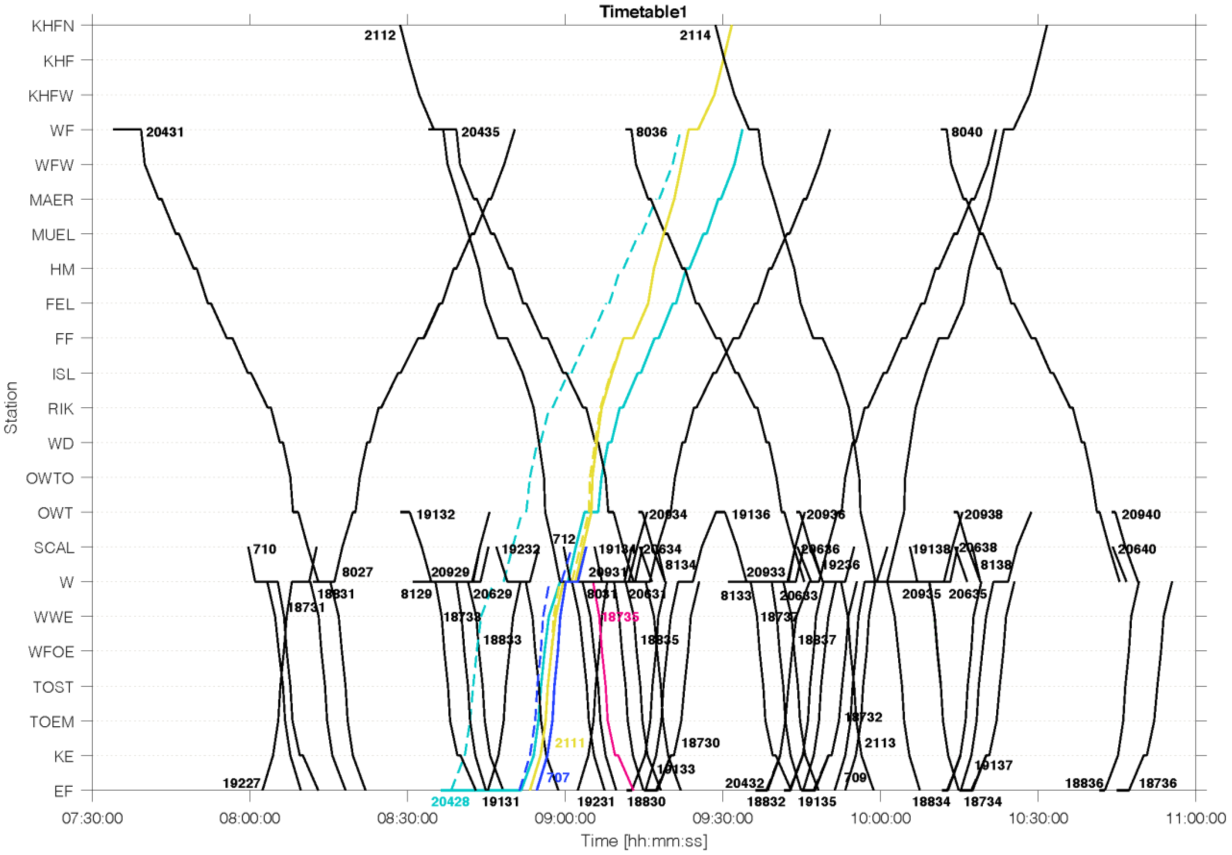


Figure 12(b) The TMS results of TestCase1, delay scenario 103, min passenger delay (Z_2), weight ω = 1 or 0.1

