Microscopic Models and Network Transformations for Automated Railway Traffic Planning

Besinovic, Nikola; Goverde, Rob M P; Quaglietta, Egidio

DOI
10.1111/mice.12207

Publication date
2016

Document Version
Final published version

Published in
Computer-Aided Civil and Infrastructure Engineering

COMPUTER-AIDED CIVIL AND INFRASTRUCTURE ENGINEERING

Computer-Aided Civil and Infrastructure Engineering **00** (2016) 1–18

Microscopic Models and Network Transformations for Automated Railway Traffic Planning

Nikola Bešinović & Rob M.P. Goverde*

Faculty of Civil Engineering and Geosciences, Department of Transport and Planning, Delft University of Technology Stevinweg 1, 2628 CN Delft, the Netherlands

&

Egidio Quaglietta

Network Rail Ltd, Network Optimisation Team, Command Control & Signalling, Milton Keynes MK9 1EN, United Kingdom

Abstract: *This article tackles the real-world planning problem of railway operations. Improving the timetable planning process will result in more reliable product plans and a higher quality of service for passengers and freight operators. We focus on the microscopic models for computing accurate track blocking times for guaranteeing feasibility and stability of railway timetables. A conflict detection and resolution model manages feasibility by identifying conflicts and computing minimum headway times that provide conflict-free services. The timetable compression method is used for computing capacity consumption and verifying the stability according to the UIC Capacity Code 406. Furthermore, the microscopic models have been incorporated in a multilevel timetabling framework for completely automated generation of timetables. The approach is demonstrated in a real-world case study from the Dutch railway network. Practitioners can use these microscopic timetabling models as an important component in the timetabling process to improve the general quality of timetables.*

*To whom correspondence should be addressed. E-mail: *r.m.p.goverde@tudelft.nl*.

1 INTRODUCTION

Timetabling is one of the major planning tasks in railway traffic and becomes increasingly complicated with the increasing demand for more services. Planners are constantly under pressure to fit additional trains into busy schedules while at the same time maintaining and improving the level of service such as seamless connections and punctuality. Timetables need to provide accurate time–distance infrastructure slots, or train paths, that secure conflict-free train runs. Moreover, the plan must adhere to daily stochastic variations in the train services, that is, be robust.

Integrated automatic timetabling models provide fast solutions that allow analyses of multiple timetable scenarios and tweaking different planning criteria. This will eventually lead to a better understanding of the capacity use and overall high-quality timetables. Tsiflakos and Owen (1993) already stressed the importance of automated decision support and presented a structural representation of railway data necessary for any further application of optimization techniques. Indeed, there is an evident need for modeling approaches that allow an efficient use of optimization algorithms and other supporting models in timetabling.

We make a distinction regarding the level of detail considered in timetabling. Two approaches can be recognized—microscopic and macroscopic. The latter considers the railway network at a higher level, in which

DOI: 10.1111/mice.12207

station is represented as a node and tracks by linking arcs. In a microscopic approach, detailed infrastructure aspects like speed limits, gradients, curves and signaling system are considered. In this article, we introduce microscopic models that can accurately evaluate timetables and support macroscopic models to construct operationally acceptable timetables which are feasible and stable. The railway research on both microscopic and macroscopic models attracted significant research (Peng et al., 2011; Xie et al., 2014; Castillo et al., 2015; Sels et al., 2016).

An extensive review of timetabling models is given in Cacchiani and Toth (2012). Kroon et al. (2009) presented the practical implementation of a set of optimization models for the Netherlands Railways. These optimization models assumed a macroscopic infrastructure model using default norms for safe separation times of following, crossing and meeting trains. This normative approach cannot guarantee to solve all route conflicts in the computed (macro) timetable, or on the other hand may lead to inefficient large buffer times. Moreover, scheduling train paths over the given infrastructure and the capacity assessment of the resulting timetable are separated processes. Therefore, macroscopic approaches should be integrated with more detailed models that ensure the operational feasibility of the timetable.

Timetable feasibility is the ability of all trains to adhere to their scheduled train paths. A timetable is feasible if (1) the individual processes are realizable within their scheduled process times, and (2) the scheduled train paths are conflict free, that is, all trains can proceed undisturbed by other traffic. A *conflict* is defined as an overlap (in time and space) between two trains on the same route which represents that one train cannot use the railway infrastructure without interfering with the other train. A few approaches have been proposed in literature based on a hierarchical integration of timetabling models with different level of details (Schlechte et al., 2011; Gille et al., 2008; Caimi et al., 2011; De Fabris et al., 2013). The current integrated models using microscopic details for timetabling do not perform any feasibility check of the timetable produced, except for Schlechte et al. (2011); while none of them considers any iterative modification to the timetable if it is proved to be infeasible at the microscopic level. In other words, Schlechte et al. used a microsimulation for conflict detection, while none of the approaches consider any conflict resolution methods. Hence, these models solve the timetabling problem in one direction only and thus represent an open-loop strategy.

