Konrad-Zuse-Zentrum
für Informationstechnik Berlin

Takustraße 7
D-14195 Berlin-Dahlem
Germany

RALF BORNDÖRFER BERKAN EROL
THOMAS GRAFFAGNINO THOMAS SCHLECHTE
ELMAR SWARAT

Aggregation Methods for Railway Networks

This work was funded by the Federal Ministry of Economics and Technology (BMWi), project *Trassenbörse*, grant 19M7015B.

ZIB-Report 10-23 (November 2010)

Aggregation Methods for Railway Networks*

Ralf Borndörfer** Berkan Erol** Thomas Graffagnino‡
Thomas Schlechte** Elmar Swarat**

January 26, 2011

Abstract

This paper presents a bottom-up approach of automatic simplification of a railway network. Starting from a detailed microscopic level as it is used in railway simulation, the network is transformed by an algorithm to an aggregated level, i.e., to a macroscopic network, that is sufficient for long-term planning and optimization. Running and headway times are rounded to a user defined discretization by a special cumulative method. After the transformation we saturate the network with given train requests by computing an optimal slot allocation. Then the optimized schedule is re-transformed to the microscopic level in such a way that it can be simulated without any conflicts between the slots. We apply this algorithm to "macrotize" a microscopic network model of the dense Simplon corridor between Switzerland and Italy. With our micro-macro transformation method it is possible for the first time to generate a profit maximal and conflict free timetable for the entire corridor and for an entire day by a simultaneous train slot optimization.

1 Introduction

Timetabling is one of the major planning tasks in railway traffic. It involves two parts. On the one hand the *railway operators* need to compute a timetable using a small number of vehicles and crews that satisfies passenger demands like short transfer and travel times. On the other hand the *infrastructure companies* must decide about the allocation of train slots to the train requests of the operators. This is especially challenging when conflicts between different requests occur. In such a situation, in particular, in highly utilized networks, manual planning can become very complex and personnel-intensive. Then infrastructure capacity might be left unused or good connections might not be guaranteed for all important points in the network. There is therefore a need for modeling methods that allow for the use of optimization algorithms in timetabling like PESP [35] or TTP [9].

Railway efficiency and the capacity of railway networks are important research topics in engineering, operations research, and mathematics for several

*This work was funded by the Federal Ministry of Economics and Technology (BMWi), project *Trassenbörse*, grant 19M7015B.

**Zuse Institute Berlin (ZIB), Takustr. 7, 14195 Berlin-Dahlem, Germany, Email {borndoerfer, erol, schlechte, swarat}@zib.de

‡SBB AG Bern, Mittelstrasse 43, 300 Bern 65, Switzerland, Infrastructure / Timetable and Network Design, Email thomas.graffagnino@sbb.ch

decades. The main challenge is to master the tradeoff between accuracy and complexity in the planning, optimization, and simulation models. Radtke [31] and Siefer et al. [15] proposed the use of both *microscopic* and *macroscopic models*. They applied microscopic models for running time calculations and the accurate simulation of railway operations, and macroscopic models for long term traffic and strategic infrastructure planning. In a similar vein, Schultz [32] suggested a procedure to insert train slots according to pre-defined priorities in a first step, and to test the reliability of this timetable in a second step by simulating stochastic disturbances. An alternative approach to determine the capacity of a network are *analytical methods*. They aim at expressing the railway efficiency by appropriate statistics, e.g., the occupancy rate. There exist two different approaches: The first is the *handicap theory* by Potthoff [30]; it is based on queueing models. The second uses *probabilistic models* to compute follow-on delays; it is mainly based on the work of Schwanhäußer [33]. He also introduced the important concept of *section route nodes* to analyze the performance of route nodes or stations. Hansen [16] presents a probabilistic model as an alternative to queueing models for a precise estimation of expected buffer and running times. Finally, there is also a substantial literature on discrete optimization approaches to timetable optimization. Due to the complexity of railway traffic, most articles consider only simplified macroscopic models with a simplified routing through the railway infrastructure on simple network topologies, such as corridors, e.g., [7, 5, 9, 23, 4, 14]. On the other hand, routing through individual stations has been considered on a much more detailed level, see [39, 24, 10]. The interaction of both approaches has only recently been studied [8], using a top-down approach.

In this paper a *bottom-up* approach of *automatic simplification* of a complex *microscopic* railway infrastructure model is presented and applied in a case study for the Simplon corridor. The term "microscopic" points out that the input data describes the infrastructure on a very detailed level, that makes it possible to simulate the railway traffic with exact track, switch, and platform assignments of the train paths like it would be in the real world. An aggregation technique condenses this microscopic representation to those data that are relevant for planning and optimization purposes. Transforming the data to a less detailed level makes it possible to compute timetables and optimal slot allocations by methods of linear and integer programming. Of course, the aggregation has to be done in such a way that enough degrees of freedom remain, and in such a way that a slot allocation on the macroscopic level can be transformed back to the microscopic level without creating any conflicts. We describe in this paper a method that does exactly this.

We test our method using real world data for the Simplon corridor from Brig (BR) in Switzerland to Domodossola (DO) in Italy provided by the SBB Schweizerische Bundesbahnen. The Simplon is known as one of the major corridors in the European railway network. It has a length of 45 km and features 12 stations. The microscopic model for this scenario consists of 1154 nodes and 1831 arcs including 223 signals, which is fairly large, see Figure 1. Furthermore the routing possibilities at the terminals Brig and Domodossola and in the intermediate stations Iselle and Varzo, and a rather unusual slalom routing for certain cargo trains through the tunnel lead to complex planning situations.

Before describing our micro-macro transformation in detail, we give a short discussion on the pros and cons of microscopic and macroscopic railway model-

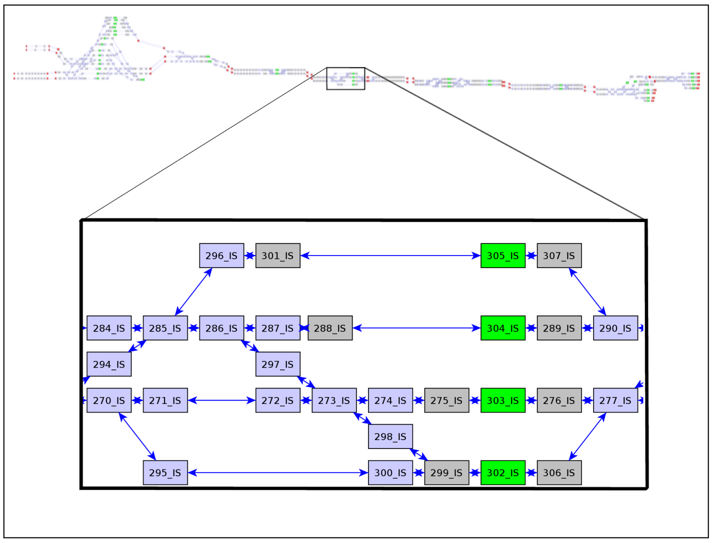

Figure 1: Microscopic network representation of the Simplon corridor and detailed representation of station Iselle as exported by the railway simulator `OpenTrack`.

ing, and why they have to be combined in order to arrive at a method that is both accurate and tractable.