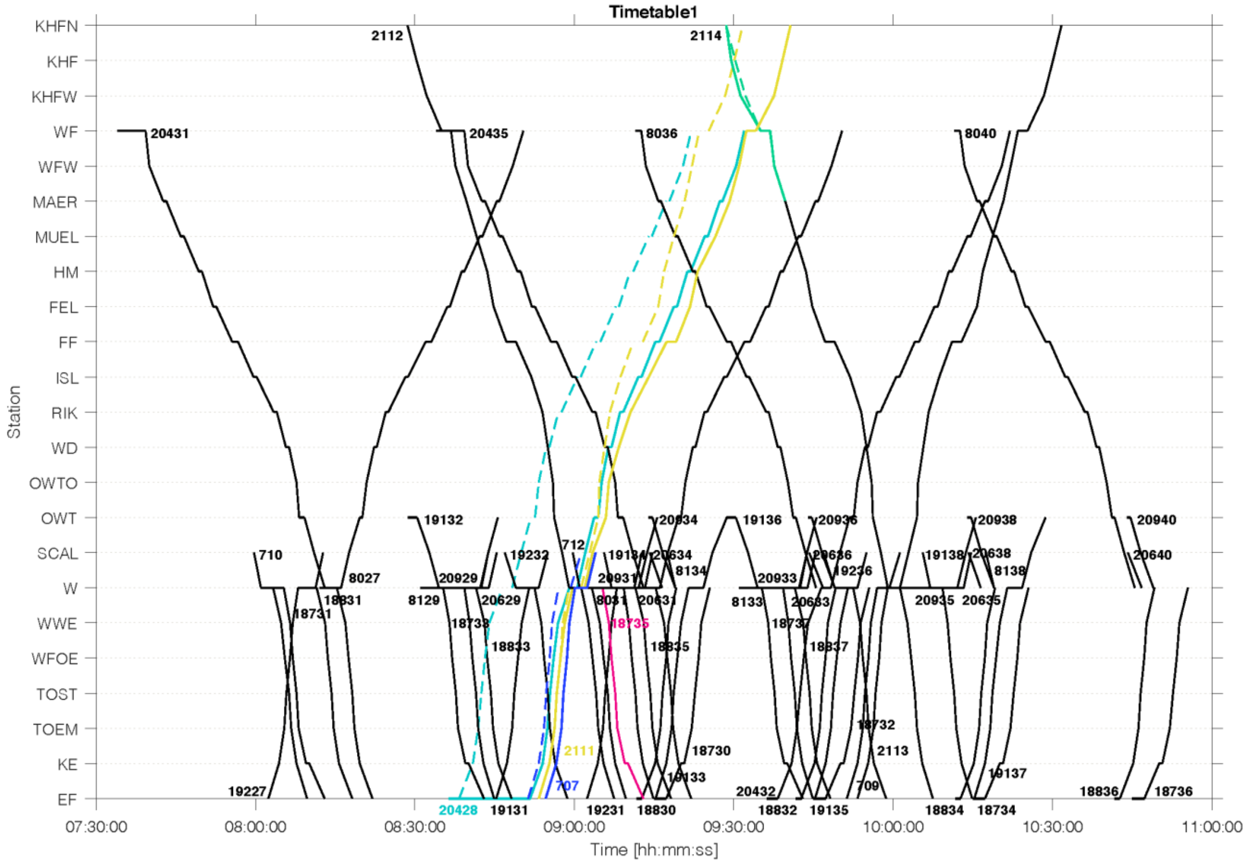


Figure 12(c) The TMS results of TestCase1, delay scenario 103, min passenger delay (Z_2), weight ω = 0.01

The results of TestCase2 are reported in Table 5. The insights remain the same to those of TestCase1; therefore, we do not elaborate the discussion of the results.

Table 5 Results of TestCase2

Objective min	Weight ω	Scenario ID	CPU time (s)	Train delay (s)					Passenger delay (s)			# of infea routes
				tot	avg	max	tot final	max final	tot	avg	max	
$Z_1 = Z_{\text{train}}$	1	101	74.74	11498	18	604	618	515	257	0	87	0
		102	63.67	12683	20	715	616	616	4	0	2	0
		103	75.00	16548	27	838	804	725	4954	10	79	0
		104	45.92	17286	28	965	825	774	6	0	3	0
$Z_2 = Z_{\text{pax}} + \omega \cdot Z_{\text{train}}$	1	101	67.41	11508	18	626	646	537	109	0	109	0
		102	56.14	12683	20	715	616	616	0	0	0	0
		103	64.98	17286	28	965	825	774	6	0	3	0
		104	49.48	17286	28	965	825	774	6	0	3	0
	0.1	101	53.30	12570	20	712	613	613	0	0	0	0
		102	53.25	12683	20	715	616	616	0	0	0	0
		103	60.89	17286	28	965	825	774	6	0	3	0
		104	44.53	17286	28	965	825	774	6	0	3	0
	0.01	101	45.78	12570	20	712	613	613	0	0	0	0
		102	46.77	12683	20	715	616	616	0	0	0	0
		103	51.86	17347	28	976	820	820	0	0	0	0
		104	46.88	17347	28	976	820	820	0	0	0	0