D'Ariano et al. (2007) proposed a model for real-time train rescheduling that includes a feasibility check and recomputing speed profiles with some simplifying assumptions, such as trains running at maximum speeds with possible braking at conflicts, a simplified interlocking model at station layouts, and fixed speed-independent clearing times. In our model, we explicitly compute operational running times, sight, setup and clearing times, and consider track sections instead of block sections which in particular matters in station areas with switches. This provides a more accurate conflict detection. A review of other real-time rescheduling models can be found in Cacchiani et al. (2014).

Timetable stability is defined as the capability of absorbing train delays (UIC, 2013). As a stability measure, we adopted the UIC (International Union of Railways) recommendation that a timetable is stable if capacity occupation rates are under certain norms depending on the traffic structure. *Capacity occupation* is defined as the time share needed to operate trains according to a given timetable pattern taking into account scheduled running and dwell times. Thus, we first compute the capacity occupation for stations and corridors and then compare obtained values with the UIC norms. The current practice of a posteriori capacity assessment of the final timetable is not efficient: a lot of time may be invested in producing a timetable that afterwards may not satisfy the stability norms.

Within tactical railway planning, capacity assessment is generally based on the microscopic UIC compression method (UIC, 2013; Landex and Jensen, 2013), while stability on the network level can be assessed by the stability analysis tool PETER (Goverde, 2007). The UIC method has been developed for assessing lines and corridors. However, the main limitation of the UIC method is that it computes the capacity in station areas by considering the platform tracks separately from the interlocking areas in between the home signals and the platform tracks (Lindner, 2011; Armstrong et al., 2015). This independence assumption results in an underestimated station capacity.

Nash and Huerlimann (2004), Siefer and Radtke (2006), and Quaglietta (2014) presented advanced microscopic simulation tools, which are able to accurately simulate railway operations based on a detailed modeling of infrastructure, signaling and train dynamics that could be used to detect conflicts in a timetable. However, these multipurpose microscopic simulation models need long computation times to evaluate conflict-freeness of timetables on large and heavily utilized railway networks. Therefore, they are not suitable for fast analyses during the design of a timetable.

Train running time computations are one of the most common models in railway applications, and have been used for computing minimum running times in timetable planning or for energy efficient driving in real-time applications. These models are commonly

including principles of optimal control theory. A detailed review can be found in Albrecht et al. (2013), or Scheepmaker and Goverde (2015). *An operational speed profile* is the one that exploits existing time supplements between departure and arrival times to allow the train arriving on time, instead of being ahead of schedule. The operational speed profile is used to assert that an acceptable speed profile exists for allocated time supplements. For example, it may occur that a macroscopic timetable assigned an excessive running time supplement that would require a train to run very slow below a certain practical minimum speed. Such a speed profile should be avoided. Second, we need the operational speed profiles for detecting timetable conflicts and assessing the infrastructure capacity.

Communication between microscopic and macroscopic models is essential for efficient and consistent bidirectional transformations. These transformations would allow generating accurate input to a macroscopic model on one side, and evaluating a timetable on the detailed microscopic level on the other. Schlechte et al. (2011) introduced a micro to macro transformation, but the reverse transformation from macro to micro has not been described in the literature yet.

The state-of-the-practice suggests that improvements in the timetable planning process are necessary in various directions (ON-TIME, 2016). Most notably, a timetable is expected to be realizable considering a great level of detail including infrastructure, rolling stock, signaling and automatic train protection (ATP). Second, timetabling tools should work as a whole, as well as in terms of individual functions, that is, a stepwise development is recommended. Third, the final timetable should satisfy specified values for performance measures such as feasibility, capacity occupation, robustness, and energy consumption (Goverde and Hansen, 2013). Finally, it is important to reduce the computation time of the planning tools.

To overcome the limitations in the state-of-the-art and answer the questions from practice, we developed a hierarchical framework of performance-based railway timetable design in the European FP7 project ON-TIME (Optimal Networks for Train Integration Management across Europe) (Goverde et al., 2016). In particular, the framework includes microscopic models presented in this article and a macroscopic timetabling model that interact iteratively by adapting microscopic running and minimum headway times until the produced macroscopic timetable is proved feasible and stable.