Railway infrastructure and train operations are often modeled using *simulation* programs. In the last 20 years several software programs for simulating train movements were developed [37], [20], [36]. Almost all railway companies use them to support their operations and planning processes. Simulation systems provide a realistic assessment of different options in infrastructure planning. They allow to study the interactions of large numbers of trains in a network, and, in particular, to evaluate the feasibility of a timetable, i.e., if a timetable works in simulation, it can be trusted to be operable in practice. We used in our work the synchronous simulation system `OpenTrack`, that was developed at the ETH Zürich [20], see also [17] for an overview and a comparison of synchronous and asynchronous simulation systems.

A simultaneous *optimization* of a large number of train slots at a microscopic granularity is currently out of reach and would also not be appropriate in many high-level strategic and tactical planning situations. For these purposes, it is better to resort to a *macroscopic model* of the railway system. Such a macroscopic model contains much less information such that the network size can be reduced significantly. In addition to that, a fixed time discretization can be used in order to make the model amenable to discrete optimization techniques. In [13] a standardized format for macroscopic railway models was introduced and a number of test instances that model a part of the German long distance network were made freely available. For line optimization [3] and for periodic timetable optimization [23], simplified macroscopic models of the railway infrastructure and estimates of event times, mostly in minutes, have been used with success.

Our contribution is to present a bottom-up approach to *railway network aggregation* that starts at the microscopic level, goes to a macroscopic model, and ends again at the microscopic level. We present in Section 2 an algorithmic approach thats implements this idea. This approach is tested in Section 3, where we present computational results for different optimization scenarios for the Simplon corridor.

2 Microscopy and Macroscopy, or There and Back Again

Railways are highly complex technical systems, which can be modeled at any level of detail. This modeling effort is no end in itself. Rather, an accurate calculation of running times and precise and unique platform and track allocations are needed to make simulation results match with the real world. The necessary precision can be achieved using microscopic data such as gradients, speed-dependent tractive efforts, speed limitations, and signal positions. However, this type of information is too complex to be handled in a discrete optimization model. Our aim is therefore to work with a *macroscopic model* with the property that the results can be interpreted in and re-transformed to the microscopic world and finally operated in reality. The main contribution of this work is to introduce an algorithm that constructs from a microscopic railway model a macroscopic model with the following properties:

- ▷ macroscopic running times can be realized in microscopic simulation,
- ▷ sticking to macroscopic headway-times leads to conflict-free microscopic block occupations,
- ▷ valid macroscopic timetables can be transformed into valid microscopic timetables.

This section defines the microscopic and macroscopic elements of our approach, and it describes a suitable transformation in detail. It is structured as follows. Subsection 2.1 discusses microscopic railway network models. Subsection 2.2 motivates our aggregation idea and introduces some details concerning the construction of macroscopic networks. The following Subsection 2.3 deals with time discretization. Finally, we propose an algorithm that performs the micro-macro transformation in Subsection 2.4. We remark that although our exposition is based on the simulation tool `OpenTrack`, the methodology is generic.

2.1 Microscopic Railway Networks

The main input for the transformation algorithm is a *microscopic infrastructure network* that is given as a graph $G = (V, E)$. `OpenTrack` uses a special graph data structure in which nodes correspond to so-called *double-vertices*. These consist of a left and a right part, see Figure 2 for examples and [25, 26] for a more detailed description. `OpenTrack` adopts the convention that if a path in G enters a node at the left end, it has to leave at the right end and vice versa. This assures that the direction of the train route is always respected and no illegal turn arounds at switches can be done. Every *track section* between two vertices is modeled as an edge, and every edge has some attributes like maximum speed or length. A double-vertex is introduced at any point where one or more of these attributes change or if there is a switch, a station, or a signal on a track. Figure 2 shows an example of a double-vertex graph in `OpenTrack`.

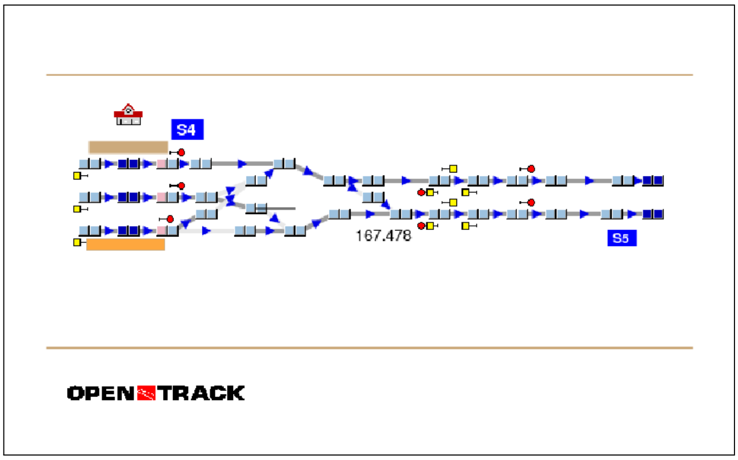

Figure 2: The topology of a part of a microscopic railway network plotted by the simulation software `OpenTrack`. Signals can be seen at some nodes, as well as platforms and station labels.

Our transformation approach is based on the consideration of a set of potential *routes* in $G = (V, E)$ for trains of standardized *train types*. A *microscopic route* is a path through the microscopic infrastructure that is valid for some train type and that starts and ends at a node inside a station or at a node representing a storage siding. Some nodes on the route can be labeled as stops, namely, when the train can potentially stop there, i.e., at nodes representing station platforms, or at stop opportunities on passing tracks. Note that the train routes induce the directions in which the microscopic infrastructure nodes and edges can be used. This will directly influence the definition, e.g., of the headway parameters of the macroscopic model, as we will explain later in Section 2.2. Let C denote the set of all train types and R the set of all given routes in $G = (V, E)$ (note that several routes can belong to one train type).

Train types should be chosen clearly arranged and conservatively with respect to their "train class" (heavy cargo trains, slow interregional or regional passenger trains) to avoid infeasible running times. Detailed simulation data has to be calculated carefully such that precise running times and blocking times in units of δ (some discretization step size, e.g., one second) can be computed, see Figure 3. *Running times* and *blocking times* are basic elements of our approach and will be discussed next.