Compare i.e., $Z_2 - Z_1$	1	101	67.41	+10	0	+22	+28	+22	-148	0	+22	0
		102	56.14	0	0	0	0	0	-4	0	-2	0
		103	64.98	+738	+1	+127	+21	+49	-4948	-10	-76	0
		104	49.48	0	0	0	0	0	0	0	0	0
	0.1	101	53.30	+1072	+2	+108	-5	+98	-257	0	-87	0
		102	53.25	0	0	0	0	0	-4	0	-2	0
		103	60.89	+738	+1	+127	+21	+49	-4948	-10	-76	0
		104	44.53	0	0	0	0	0	0	0	0	0
	0.01	101	45.78	+1072	+2	+108	-5	+98	-257	0	-87	0
		102	46.77	0	0	0	0	0	-4	0	-2	0
		103	51.86	+799	+1	+138	+16	+95	-4954	-10	-79	0
		104	46.88	+61	0	+11	-5	+46	-6	0	-3	0

Figure 13 illustrates the TMS results of TestCase2 in a time-space graph, also for the representative delay scenario 103. The trains of interest (corresponding to TestCase1) are as follows: (1) the S24 train 20444 in turquoise; (2) the IR train 2119 in yellow (3) the IC train 715 in dark blue; and (4) the S7 train 18751 in pink. Table 6 reports the passenger demand on the 4 relevant trains, which is obtained from MATSim with a setting of 10% population.

Table 6 Passenger demand on the 4 trains of interest, in TestCase2

Train course ID	Amount of agent	Approx. Amount of passengers (1:10)	Origin	Destination
20444	0	0	-	-
2119	75	750	EF	W
	2	20	W	WF
	1	10	EF	KHFN
	1	10	W	KHFN
715	1	10	EF	WHE (invisible in Figure 13)
18751	0	0	-	-

Figure 13 presents the TMS results of minimizing the total passenger delay, while considering a weight ω = 1 or 0.1 for train delay.