The aim of this article is to provide a methodology for timetable design that will cater for more structural insight into a timetable and make the process itself more efficient, which would result in timetables of a higher overall quality. In the past, we introduced a conceptual ON-TIME framework (Bešinović et al., 2014). In this article, we describe the deterministic microscopic timetabling models and provide efficient automatic transformations between microscopic and macroscopic networks. Microscopic models compute accurate running and minimum headway times that are used as input to a macroscopic model, and verify that the timetables produced by the latter are feasible at the level of track sections. For timetable evaluation, and particularly the micro–macro framework, operational speed profiles may be recomputed numerous times. Thus, we define a new model for fast computing operational speed profiles, although various models based on optimal control exist in the literature. Stability is checked by verifying that the infrastructure capacity occupation respects the UIC guidelines (UIC, 2013). We propose an analytic model for capacity assessment that efficiently deals with both stations and corridors. Network transformations are required to provide the relevant data for specific computations. Aggregating the data to a macroscopic level allows the application of macroscopic optimization models while considering a consistent operationally relevant railway infrastructure. After computing a macroscopic timetable, the reverse transformation is applied from macro to micro. This is done by recomputing the operational speed profiles with respect to the arrival/departure times from the macroscopic timetable. All microscopic models can be used for designing and evaluating both periodic and nonperiodic timetables and each model can be used individually or as a building element of the timetabling framework. The microscopic models have been tested on a part of the Dutch railway network including the main corridor Utrecht–Den Bosch–Eindhoven.

The main contributions of this article are the following:

- fast computation of operational train trajectories from scheduled event times that enable microscopic timetable evaluation;
- capacity assessment based on max-plus automata that compute the capacity occupation in stations more realistically than the current UIC method;
- automatic conflict detection that accurately determines existing conflicts at the level of track sections;
- consistent network transformations from micro to macro and vice versa.

The remainder is organized as follows. Section 2 gives the structure of the general framework. Section 3 describes the network and data modeling. It also includes conversions from micro to macro and vice versa. Section 4 presents a detailed description of the microscopic modules and their functions. Further, it

introduces the basics of max-plus automata theory and its application to the UIC compression method. Section 5 illustrates the approach in a Dutch case study. Section 6 reflects on the developed models and finally Section 7 presents conclusions and future research.

2 THE MICRO–MACRO TIMETABLING APPROACH

The ON-TIME project defined a framework for achieving high-quality railway timetables with an integrated set of state-of-the-art timetabling techniques. More details about the models used and the framework developed can be found in (Goverde et al., 2016). One of the main objectives of the project was to build up "a scheduled train-path assignment application, with automatic conflict detection capabilities, that builds on the concept of robust timetables, has a unified network coverage, is microscopic at selected parts of the control area, is scalable, and able to connect to Traffic Management Systems, with user-friendly interfaces and execution states that correspond to the IM timetabling management milestones." This objective has been reached by the two-level functional framework represented in Figure 1, which indicates the interactions among the microscopic and macroscopic models.

Input data of the framework are microscopic characteristics of the infrastructure (e.g., track gradients, position of stations, switches), the rolling stock (e.g., mass, number of coaches, tractive effort-speed curve, resistance parameters), the signaling and ATP system (braking behavior, signal aspect sequence) and the interlocking (e.g., local feasible routes). Both input and output of the framework are in a standardized railway data format, known as RailML (RailML, 2015).

The timetabling computation is an iterative process of two models:

- A microscopic model that computes reliable train running and blocking times at a highly detailed level and checks for feasibility and stability of the timetable,
- A macroscopic model that produces a timetable at aggregated network level, by identifying arrival/departure times at/from stations or junctions to optimize a given objective function (e.g., minimize journey times). This is an optimization model that can also provide timetables that are robust versus stochastic operation disturbances.

In the first iteration a timetable is not available yet, so the microscopic model computes minimum running times and blocking times, which are aggregated

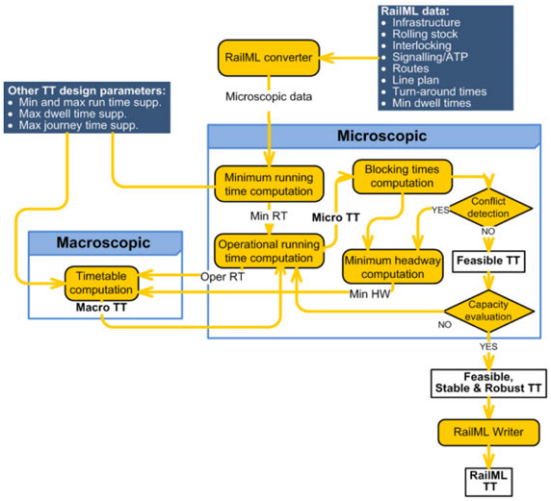

Fig. 1. Scheme of the micro–macro framework for timetable design.

to macroscopic running times and minimum headway times and sent to the macroscopic model to calculate a timetable. When a macroscopic timetable has been produced this is sent back to the microscopic model which computes updated blocking times required for detecting track conflicts based on the operational running times (i.e., the running times including time supplements scheduled by the macroscopic timetable). If there are track conflicts, these are resolved and minimum headway times are computed which are transferred to the macro model again. This iterative process is repeated until all track conflicts have been solved and the macroscopic timetable can be defined as feasible.