In [28, 6] the basic laws of dynamics are applied to derive the *dynamics of a train movement*. These methods have been implemented in state-of-the-art railway simulation software packages, e.g., `OpenTrack`, in order to come up with plausible values for exact running times, see [27]. Different tools differ in their data structures, interfaces, and in some minor interpretations. However, the main concepts of running and blocking times are the same. We remark that our approach can not only be used in connection with `OpenTrack`, but that it can be easily adapted to any simulation tool that provides accurate running and blocking times, such as `RailSys` or `RUT-K`.

In Europe, *blocking times* are used to quantify the infrastructure capacity

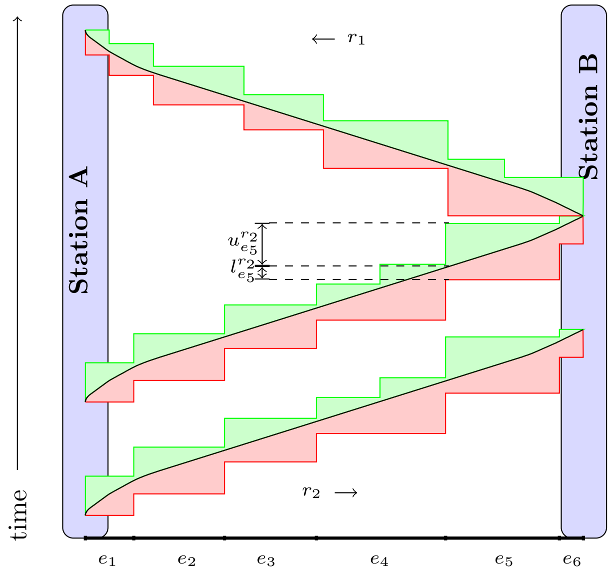

Figure 3: Blocking time diagrams for three trains on two routes using six blocks. Below are two subsequent trains on route r_2 and at the top a train on the opposite route r_1.

consumption of train movements. The approach is based on the early work of Happel [18, 19] and the intuitive concept to associate the use of physical infrastructure resources over certain time intervals with trains or train movements, see also [29, 22] for a comprehensive description of blocking time theory. We will now give a brief discussion of blocking times that contributes to a better understanding of our transformation algorithm.

The origin of the *blocking time stairs*, shown in Figure 3, is the well-known train protection system called *train separation in a fixed block distance*. In this method, the railway network is divided into *block sections*, which are bordered by *main signals*. A block section must not be occupied by more than one train at a time. When a signal allows a train to enter a block section, the section is locked for all other trains. In this way, the entire route between the block starting main signal and the overlap after the subsequent main signal is reserved for the entering train.

Figure 3 shows that the time interval during which a route r occupies a track segment consists of the relative reservation duration l^r_e and the relative release duration u^r_e on edge $e \in E$. The relative reservation duration is the sum of the approach time, the signal watching time, sometimes called reacting time, and time needed to set up the route. The relative release duration is the sum of the release time, the clearing time, sometimes called switching time, and time needed by the train between the block signal at the beginning of the route and the overlap. The switching time depends significantly on the installed technology, see [34, 22]. In order to prevent trains that want to pass a block section from undesirable stops or brakings, the block reservation should be finished before the engine driver can see the corresponding distant signal. Then the section stays locked while the train passes the track between the beginning of the visual distance to the caution signal and the main signal and thereafter the block section until it has cleared the overlap after the next main signal. Then the section is released. This regime can be improved in block sections that contain con- or diverging tracks, because in such cases it is often possible to release parts of the section before the train has passed the overlap after the next main signal. We finally remark that blocking times are also used in moving block systems like the future ETCS Level 3 system. Arbitrarily small blocks, i.e., blocks with lengths converging to zero, are considered in simulations of moving block systems in order to emulate the blocking times, see also [11] and [38] for an investigation of the influence of ETCS Level 3 on the headway times. Simulation tools have to respect all these technical details. From an optimization point of view, however, it is sufficient to consider abstract blocking time stairs, regardless of the safety system they stem from or how they were computed.

We summarize the microscopic information that we use:

- an (undirected) infrastructure graph $G = (V, E)$,
- a set of train types C,
- a set of directed train routes R, $r = (e_1, e_2, \ldots, e_{n_r})$ with $e_i \in E$,
- each route $r \in R$ belongs to one train type,
- a time discretization granularity δ,
- positive running times $\tilde{d}_e^r$ on edges $e \in E$ for all routes $r \in R$ measured in δ,
- positive release durations u_e^r on edges $e \in E$ for all routes $r \in R$ measured in δ,
- positive reservation durations l_e^r on edges $e \in E$ for all routes $r \in R$ measured in δ,
- the orientation of edges is induced by routes (one or both directions),
- stop opportunities for some nodes $v_i \in V$, which are induced by traversing routes.

2.2 Network Aggregation

The desired *macroscopic network* is a directed graph $N = (S, J)$ for train types C, that is derived from a microscopic network $G = (V, E)$ and a set of routes R. The construction involves aggregating (inseparable) *block sections* (paths in the microscopic network G) to macroscopic *tracks* J and *station areas* (subgraphs of the microscopic network G) to macroscopic *stations* S. The aggregation will be done in a way that depends on the given routes R and on the defined train types C, such that the complexity of the macroscopic network depends only on the complexity of the interactions between the *given* train routes, and not on the complexity of the network topology, which covers all interactions between *all* potential train routes, which is much more. This is a major advantage over other approaches, because the aggregation is detailed where precision is needed and compressed where it is possible.

We will now describe the idea of the construction by means of an example. First, all potential departure and arrival nodes at some station that are used by the routes R are mapped to one macroscopic station node. Additional macroscopic nodes will be introduced in order to model interactions between routes due to shared resources. The potential interactions between train routes in a double-vertex graph are:

- complete coincidence, i.e., routes have identical microscopic paths,

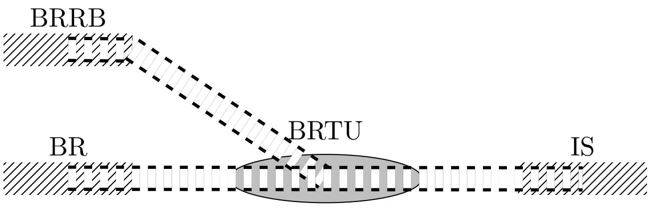

Figure 4: A railway network.

▷ convergence, i.e., routes merge at a microscopic node (and traverse it in the same direction),
▷ divergence, i.e., routes separate at a microscopic node (and traverse until then in the same direction),
▷ crossing, i.e., routes cross at a microscopic node (and traverse it in the opposite direction).

Let us discuss some of these interactions between train routes at the example of the infrastructure network shown in Figure 4.