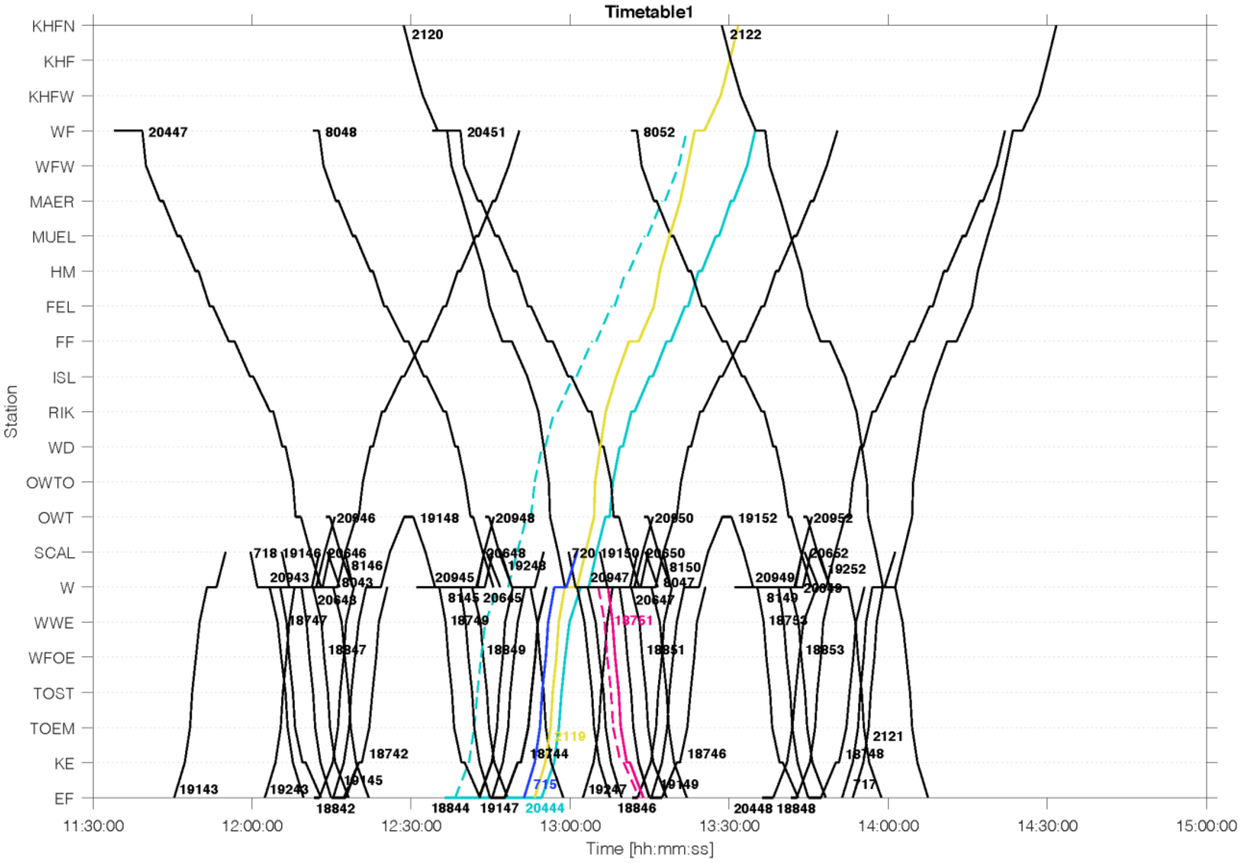


Figure 13 The TMS results of TestCase2, delay scenario 103, min passenger delay (Z_2), weight ω = 1 or 0.1

Here, we particularly compare the results of TestCase1 and TestCase2, for the representative delay scenario 103, showing the diversity in dispatching solutions due to different passenger demand. In delay scenario 103, we let the periodic train course 204XX (S24, in turquoise) incur an initial departure delay of 800 seconds at station EF. Figure 13 for TestCase2 corresponds to Figure 12(b) for TestCase1.

If looking at Figure 12(b) and Figure 13 together, we observe different dispatching strategies, caused by the different distributions of passengers on the trains. In Figure 12(b), the initial delay of train 20428 causes secondary delays of train 2111 and train 707. The orders of train 707 and train 2111 are swapped, and train 2111 overtakes train 20428 at OWT. However, in Figure 13, train 20444 undertakes a larger departure delay at EF (larger than the 800-second initial delay that we set). Consequently, train 715 and train 2119 run as planned without any delay, but train 18751 has a departure delay at station W, caused by the large departure delay of train 20444 and the need of the 18751 to cross the main track of the opposite direction when leaving W towards EF.

6. Conclusions

6.1. Summary

The present project investigated framework and tools, able to connect two well known, extensible simulators: a microscopic rail simulation (OpenTrack), an agent-based transport simulation (MATSim), and moreover using an optimizer for traffic management system (TMS). This way, joint analyses can be performed, taking advantage of the detail and characteristics of each simulator and optimizer. Agent-based simulation provides detailed passengers' behaviours and includes a multi-modal network. Microscopic rail simulation describes the rail network in a realistic way including detailed train speed in each section. The optimizer generates the optimized dispatching strategies with the objective minimizing passenger or train delay.

One main outcome of the project is the tooling that interconnects those systems, as a framework. Coupling the three parts can be used for a system-level evaluation of operational performance considering both supply and demand. Innovative measures of transport network performance can be quantified, such as the sum of passenger delays caused by corresponding train delays, reduced passenger delays thanks to dispatching strategies. At the same time, score values of train riders from MATSim are a proxy of the passengers' satisfaction concerning reliability of the rail services. By optimizing the rail network dispatching strategies to reduce delays and increase score values, rail networks have the potential to increase their attractiveness and consequently ridership.

The framework has been released open source online on GitHub, which matches the availability of the MATSim implementation.

6.2. Value as enabler

The value of the present project goes beyond the pure software tool developed and relates to a set of potential use cases. We expect the interconnection to industrially used tools, as well as the easier usage of available file formats, would enable a stronger dissemination of academic models for railway traffic management into industry.

Possible coupling which are enabled and eased by the developed framework include:

- Opentrack-checked travel times can be made available to Matsim, for a more accurate representation of travel times and transfer times
- Matsim generated demand can be made available to Opentrack simulations, to evaluate not only train delay but also passenger delay, or passenger reliability
- Passenger delays can quantify a reliability penalty to be included into mode/route choice for Matsim studies
- Passenger load value onboard trains can be used to simulate/ adjust dwell time, especially in case of (over-)crowded vehicles or trains
- Moreover, data which uses the same interface/ structure, but is not directly coming from Matsim or Opentrack can be converted towards other tools. For instance, it is possible to export some data via the interface to external tools using Matsim or Opentrack interface like OpenPowernet, OpenTimetable, Visum

Specific aspects pertain also the usage of an optimizer, or a decision support for specific parts, like:

- Study of interaction effects, especially in transfer management or delay management, when the original or an updated disposition timetable is used. This could also include dissemination of updated information about the disposition timetable, and possible compliance of the passengers.
- Robustness study especially for poor operating situations in limited systems (for example S-Bahn Tessin, TILO) taking into account accumulating train delays by an increasing number of waiting passengers
- Impact of vehicle circulations from the original and updated disposition timetable and determining the impact of rolling stock rostering towards delays of trains and of passengers.