In the next step, the microscopic model evaluates the stability of the timetable. If the timetable is not stable enough, new operational running times are computed by, for example, increasing the value of time supplements and/or buffer times. This is performed until the timetable stability is also verified. For the transformations from the microscopic level to the macroscopic level, and vice versa, efficient procedures have been developed to aggregate and disaggregate input and output data. In general, microscopic models are necessary to (1) compute initial input data for the macroscopic timetabling model, (2) assess the timetable feasibility and stability when used independently, and (3) guarantee operational feasibility and stability when included in the micro–macro framework.

3 NETWORK AND DATA MODELLING

As already pointed out by Tsiflakos and Owen (1993), we need to use structurally organized input data. In the

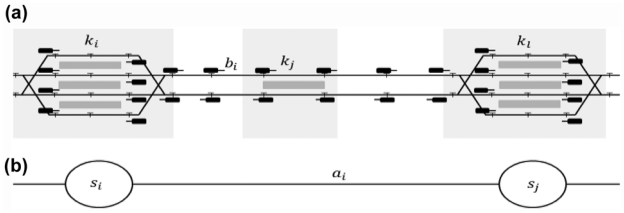

Fig. 2. Representation of a (a) microscopic network and (b) macroscopic network.

past years a significant effort has been seen in defining a standardized railway data format RailML. This RailML data format is more and more adopted for communication between railway software tools, and therefore we also adopted this RailML data exchange format. The input to our models thus consists of a set of RailML files composed of: (1) microscopic infrastructure data, (2) rolling stock data, including train formations, (3) interlocking, signaling and ATP system data, (4) available routes, and (5) train lines. A *train line* is defined with origin and destination points, stopping pattern at timetable points (stations, stops) and a corresponding rolling stock type. It also includes the service category, such as local or intercity, and the frequency represented in number of trains per hour. These data are converted to a suitable internal format of ASCII data which is used by the microscopic models. Additional parameters, such as connections and transfer times, dwell times, and other timetable design parameters and norms are provided externally. The hierarchical framework for timetable design is composed of two network models that respectively represent the same network with a microscopic and a macroscopic level of detail.

3.1 Network modeling

3.1.1 Microscopic network. The microscopic model considers *homogeneous behavioral sections* for the accurate computation of train trajectories and running times (Figure 2a). A homogeneous behavioral section is defined as a section with a certain length l, and constant characteristics of speed limit v_{lim}, gradient g, and radius ρ. The microscopic network is based on a graph whose arcs are obtained by aggregating the homogeneous behavioral sections into track sections denoted by b. A track section corresponds to a track-free detection section, or several track-free detection sections including at most a single switch. On the open track, a block is considered as one track section, while in interlocking areas one block may include multiple sections.

The nodes of the microscopic network coincide with the joints between consecutive block/track sections or to infrastructure elements such as signals, switches, and platforms. This level of infrastructure details allows very accurate infrastructure capacity assessment, feasibility checks and minimized wasted capacity, which is particularly important in highly utilized networks.

We distinguish between functions working on the behavioral section level of the infrastructure network on one hand, and the track section or block section level on the other. Computations of minimum and operational running times and corresponding speed profiles are applied on the former, while computation of blocking times and minimum headway times, conflict detection, and capacity assessment are applied on the latter. We also define a set of *microscopic timetable points* K, where each point k represents an infrastructural point of interest such as *stations* that provide passenger (and/or freight) interaction and allow train overtaking, *stops* that do not have enough tracks to facilitate overtaking or dwelling of more than one train, and *junctions* where two or more railway lines intersect or merge and no trains are scheduled to stop.

3.1.2 Macroscopic network. The macroscopic network $N = (S, A)$ is automatically produced from the microscopic one and used for the macroscopic timetabling model (Figure 2b). Nodes in a macroscopic network are referred to as *macroscopic timetable points*, $s \in S$. The potential candidates for s are stations and junctions from K. An arc $a \in A$ represents the corridor between two successive macroscopic points s_i and s_j. Each arc is comprised of a set of microscopic arcs, $a = (b_1, b_2, \ldots, b_n)$. The generation of a macroscopic network is explained in Section 3.3.