Consider first a single standard train that runs from platform BRRB (we denote any place where stopping is allowed as a platform) to platform IS. Then it is enough to consider just one single track from station BRRB to IS in the macroscopic infrastructure. Note that this macroscopic track could correspond to a long path in the microscopic representation. Consider now additional standard trains from BRRB to IS. Possible interactions and conflicts between these train routes are the self correlation on the *directed track* from BRRB to IS, as well as the platform capacity for standard trains, which allows, say, exactly one train to wait in BRRB or IS. Another standard train running from BR to IS calls for the definition of a *pseudo-station* BRTU at the track junction in order to model the train route convergences correctly. (Our model distinguishes between regular station nodes, where a train can stop, and pseudo-station nodes, which are not stop opportunities, i.e., in our model trains are not allowed to wait at a pseudo-station or to change their direction there.) The pseudo-station BRTU splits the track from BRRB and IS into two tracks: from BRRB to BRTU and from BRTU to IS. The second of these tracks is used to model the resource conflict between converging routes of trains from BRRB to IS and trains from BR to IS, which is locally restricted to the track from BRTU to IS (or more precisely from the first blocks to reserve containing the switch of BRTU). If it is possible to run trains on the same microscopic segment in the opposite direction from IS to BRRB, another directed track has to be defined in the macroscopic network. Besides the standard self correlation, the conflict for opposing routes also has to be modeled, see Figure 3. Diverging or crossing situations between opposing train routes can be handled in an analogous way. Along the lines of these examples, we can exploit aggregation potentials in the infrastructure by representing several microscopic edges on a route by only one macroscopic track. Of course, macroscopic track attributes can also be compressed. For example, if we assume that the route from BRRB to IS and the route from BR to IS are operated by the same train type, we can use a single value for the running time on the track from BRTU to IS.

After constructing the regular stations, the pseudo-stations, and the tracks

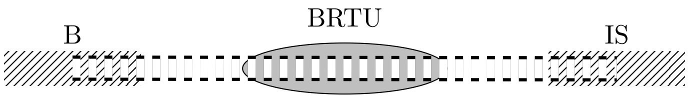

Figure 5: Aggregated macroscopic infrastructure network.

between them, the network can be further reduced by a second aggregation step. Again consider the situation in Figure 4. Suppose platforms BRRB and BR belong to the same station B. If BRTU is a close junction associated with B, then it may be viable to contract nodes BRRB and BR to one major station node B with a directed platform capacity of two as shown in figure 5. Of course, by doing so we loose the accuracy of potentially different running times between different platforms of B and the other stations, and we also loose control over the routing through or inside B, which both can produce small infeasibilities on the operational level. However, one can often achieve significant reductions in network sizes in this way, without loosing too much accuracy.

2.3 Time Discretization

Discrete optimization models for timetabling and slot allocation are based on the use of *space-time graphs*, i.e., the time is discretized. Similar as for the topological aggregation, there is also a tradeoff between model size and accuracy in the temporal dimension. This tradeoff is controlled by the *discretization step size*. The discretized times in the macroscopic model will be based on microscopic simulation data, which is very precise. In fact, simulation tools provide running and blocking times with an accuracy of seconds (or even smaller). Our aim is to aggregate these values in the macroscopic model. We propose for this purpose a conservative approach, which means that running and arrival times will never be underestimated in the macroscopic model.

2.3.1 Running Times

Let $\Delta \in \mathbb{N}$ be a fixed time discretization, i.e., a unit of time, in which all macroscopic times will be measured; e.g., using units of six seconds is denoted as $\Delta = 6$. Then a first idea is to simply round up all running times to the next unit of Δ; let us call this procedure *ceiling rounding*. Figure 6 shows the difference between microscopic and ceiling rounded running times for a microscopic running time of $\tilde{d}_j^r = 74$ at some track j in some route r with respect to different time discretizations Δ. Fine discretizations like less than 15 seconds produce small deviations, while larger time discretizations can increase the error significantly. The main problem with ceiling rounding is that the error accumulates along a route. In fact, the worst case rounding error for each track equals $\Delta - 1$, such that a route of n tracks can be off by an error of $n\Delta - n$. Such big rounding errors lead to undesirable extensions of travel times and an inefficient use of the infrastructure capacity.

We therefore propose an alternative approach in terms of a more sophisticated *cumulative rounding* technique. This procedure aims to control the rounding error by only tolerating small deviations between rounded and microscopic running times. The idea is simple: considering running times for each route

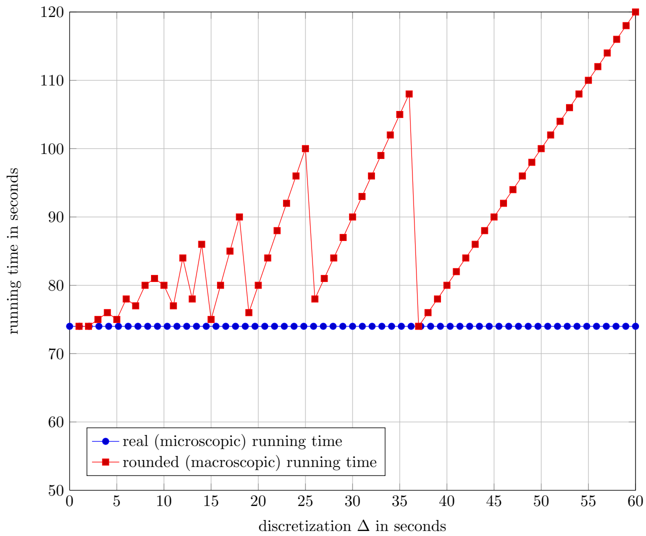

Figure 6: Microscopic and macroscopic running times on a railway track for time discretizations varying between 1 and 60 seconds.

on each track with respect to the cumulative rounding error, it is sometimes allowed to round down, because enough buffer time was collected on the way. We must, however, make sure that running times are never rounded to zero, because in our model zero running times are not counted as infrastructure usage, and this can lead to infeasible timetables. A formal description of the procedure applied to a single track of the macroscopic infrastructure network is given in Algorithm 1. There, we denote by $\tilde{d}_j^r$, d_j^r, and ϵ_j^r the microscopic running time of route r at track j, the discretized running time, and the cumulative rounding error, respectively. At the beginning of a route the cumulative rounding error ϵ_{-1}^r equals zero.

An analysis of the cumulative rounding algorithm shows that the rounding error never exceeds Δ, given that the microscopic running times are never smaller than Δ. For each track j on route r where this condition is not fulfilled, i.e., $\tilde{d}_j^r < \Delta$, the error can grow by Δ. For example, if we have one track j with $\tilde{d}_j^r < \Delta$, the upper bound for the rounding error along route r equals 2Δ.