6.3. Value as exploration platform for passenger-oriented control of railways

The current report proposes just an illustrative example where the framework can showcase its possibilities. This is based on a specific test case, which shows a possible application by means of optimization for a microscopically realistic train traffic description, as well as the inclusion of a complex demand management. We reflect in this section about the possibility that such a more comprehensive integration will produce.

Having established a common framework, including data formats and structure, multiple optimizers could be coupled to any of the concerned simulators, to allow interoperability of different tools. This allows to define benchmarks in a much easier way, to be used for evaluation in agent-based simulation and transport models, as well as railway simulation and optimization.

The specific use case referred to a passenger-oriented optimizer for disposition timetables, which is able to include the amount of passenger delay as a weighted sum together with train delay. This highlighted the practical complexity of getting realistic data, which is now solved by means of the available framework; the computational complexity of solving the mathematical program into an optimal solution within a small computing time; and the system-wide implications of using those insights to sketch a roadmap for possible utilization of similar approaches in real life. Summarizing,

- passenger delays and train delays can differ; and having different loads is reflected in different optimization solutions

A decision support for dispatching or determining disposition timetables should include a variety of key performance indicators, which go beyond the average train delay currently considered by the simpler mathematical models. The passenger delay is one such increase of complexity. Though, this is not enough to describe completely which decision to take, as other factors can also play a role:

More parameters are required to properly support decision making, among which:

- the available capacity of the network should be considered. In heavily loaded networks with mixed passenger and freight traffic and a high chance of delay propagation, more focus could be given to having the traffic run smooth and to allow more trains to pass in a given time window by minimizing speed differences on specific sections, and less to passenger delay as an absolute number,

- Demand cannot be characterized only by its volume, but also by distinguishing between peak and non-peak period; night and day; week/ weekend. In general, a higher service frequency would allow less explicit consideration of passenger delay. The optimization model could be able to include all those aspects in the description of the problem, if the boundary constraints are properly defined (i.e. what happens with stranded passengers; are all services modeled, or only part of the network is modeled, and so on).
- distinction between average delay and maximum delay; and maximum delay for different OD pairs, in order to balance average perspective with inclusion of specific requirements of all groups; fairness and equity
- Availability of rolling stock for return services, short turns, or if trainsets are to be strengthened/combined or reduced/split, at certain points of the network.

This leads to a need for multiple models, able to associate realistic, correct value to all performance measures, before being able to try in practice the models. Otherwise, there is a risk of losing support from stakeholders when unexplainable decisions (possibly based on wrong data, or incorrect assumptions) have to be considered.

For this it is worthwhile identifying a study case, possibly a bottleneck, where the difference between current non-optimized solutions and optimized solution is relevant; and where the difference between solution optimizing for passenger delay or train delay is relevant. With regard to the Swiss railway network, such spots could be the sections Chiasso-Ceneri-Locarno/Biasca or Bex-Lausanne-Genève, which both have dense regional traffic as well as long-distance traffic and, in addition, heavy freight traffic, where the question passenger delay versus route capacity is very important. On such a test case, one could re-evaluate the past decisions taken; could identify simpler way of presenting the optimized solutions, for instance by approximated decision rules, thresholds. Overall, a sub-optimal solution which can be easily understood and justified can deliver more value than an optimal solution which is not trusted by the decision maker.

On such a test case, the impact towards passengers, for specific passenger groups can be studied in detail, for instance determining the gap between expected demand (from the MATSim model) and real demand; and the usability of information dissemination (giving the most-correct information at the right moment, to the right person) when there are multiple travel options.

For instance, for online transfer management, the passenger behaviour must be included correctly to avoid surprising or counteracting effects, especially passenger volume must be as correct as possible. Concerning passenger flows, this should also include effects of capacity in the vehicles, when abnormal loads are experienced or expected by the models.

In general, the more effects are considered, the more fuzzy the outcome can be, and the more relevant the impact from the assumptions used. In a closed loop decision support, the variability of the suggested solutions over time has also to be established and for what possible, reduced.