3.2 Timetables, trains, and routes

We distinguish between a microscopic and macroscopic timetable. A *macroscopic timetable* (*macroTT*) includes scheduled running, dwell and transfer times, as well as event times such as arrivals, departures, and passages between and in macroscopic points. A *microscopic timetable* (*microTT*) includes all the aforementioned event times for microscopic timetable points and the corresponding train speed profiles defining the exact train trajectories.

The set of trains is indicated by T. For each train $t \in T$, $S_t \subseteq S$ is a set of served macroscopic timetable points. We assume that for each train the route ρ_t (i.e., the sequence of traversed tracks without the corresponding travel times) is provided. Here, we differentiate between a microscopic route $\rho_t^{micro} = (b_1, b_2 \ldots, b_{n_t})$,

where n_t is the number of microscopic arcs for train t, and a macroscopic one $\rho_t^{macro} = (a_1, a_2 \ldots, a_{m_t})$, where m_t is the number of macroscopic tracks for train t.

For each train $t \in T$ and each macroscopic arc a the minimum running time r_{ta}, the nominal running time $\underline{r}_{ta}$ including a running time supplement, and the maximum running time $\bar{r}_{ta}$ are given. All running times are computed by microscopic algorithms, while the nominal and maximum ones are given as input to the macroscopic model. The scheduled running times in macroTT are called operational running times and denoted as $\tilde{r}_{ta}$. Similarly, we define running times $r_{tk_1k_2}$, $\bar{r}_{tk_1k_2}$, $\tilde{r}_{tk_1k_2}$ between two microscopic points k_i and k_j, representing the minimum, maximum, and scheduled ones, respectively.

For each train $t \in T$ and each microscopic point $k \in K$ the nominal dwell time $\underline{w}_{tk}$ and maximum dwell time $\bar{w}_{tk}$ is provided. Because the aim of timetable planning is to provide an acceptable quality of service, certain *design norms* need to be predefined. The set of these parameters consists of minimum transfer times, turnaround times, minimum and maximum, running time supplements (%), and maximum allowed journey times of train lines (%).

3.3 Microscopic to macroscopic conversion

Algorithm 1 describes the automatic procedure for the micro to macro network and data transformations, which are similar to Schlechte et al. (2011). Our approach differs in two points. First, the algorithm of Schlechte et al. does not compute running or blocking times, but uses the commercial software OpenTrack to do so. Second, their algorithm performs a search over all infrastructure elements (i.e., block sections) to determine macroscopic points, while we do it exclusively over microscopic timetable points. Note that a set of microscopic points is quite extensive and includes much more than just stations and stops, but also each important junction, switch, crossing, movable bridge, or platform. In terms of complexity, this means that our algorithm has significantly less work than that of Schlechte et al., making our model computationally faster. The CPU time for our micro to macro conversion is under one second.

The conversion from microscopic to macroscopic models includes three steps: computing process times, generating a macroscopic network, and aggregating process times for the macroscopic network. The algorithm first computes the minimum running times and corresponding blocking times. Then, it aggregates microscopic arcs (track sections) b_i to macroscopic arcs $a = (b_1, b_2, \ldots, b_n)$. Each arc a is described with the number of tracks and its orientation (mono- or bidirectional). The former is determined by identifying different routes between two nodes using the function *DetermineTracks*, while function *DetermineDirection* determines the latter. The subset of macroscopic points S is then derived from the microscopic points K. The algorithm compares all pairs of train routes separately. The macroscopic point is chosen based on the interplay between train routes. The microscopic point k is in S only if (1) any two routes are converging, diverging, or crossing in k, or (2) k is the origin or destination point of any route. For example, for two routes using microscopic points $\{k_1, k_2, k_3, k_5\}$ and $\{k_1, k_2, k_4\}$, respectively, the set of macroscopic points is $S = \{k_1, k_2, k_4, k_5\}$. Point k_2 is included because it is a diverging point (first criterion), while k_1, k_4, and k_5 satisfy the second criterion.