Figure 7 compares the two rounding methods by illustrating the minimum, average, and maximum rounding errors of the macroscopic running times at the end of example routes for all considered train types through the Simplon corridor with respect to time discretizations varying from 0 to 60 seconds. The routes have a length of at most ten macroscopic tracks. It is apparent that cumulative rounding dampens the propagation of discretization errors substantially already for short routes.

2.3.2 Headway Times

Based on the occupation and release times in Figure 3, it is possible to define a minimal time difference after which a train can succeed another train on the same track or after which a train can pass another train from the opposite

Algorithm 1: Cumulative rounding method for computing discretized macroscopic running times.

Data: Track $j = (s_1, s_2) = (e_1, \ldots, e_m) \in J$ with $s_1, s_2 \in S$ and $e_i \in E$, $i = 1, \ldots, m$, a train route $r \in R$ with microscopic running time $\tilde{d}_j^r$ for track j, a cumulative rounding error ϵ_{j-1}^r and a time discretization $\Delta > 0$

Result: running time d_j^r and cumulative rounding error ϵ_j^r

begin

  choose $k \in \mathbb{N}$ **with** $(k-1)\Delta < \tilde{d}_j^r \leq k\Delta$;

  if $0 < (k-1)$ ***and*** $\tilde{d}_j^r - (k-1)\Delta \leq \epsilon_{j-1}^r$ **then**

    $d_j^r := (k-1)\Delta$; `// round down`

    $\epsilon_j^r := \epsilon_{j-1}^r - (\tilde{d}_j^r - (k-1)\Delta)$; `// decrease rounding error`

  else

    $d_j^r := k\Delta$; `// round up`

    $\epsilon_j^r := \epsilon_{j-1}^r + (k\Delta - \tilde{d}_j^r)$; `// increase rounding error`

  end

  return *pair*(d_j^r, ϵ_j^r) ;

end

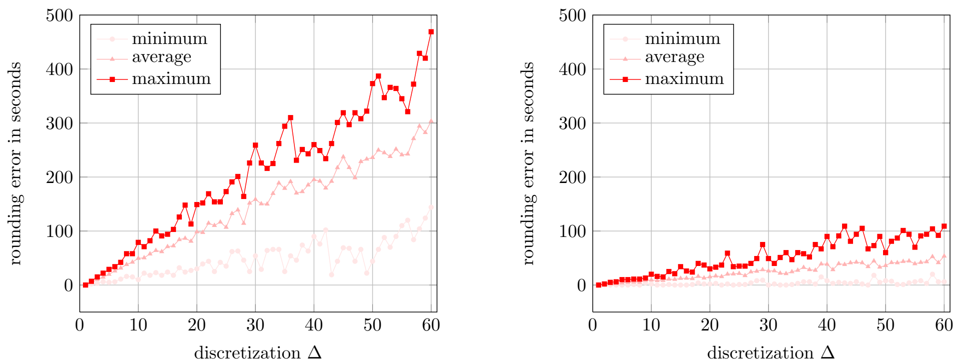

Figure 7: Comparing errors from ceiling rounding (left) and cumulative rounding (right) for different time discretizations varying between 1 and 60 seconds.

direction. (We restrict ourselves w.l.o.g. to the consideration of minimal headway times for the combination of departure events.) Algorithm 2 describes the calculation of the *minimal headway time* for the case of two routes r_1 and r_2 that traverse a track in the same direction. (We assume that both trains have the same departure time at s_1 when calculating the blocking times.) Here, we denote the corresponding train types by $c_1, c_2 \in C$.

In case of crossing or opposite routes r_1 and r_2 on a *single-way* track $j = (s_1, s_2)$, the headway time is calculated differently. By definition each single-way track j has exactly one counterpart $\bar{j} = (s_2, s_1) \in J$, which is directed in the opposite direction, and block feasibility with respect to this opposite direction must be ensured by means of a second headway matrix. The entries of

Algorithm 2: Rounding method for computing discretized minimal headway times.

```
Data: Track j = (s_1, s_2) = (e_1, ..., e_m) ∈ J with s_1, s_2 ∈ S, relative
      release duration u_{e_i}^{r_1} and relative reservation duration l_{e_i}^{r_2} with
      r_1, r_2 ∈ R, c_1, c_2 ∈ C, e_i ∈ E, i = 1, ..., m, time discretization
      Δ > 0.
Result: Minimal headway time h(j, j, c_1, c_2) for train type sequence
        c_1, c_2 on track j
begin
    h ← 0;
    for x = {e_i | e_i ∈ r_1 ∩ r_2} do
        h = max{u_x^{r_1} + l_x^{r_2}, h} ;          // update timing separation
    end
    return ⌈h/Δ⌉;
end
```

this matrix are calculated as follows. Let $j = (e_1, \ldots, e_m)$ be traversed by the directed route r_1. Then the minimum headway time for a departure of a train of type c_2 on an opposite route r_2 at station s_2 after a departure of a train of type c_1 on route r_1 from station s_1 is:

$$h(j, \bar{j}, c_1, c_2) = \sum_{i=1}^{m-1} d_{e_i}^{r_1} + u_{e_m}^{r_1} + l_{e_m}^{r_2}. \tag{1}$$

This time can be discretized by rounding. In practice additional buffer times are added to all headway times in order to increase the robustness of the timetable.

2.4 An Algorithm for micro-macro transformation

Algorithm 3 puts the pieces together in order to transform a railway infrastructure network from a microscopic level to a macroscopic level. The method has been implemented in a software tool NETCAST [12]. The procedure consists of three main steps, namely, *macroscopic network detection* (ND), *aggregation* (AG), and *time discretization* (TD), which will be discussed in this subsection.

Macroscopic network detection (ND) means to construct the digraph $N = (S, J)$ from the microscopic network $G = (V, E)$ and a set of train routes R. Denote by $B(r)$ the set of stations visited by route $r \in R$, i.e., the set of microscopic nodes where the train could stop and/or is allowed to wait. All visited stations become macroscopic station nodes. If an interaction, i.e., a convergence, divergence, or crossing, between two routes is detected, one or two pseudo stations are created, respectively. This detection is done by a simple pairwise comparison of train routes. An important aspect of network detection is that the mapping from a microscopic node to its macroscopic representative is unique, i.e., a microscopic node belongs to at most one junction or station in the microscopic model and hence to at most one (pseudo) station.

The resulting set of stations S_{tmp} is further compressed in the *aggregation* (AG) step by the routine *aggregateStations*(), that enforces the imaginable aggregations as informally described in Section 2.2. At this point, the macroscopic

Algorithm 3: An Algorithm for micro-macro transformation.