For such a test case, it would be of great interest to understand sensitivity of the results to the possible sources of error, including imprecise status estimation, passenger demand, passenger destination. A robust solution is a good asset, in case some of the required parameters can be correctly estimated only against a high cost, or not at all.

Such a study would complement the test case already proposed in the literature, for instance in the ONTIME Project (Quaglietta et al, 2020).

References

Borndörfer, R., Erol, B., Graffagnino, T., Schlechte, T., Swarat, E., 2014. “Optimizing the Simplon railway corridor”, *Ann Oper Res*, vol. 218 (1), pp. 93–106.

Cadarso, L., Maroti, G., Marin, A., 2015. “Smooth and controlled recovery planning of disruptions in rapid transit networks”, *IEEE Transactions on Intelligent Transportation Systems*, vol. 16 (4), pp. 2192–2202.

Corman, F., D’Ariano, A., Marra, A. D., Pacciarelli, D., Samà, M., 2017. “Integrating train scheduling and delay management in real-time railway traffic control”, *Transportation Research Part E: Logistics and Transportation Review*, vol. 105, pp. 213-239.

Corman F, Quaglietta E., Goverde, R.M.P. (2018). Automated real-time railway traffic control: An experimental analysis of reliability, resilience and robustness. *Transportation Planning and Technology* 41(4), pp. 421-447 doi:10.1080/03081060.2018.1453916

Franke, B., Lieberherr, J., Kaeslin, L., 2018. “Kombinierte betriebliche und verkehrliche Bewertung von Bahn-Angeboten”, *Eisenbahntechnische Rundschau (ETR)*, Nr. 7+8, S. 26-29. (in german)

Horni, A, Nagel, K., Axhausen, K. W. (eds.), 2016. *The Multi-Agent Transport Simulation MATSim*, Ubiquity Press, London.

Högdahl, J., Bohlin, M., Fröidh, O., 2019. “A combined simulation-optimization approach for minimizing travel time and delays in railway timetables”, *Transportation Research Part B: Methodological*, vol. 126 (3), pp.192–212.

Leng, N., Liao, Z., Corman, F., 2020. “Role of Timetable, Rolling Stock Rescheduling, and Information Strategies to Passengers in Public Transport Disruptions”, *Transportation Research Record*, vol. 2674 (9), pp. 135-147.

Luan, X., Corman, F., Meng, L. (2017). Non-discriminatory train dispatching in a rail transport market with multiple competing and collaborative train operating companies. *Transportation Research Part C: Emerging Technologies*, 80, 148-174.

Luan, X., De Schutter, B., Meng, L., Corman, F. (2020). Decomposition and distributed optimization of real-time traffic management for large-scale railway networks. *Transportation Research Part B: Methodological*, 141, 72-97.

Medeossi, G., Longo, G., de Fabris, S., 2011. “A method for using stochastic blocking times to improve timetable planning”. *Journal of Rail Transport Planning & Management*, vol. 1 (1), pp.1–13.

Nash, A., Huerlimann D., 2004. “Railroad simulation using OpenTrack”, *WIT Transactions on The Built Environment*, vol. 74, pp. 45-54.

Quaglietta, E., Pellegrini, P., Goverde, R.M., Albrecht, T., Jaekel, B., Marlière, G., Rodriguez, J., Dollevoet, T., Ambrogio, B., Carcasole, D. and Giaroli, M., 2016. The ON-TIME real-time railway traffic management framework: A proof-of-concept using a scalable standardised data communication architecture. *Transportation Research Part C: Emerging Technologies*, *63*, pp.23-50

Rieser, M., Métrailler, D., Lieberherr, J., 2018. “Adding realism and efficiency to public transportation in MATSim”, In: *Proceedings of The 18th Swiss Transport Research Conference (STRC 2018)*, Ascona, Switzerland.

Schöbel, A., 2007. “Integer programming approaches for solving the delay management problem”, *Algorithmic Methods for Railway Optimisation, Lecture Notes in Computer Science*, vol. 4359, pp. 145–170.

Sels, P., Dewilde, T., Cattrysse, D., Vansteenwegen, P., 2016. “Reducing the passenger travel time in practice by the automated construction of a robust railway timetable”, *Transportation Research Part B: Methodological*, vol. 84 (1), pp.124–156.

Stephan, A., 2008. “OpenPowerNet –simulation of railway power supply systems”, *WIT Transactions on The Built Environment*, vol. 103, pp. 449-459.