After initializing the macroscopic network, headways are determined at each macroscopic point s and for all possible interactions between each two train routes. The computation of the blocking times and minimum headway times are executed on the block section level of the infrastructure network. Once all process times are computed on the microscopic models, the algorithm performs the aggregation of process times and the discretization of time. The function *AggregateProcessTimes* aggregates the microscopic running times (i.e., between each two microscopic timetable points) to aggregated process times between two timetable points in the macroscopic network. The minimum running time r_{ta} between two macroscopic points may comprise several microscopic running times and dwell times because $S \subset K$, that is, not all micro points are in S. The nominal running time over a is obtained by adding a running time supplement $\lambda_{\min}$ to the minimum running times plus the intermediate dwell times:

$$\underline{r}_{ta} := \sum_{i=1}^{m} (1 + \lambda_{\min})\, r_{tk_ik_{i+1}} + \sum_{i=1}^{n} w_{k_i}$$

where arc a is bounded by some macro points $[s_i, s_j]$, m is the number of consecutive running sections, and n is the number of intermediate microscopic points between s_i and s_j. Similarly, the maximum running time $\bar{r}_{ta}$ over a is obtained with respect to a maximum running time supplement $\lambda_{\max}$. Initially, $\lambda_{\min}$ is provided such as 5%. In any following iteration it is computed from the macroTT returned by the macroscopic timetable model.

The macroscopic model may use a coarser time granularity, so a time discretization of process times is performed as well. The incorporated function represents an innovative rounding method that has the objective to control the rounding error by combining rounding up and rounding down. By applying

AggregateProcessTimes, we obtain all process times that are necessary for macroscopic computation.

The network transformation is applied in the initial stage of timetable planning to provide the required network input to a macroscopic model because the given line requests (origin/destinations and stop patterns) are considered as fixed. Hence, the macroscopic network structure remains the same during all iterations. On the other hand, *AggregateProcessTimes* is run each time (e.g., iteration) a data input (for a macroscopic model) is adjusted based on the output of the microscopic models such as the updated train speed profiles, running times and headway times that need to be aggregated for each new run of the macro model.

Algorithm 1. Micro to macro conversion

Input: Microscopic network M, microscopic points K, dwell times W, timetable design norms λ, set of trains T
Output: macroscopic network $N = (S, A)$, macroscopic running, dwell and headway times
Forall $t \in T$
  Compute microscopic running times $R_{t,micro}$
  Compute blocking times B_t
End Forall
Forall microscopic timetable points $k \in K$
  Forall pairs of train lines
    If k is origin or destination point OR lines converge OR lines diverge OR lines cross
      add k to macroscopic nodes: $S \rightarrow S \cup \{k\}$
    End If
  End Forall
End Forall
Forall adjacent timetable points $s \in S$
  Create a macroscopic arc $a = (s_i, s_j)$
  DetermineTracks of arc $a = \{b_i\}, i = 1, \ldots, n$
  DetermineDirection of arc a
End Forall
Forall macroscopic timetable points $s \in S$
  Compute minimum headway times $h_{st_it_j}$
End Forall
AggregateProcessTimes for $N = (S, A)$

3.4 Macroscopic to microscopic conversion

After obtaining a macroTT, we need to translate it to a microscopic level of detail in microTT, see Algorithm 2. In other words, from the scheduled event times for the macroscopic timetable points we reconstruct the train trajectories and scheduled times for all microscopic timetable points. To do so, we apply the following three steps for each train. Step 1 derives running time supplements for a macroscopic route ρ_t^{macro} and distributes them to the corresponding microscopic route ρ_t^{micro}. Step 2 determines the operational speed profile for the given time supplements (Section 4.2). Finally, the computation of blocking times concludes Step 3 (Section 4.3). Step 1 is explained in more detail in the following subsection and is followed by an example of the macro to micro conversion.

Algorithm 2. Macro to micro conversion

Input: microscopic network M, *macroTT*
Output: *microTT*
Forall trains $t \in T$
  1. *Determine allocated running time supplements* Ψ_t
  2. *Compute operational speed profiles (see Section 4.2)*
  3. *Compute blocking times* B_t *(see Section 4.3)*
End Forall

3.4.1 Allocation of running time supplements. In Step 1 we determine the running time supplements that are allocated in a given macroTT. Based on the scheduled running time (difference between the scheduled departure time and scheduled arrival time at the next considered point) in macroTT, we compute the corresponding allocated running time supplement between two macroscopic points. We denote ψ_{ta} as the difference between the scheduled and minimum running time for macroscopic arc a of train t, $\tilde{r}_{ta}$ and $\underline{r}_{ta}$, respectively. This defines a vector Ψ_t of the time supplements ψ_{ta} between each two macroscopic timetable points over the corresponding route q_t^{macro}. This is done for all trains $t \in T$.