Data: microscopic infrastructure graph $G = (V, E)$, set of routes R, stations $B(r)$, train types $c(r) \in C, r \in R$, time discretization $\Delta > 0$

Result: macroscopic network $N = (S, J)$, with stations S and tracks J

```
begin
ND    S_tmp := ∅;
      foreach r ∈ R do
          foreach b ∈ B(r) do
              create s ;                                // create standard station
              S_tmp = S_tmp ∪ {s}
      foreach (r1, r2) ∈ (R × R) do
          while divergence or convergence between r1 and r2 is found do
              create p ;                                // create pseudo station
              S_tmp = S_tmp ∪ {p};
          while crossing between r1 and r2 is found do
              create p, q ;                             // create pseudo stations
              S_tmp = S_tmp ∪ {p, q};
AG    S := aggregateStations(S_tmp);
      J := {(s1, s2) ∈ S × S | ∃r ∈ R with s2 = nextStation(r, s1)};
TD    foreach j ∈ J do
          foreach r ∈ R do
              d_j^{c(r)} := calculateRunningTime(j, r, c, Δ);
          foreach (r1, r2) ∈ (R × R) do
              h(j, j, c(r1), c(r2)) =
              max{h(j, j, c(r1), c(r2)), calculateHeadway(j, r1, r2, Δ)};
              if j is single-way then
                  h(j, j̄, c(r1), c(r2)) =
                  max{h(j, j̄, c(r1), c(r2)), calculateHeadway(j, j̄, r1, r2, Δ)};
      return N = (S, J);
end
```

network detection is finished with respect to the set of stations. It remains to divide the routes R into tracks with respect to the macroscopic stations S. Here, $nextStation(r, s)$ denotes the subsequent station of station s on train route r. It is important to note that there can be more than one track between two stations, especially after aggregation steps have been carried out. A typical example are two tracks between two aggregated macroscopic stations, that correspond to physically different microscopic track sections.

The *time discretization* (TD), the calculation of the rounded running and headway times, is the last step of the algorithm. We denote the running time of train type c over track j on route r by $d_j^{c(r)}$, the headway time for the self correlation case, i.e., when a train on route r_2 follows a train with

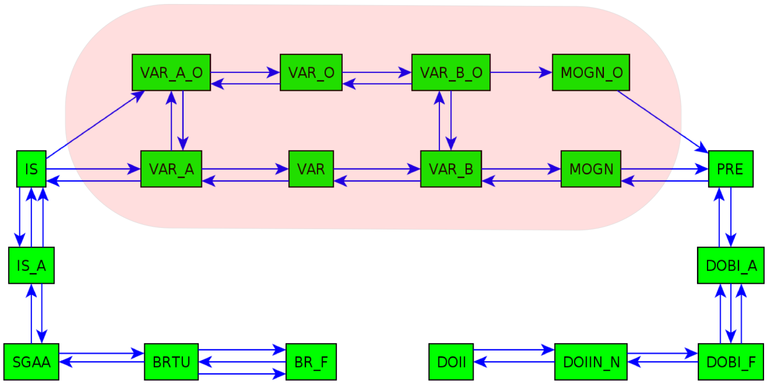

Figure 8: Constructed aggregated macroscopic network by NETCAST.

route r_1, by $h(j, j, r_1, r_2)$, and the headway time for the single-way case by $h(j, \bar{j}, r_1, r_2)$. The running times are calculated by the cumulative rounding procedure $calculateRunningTime()$ according to Algorithm 1. The function $calculateHeadway()$ provides the headway times according to Algorithm 2 and formula (1). The running times for each route, and the headway times for each pair of routes are calculated and (conservatively) aggregated according to the assignment of routes to train types $c \in C$. If there are several routes for the same train type, the maximum running and/or headway time is taken. We remark that we have omitted a discussion of so-called *running modes* of trains (stopping in or passing through a station) in this exposition, but running and headway times with respect to running modes are implemented in the micro-macro transformation tool NETCAST.

Figure 8 shows a macroscopic network model for the Simplon corridor that has been generated using Algorithm 3. We summarize the resulting macroscopic data:

- ▷ (directed) network $N = (S, J)$ with abstract stations S and tracks J,
- ▷ mapping of sub-paths of routes to tracks,
- ▷ mapping of microscopic nodes to stations,
- ▷ running times on all tracks for all C measured in Δ,
- ▷ headway times on all tracks for all pairs of C measured in Δ,
- ▷ (opposite direction) headway times on single-way tracks for all pairs of C measured in Δ.

3 Case Study Simplon

We tested our micro-macro transformation approach on real world data for the Simplon corridor as already mentioned in the introduction. The first step was to choose six standard train types, namely, two types of passenger trains, regional (R) and intercity trains (EC), one motor-rail type *GV Auto*, and three types of freight trains, viz., standard freight trains *GV MTO*, container trains *GV SIM*, and “rolling highway” trains *GV RoLa*. Trains of the latter two types must not use tracks on one side of the Iselle-Preglia tunnel because of their width. This necessitates a so-called *slalom route* when such trains depart

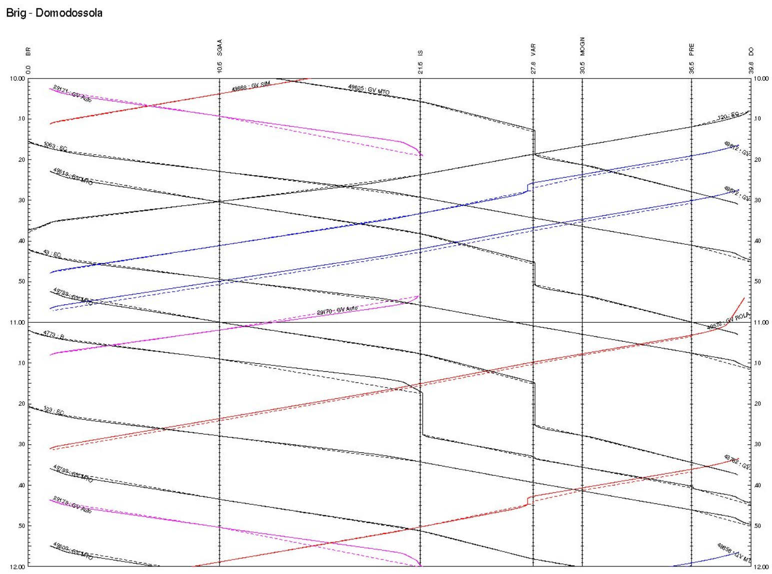

Figure 9: `OpenTrack` traffic diagram of a re-transformed conflict free timetable computed using a micro-macro transformation.

from Brig. The passenger train slots were given as fixed, i.e., our case study dealt with the saturation of the corridor by freight trains subject to a given passenger timetable. For these six aggregated train types we considered up to 28 different routes through the microscopic network $G = (V, E)$. These differ in their stopping patterns and in their routing through the important station Varzo, where over-width trains can pass each other.

In addition to the 12 existing stations, some pseudo-nodes were defined in order to model all train interactions correctly. Detecting convergences, divergences, and crossings as described in Section 2 produces a network $N = (S, J)$ with 55 station nodes and 87 tracks. Conducting some further aggregations, especially in station areas, we constructed a network *simplon_big* with 18 stations and 40 tracks. A second network *simplon_small* with 12 stations and 28 tracks was built with an even coarser station model.

The macroscopic model was verified using a dense manual reference timetable created by the authors. This timetable runs 14 passenger and 21 freight trains in the time window from 8 to 12am through the Simplon. We abused our slot optimization module `TS-OPT` [2] to reproduce this timetable in our macroscopic model by requesting exactly these 35 trains. And indeed, if a fine discretization of, e.g., $\Delta = 6$ seconds, is used, it is possible to reproduce the timetable accurately. Figure 9 compares the reproduced macroscopic timetable and its re-transformed microscopic counterpart as simulated using `OpenTrack`. The dotted lines represent macroscopic train movements; they are linear. The "real" (simulated) timetable is plotted using solid lines; here, acceleration and braking phases are clearly perceivable.

With an accurate macroscopic model, we set out for optimization runs. The goal was to saturate the residual capacity of the corridor (remember the passenger trains are given as fixed) by scheduling additional freight trains ($GV\ MTO$,

Table 1: Solution statistics for several time discretizations for a macroscopic railway network with 12 stations and 28 tracks; ip denotes results for the slot allocation problem formulated as an integer program, and lp for its linear relaxation.

instance	*request1*			*request2*		
discret. in sec.	6	10	30	6	10	30
time(lp) (sec.)	135.67	48.88	17.77	190.36	64.59	2.83
time(ip) (sec.)	72774.55	12409.19	110.34	2923.76	2639.62	34.83
#trains	196	187	166	176	163	143

GV SIM, *GV RoLa*). To this purpose, we defined some artificial demand by creating two sets of train requests covering a 24h time horizon. Both of these sets feature a lot of competing train slots. The first set, *request1*, contains 390 slot requests including 63 fixed passenger trains; this set contains our manually constructed test timetable. The second set, *request2*, contains 255 slot requests (including the passenger train requests); in this set, the freight train requests are uniformly distributed over the time horizon. The objective was a profit for each train request minus a penalty for deviations from optimal arrival and departure times. We remark that our study ignores certain capacity restrictions in the station areas at Brig and Domodossola. All computations were done on machines with a 3 Ghz Intel Quad Core Processor and 8 GB RAM on Suse-Linux 11.2. CPLEX 12.1. was used as a LP and MIP solver [21].

The scenarios for request set *request1* could be solved to proven optimality for both networks using time discretizations of 30 seconds or coarser. The solution for *simplon_big* (with 10s discretization) exhibits a timetable with a theoretical capacity of 203 trains, i.e., the optimized timetable manages to run 140 freight trains between the passenger trains through the Simplon. If we add buffer times and adjust headway times according to some local characteristics, a more realistic schedule with 170 trains can be computed, which is almost identical to the one that is currently in operation. These results demonstrate the accuracy as well as the potential of the method.

We finally analyzed the influence of different time discretizations on solution time and quality. Table 1 shows the results for request sets *request1* and *request2* using the *simplon_small* network. As expected, a coarser time discretization reduces solution times, but decreases solution quality (in terms of numbers of trains). It was, however, a surprise for us that the effect is already so large in this range. This hints at a potential for finer, more “local” time discretization methods. We also remark that larger buffer times reduce the effects of a coarser discretization. For a more detailed description of the Simplon case study we refer to [1].

4 Conclusion

In this paper we proposed an algorithmic bottom-up approach to transform a microscopic railway network to an aggregated macroscopic network model and back. The transformation is done in such a way that the macroscopic model contains all the information that is necessary in order to compute a conflict-free slot allocation. Our micro-macro transformation algorithm detects the macroscopic network structure by analyzing interactions between standard train routes. In this way, the algorithm can ignore or compress parts of the network that are not used by the considered train routes, and still account for all route conflicts by constructing suitable pseudo stations. Time is discretized by a cumulative rounding procedure that minimizes the differences between aggregated and real running times. We tested our approach at the example of the challenging Simplon railway corridor. Our micro-macro transformation approach produced macroscopic models of the Simplon corridor that were small enough to allow for a simultaneous optimization of more than 300 train slots. In this way, it was possible for the first time to compute an operable (i.e., operable in our simulation setting) 24h timetable for the Simplon corridor by an optimization algorithm. Another important issue is the use of our method in larger networks than a corridor like the Simplon. We feel confident that our method is also applicable to more complex settings, but so far no exact microscopic infrastructure data for other networks is available to us.

Acknowledgments

We thank Martin Balser for contributions to time discretization and cumulative rounding, and Daniel Hürlimann for his excellent support for the simulation tool `OpenTrack`. Finally, we thank Martin Grtschel, Gottfried Ilgmann and Klemens Polatschek for their important support in organising und realizing this project, and in particular the Simplon case-study.

References

[1] Ralf Borndörfer, Berkan Erol, Thomas Graffagnino, Thomas Schlechte, and Elmar Swarat. Optimizing the Simplon railway corridor. Technical Report ZIB Report 10-24, Zuse-Institut Berlin, Takustr. 7, 14195 Berlin, 2010.

[2] Ralf Borndörfer, Berkan Erol, and Thomas Schlechte. Optimization of macroscopic train schedules via TS-OPT. In I.A. Hansen, E. Wendler, U. Weidmann, M. Lüthi, J. Rodriguez, S. Ricci, and L. Kroon, editors, *Proceedings of the 3rd International Seminar on Railway Operations Modelling and Analysis - Engineering and Optimisation Approaches*, 2009.

[3] Ralf Borndörfer, Martin Grötschel, and Marc E. Pfetsch. A column-generation approach to line planning in public transport. *Transportation Science*, 41(1):123–132, 2007.

[4] Ralf Borndörfer and Thomas Schlechte. Models for railway track allocation. In Christian Liebchen, Ravindra K. Ahuja, and Juan A. Mesa, editors, *ATMOS 2007 - 7th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems*, Dagstuhl, Germany, 2007. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany.

[5] U. Brännlund, P. O. Lindberg, A. Nou, and J.-E. Nilsson. Railway timetabling using lagrangian relaxation. *Transportation Science*, 32(4):358–369, 1998.