Recall that not all microscopic timetable points are necessarily also macroscopic, but the macroscopic points are a subset of the microscopic points. This means that several microscopic timetable points may exist between two adjacent macroscopic points. By computing an operational train trajectory over an arc a and considering just a given time supplement ψ_{ta}, one may obtain an unequal distribution of time supplements between two consecutive microscopic timetable points. Hence, we need to migrate from time supplements over arcs, to the lower level, that is, time supplements between each two microscopic points, which results in distributing time supplements in a more justified manner. To do so, we assign ψ_{ta} proportionally to all sections between each two adjacent microscopic points based on the running time over that section. So, each section k_1k_2 receives a portion: $\psi_{tk_ik_j} = \psi_{ta}\ r_{tk_ik_j}/r_{ta}$, where r_{ta} is the minimum running time between two macroscopic points

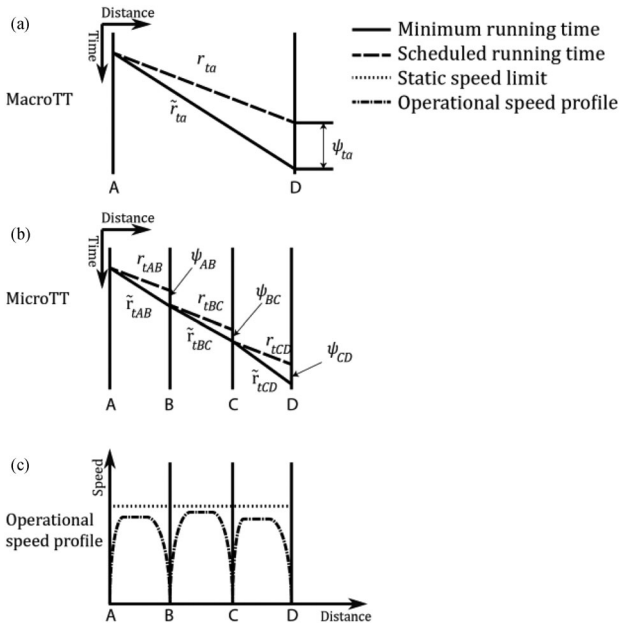

Fig. 3. Macro to micro transformation.

over arc a, $r_{tk_ik_j}$ the one between two microscopic points k_i and k_j and $\psi_{tk_ik_j}$ is the corresponding running time supplement. By doing this, we enforce an equal time supplement distribution and prevent that some sections get no time supplements.

Figure 3 gives a graphical representation of the macro to micro transformation for a given train t operating between A and D. Let A and D be macroscopic timetable points, while B and C the microscopic ones. The macroscopic arc $a = (A, D)$. The train stops at all points. Solid lines represent scheduled running times, and dashed lines are the minimum ones. The macroscopic timetabling model produces (macroTT) the scheduled running time, $\tilde{r}_{ta}$, and corresponding minimum running time r_{ta}. Step 1 computes the running time supplement ψ_{ta}, as $\psi_{ta} = \tilde{r}_{ta} - \underline{r}_{ta}$ (Figure 3a). Then, ψ_{ta} is distributed proportionally between each two neighboring microscopic points (Figure 3b). In Figure 3c, the dotted line is the static speed limit along the route. Step 2 computes the operational speed profiles for each section between two microscopic points (Figure 3c) which is explained in the following section.

4 MICROSCOPIC COMPUTATIONS

4.1 Minimum running times

The minimum running time is the time required for driving a train from one infrastructure point to another assuming conflict-free driving as fast as possible. Running times are computed using microscopic train dynamic models that require detailed rolling stock and infrastructure data, including route-specific static speed profiles.

Running times are modeled by means of the Newton's motion equations (Brünger and Dahlhaus, 2014). The tractive effort is assumed a piecewise function of speed consisting of a linear part and one or more hyperbolic ones. The resistance force is modeled based on the Davis resistance equation, a second-order polynomial of speed. The braking rate is defined as a single deceleration rate. A train speed profile and the associated running time are determined as function of distance (Bešinović et al., 2013). These first-order ordinary differential equations are solved by the numerical Dormand–Prince method (Butcher, 2013), which is a variant of the more general Runge–Kutta approach. The output of this function constitutes microscopic running times $r_{tk_ik_j}$ for each $t \in T$ and where k_i and k_j are the subsequent microscopic points along the route ρ_t^{micro}. It also includes the corresponding train trajectories, that is, time–distance and speed–distance diagrams.

4.2 Operational running time computation

In Step 2, for each train $t \in T$ and corresponding Ψ_t, we compute the operational running time consisting of the detailed train trajectory and scheduled times at microscopic timetable points, which are used for further microscopic analyses as conflict detection and capacity assessment.

By definition, the scheduled running time contains time supplements added to the microscopic minimum running time to absorb a stochastic variation of train runs during real operations. In the initial stage of the timetable planning, the time supplement is usually 5% of the minimum running time, which is a common value for the Netherlands Railways. At the microscopic level, the operational speed profile is obtained by applying cruising with a speed lower than the maximum speed. The insertion of cruising phases at lower speeds is realized by means of a customized bisection algorithm. This identifies the speeds and the cruising phases that return a running time equal to the operational one provided by the timetable.