[6] Olaf Brünger and Elias Dahlhaus. *Running time estimation*, chapter 4, pages 58–82. Railway Timetable & Traffic. Eurailpress, 2008.

[7] X. Cai and C. J. Goh. A fast heuristic for the train scheduling problem. *Comput. Oper. Res.*, 21(5):499–510, 1994.

[8] Gabrio Caimi. *Algorithmic decision support for train scheduling in a large and highly utilised railway network*. PhD thesis, ETH Zrich, 2009.

[9] Alberto Caprara, Matteo Fischetti, and Paolo Toth. Modeling and solving the train timetabling problem. *Operations Research*, 50(5):851–861, 2002.

[10] Alberto Caprara, Laura Galli, and Paolo Toth. Solution of the train platforming problem. In Christian Liebchen, Ravindra K. Ahuja, and Juan A. Mesa, editors, *ATMOS*, volume 07001 of *Dagstuhl Seminar Proceedings*. Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany, 2007.

[11] Daniel Emery. Enhanced ETCS Level 3 train control system. In A. Tomii, J. Allan, E. Arias, C.A. Brebbia, C. Goodman, A.F. Rumsey, and G. Sciutto, editors, *Computers in Railways XI*. WIT Press, 2008.

[12] Berkan Erol. Models for the train timetabling problem. Diploma thesis, TU Berlin, 2009.

[13] Berkan Erol, Marc Klemenz, Thomas Schlechte, Sören Schultz, and Andreas Tanner. TTPLIB 2008 - A library for train timetabling problems. In A. Tomii J. Allan, E. Arias, C.A. Brebbia, C. Goodman, A.F. Rumsey, G. Sciutto, editor, *Computers in Railways XI*. WIT Press, 2008.

[14] Frank Fischer, Christoph Helmberg, Jürgen Janßen, and Boris Krostitz. Towards solving very large scale train timetabling problems by lagrangian relaxation. In Matteo Fischetti and Peter Widmayer, editors, *ATMOS 2008 - 8th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems*, Dagstuhl, Germany, 2008. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany.

[15] A. Gille, M. Klemenz, and T. Siefer. *Applying multiscaling analysis to detect capacity resources in railway networks*, chapter A 7, pages 73–82. Timetable Planning and Information Quality. WIT Press, 2010.

[16] IA Hansen. *State-of-the-art of railway operations research*, chapter A 4, pages 35–47. Timetable Planning and Information Quality. WIT Press, 2010.

[17] IA Hansen and J. Pachl. *Railway, Timetable & Traffic*. Eurailpress, Hamburg, 2008.

[18] Oskar Happel. *Ein Verfahren zur Bestimmung der Leistungsfähigkeit der Bahnhöfe*. PhD thesis, RWTH Aachen, 1950.

[19] Oskar Happel. Sperrzeiten als Grundlage für die Fahrplankonstruktion. *Eisenbahntechnische Rundschau (ETR)*, 2, 1959.

[20] Daniel Hürlimann. *Object oriented modeling of infrastructure elements and business processes in railways.* PhD thesis, ETH Zürich, 2001.

[21] IBM ILOG CPLEX, IBM Software Group. *User-Manual CPLEX 12.1*, 2009.

[22] Sebastian Georg Klabes. *Algorithmic railway capacity allocation in a competitive European railway market.* PhD thesis, RWTH Aachen, 2010.

[23] C. Liebchen. *Periodic timetable optimization in public transport.* Springer, 2006.

[24] R. Lusby, J. Larsen, D. Ryan, and M. Ehrgott. Routing trains through railway junctions: A new set packing approach. Technical report, Informatics and Mathematical Modelling, Technical University of Denmark, DTU, Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby, 2006.

[25] Markus Montigel. Formal representation of track topologies by double vertex graphs. *Proceedings of Railcomp 92 held in Washington DC, Computers in Railways*, 3, 1992.

[26] Markus Montigel. *Modellierung und Gewährleistung von Abhängigkeiten in Eisenbahnsicherungsanlagen.* PhD thesis, ETH Zurich, 1994.

[27] Andrew Nash and Daniel Huerlimann. Railroad simulation using OpenTrack. In *Computer Aided Design, Manufacture and Operation in the Railway and other Advanced Transit Systems*, 2004.

[28] Jörn Pachl. *Systemtechnik des Schienenverkehrs.* Teubner Verlag, Stuttgart-Leipzig-Wiesbaden, 3 edition, 2002.

[29] Jörn Pachl. *Timetable design principles*, chapter 2, pages 9–42. Railway Timetable & Traffic. Eurailpress, 2008.

[30] G. Potthoff. *Verkehrsströmungslehre, Band 1- Die Zugfolge auf Strecken und in Bahnhöfen, 3. Auflage.* transpress Verlag Berlin, 1980.

[31] Alfons Radtke. *Infrastructure Modelling*, chapter 3, pages 43–57. Railway Timetable & Traffic. Eurailpress, 2008.

[32] K. Schultze. *Modell für die asynchrone Simulation des Betriebes in Teilen des Eisenbahnnetzes.* PhD thesis, RWTH Aachen, 1985.

[33] Wulf Schwanhäußer. *Die Bemessung der Pufferzeiten im Fahrplangefüge der Eisenbahn.* PhD thesis, RWTH Aachen, 1974.

[34] Wulf Schwanhäußer, Ingolf Gast, Kurt Schultze, and Olaf Brünger. Programmfamilie SLS Benutzerhandbuch. Technical report, Deutsche Bundesbahn, 1992.

[35] P. Serafini and W. Ukovich. A mathematical model for periodic scheduling problems. *SIAM Journal on Discrete Mathematics*, 2:550–583, 1989.

[36] Thomas Siefer and Alfons Radtke. Railway-simulation key for better operation and optimal use of infrastructure. In *Proceedings of the 1st International Seminar on Railway Operations Modelling and Analysis*, 2005.

[37] Ekkehard Wendler. *Analytische Berechnung der planmässigen Wartezeiten bei asynchroner Fahrplankonstruktion.* Verkehrswiss. Inst. der Rheinisch-Westfälischen Techn. Hochsch. Aachen, 1999.

[38] Ekkehard Wendler. Influence of ETCS on the capacity of lines. In *Compendium on ERTMS*, Compendium on ERTMS: European Rail Traffic Management System. Eurailpress, 2009.

[39] Peter J. Zwaneveld, Leo G. Kroon, H. Edwin Romeijn, Marc Salomon, Stephane Dauzere-Peres, Stan P. M. Van Hoesel, and Harrie W. Ambergen. Routing Trains Through Railway Stations: Model Formulation and Algorithms. *Transportation Science*, 30(3):181–194, 1996.