The input of this model is therefore the arrival/departure times and the operational running times planned in macroTT. The output are the microscopic train trajectories that satisfy the operational running time in microTT. In the following, we leave out the indices to keep the text easier to read.

We focus on computing an operational speed profile between two consecutive stopping points. To acquire the operational profile we use an *operational parameter* p [%], which represents the ratio between the given static speed limit and an actual speed that should be used to consume the given time supplement ψ. Lower and upper bounds for p are 30 and 100, respectively. Lower bound prevents that a train cruises at unacceptably low speed. For example, if the maximum speed is 130 km/h, the minimum allowed speed would be 39 km/h. Upper bound gives the minimum running time. The operational parameter is applied on open-track to exploit the running time supplement, while maintaining the maximum speed through areas with restricted speeds (i.e., sections with the maximum speed of 40 km/h). The running time with respect to the operational parameter is computed by applying the running time function (described in Section 4.1) for adjusted static speed limits over the infrastructure. If several microscopic points exist between two adjacent macroscopic timetable points like stops at the open-track, then for each train line p is a vector with different values between each two microscopic points.

The function uses an adjusted bisection algorithm to find an operational parameter p with a corresponding operational speed profile as described in detail in Algorithm 3. The focus here is on a single section between two microscopic timetable points. The function inputs are the scheduled running time $\tilde{r}$ from the microTT and the microscopic minimum running times r as well as a tolerated error $\epsilon_{tolerance}$ [s], which is applied as a stopping criterion. The algorithm introduces the currently computed running time $r_{current}$ for the given operational parameter and the absolute computed error ϵ_{abs}, that is, the difference between $\tilde{r}$ and $r_{current}$. Initially, $r_{current}$ is set equal to the minimum running time and p is set to 100.

In each repetition, the algorithm

1. computes a speed profile (and running time) for value p,
2. refines the search range $[p_{lb}, p_{ub}]$ for p depending on the relation of $\tilde{r}$ and $r_{current}$,
3. updates p and ϵ_{abs}.

Steps 1–3 are repeated until the absolute error satisfies the stopping criterion.

Consequently, the blocking times are computed for all operational speed profiles and feasibility and stability of the *microTT* is evaluated applying the algorithms described in the next section.

Algorithm 3. Computation of operational speed profile

Input: Micro network, time supplements (from Step 1), $\epsilon_{tolerance}$, train lines T
Output: Operational speed profiles for all train lines
Initialize $p_{lb} = 30$, $p_{ub} = 100$
Forall tuples (train line, running section, time supplement)
  Set bounds for operational parameter p, $[p_{lb}, p_{ub}]$
  Initialize $r_{current} \leftarrow r$, $\epsilon_{abs} \leftarrow |r_{current} - r_{oper}|$, $p \leftarrow 100$
  While $\epsilon_{abs} > \epsilon_{tolerance}$
    $r_{current} \leftarrow$*RunningTimeComputation*(p)
    If $r_{oper} - r_{current} > 0$
      Update lower bound $p_{lb} \leftarrow p + \frac{p_{ub} - p_{lb}}{2}$
      Update operational parameter $p \leftarrow p_{lb}$
    Else
      Update upper bound $p_{ub} = p - \frac{p_{ub} - p_{lb}}{2}$
      Update operational parameter $p \leftarrow p_{ub}$
    End If
    Update error $\epsilon_{abs} \leftarrow |r_{current} - r_{oper}|$
  End While
End Forall

Algorithm 4. The conflict detection procedure

Input: set of track sections $b \in B$, set of blocking times $D_t \in D$
Output: set of conflicts Γ
Initialize $\Gamma := \emptyset$
Forall $b \in B$
  Create a set of trains T_b that use block b and corresponding blocking times D_b
  Sort D_b based on start of blocking times
  Create a pairing list of adjacent trains (t_i, t_{i+1})
    Forall pairs (t_i, t_{i+1})
      If $d_t^e - d_{t+1}^s > 0$
        Insert into Γ a conflict $\gamma = (b, t_1, t_2, \eta)$ between trains (t_i, t_{i+1})
      End If
    End Forall
End Forall

4.3 Blocking times

A blocking time is the time interval that a given section (block section or track detection section) is exclusively allocated to a single train and therefore blocked for other trains. In railways it is not allowed for two trains to be contemporary in the same block section. Blocking times are computed according to the classical blocking time theory (Hansen and Pachl, 2014).

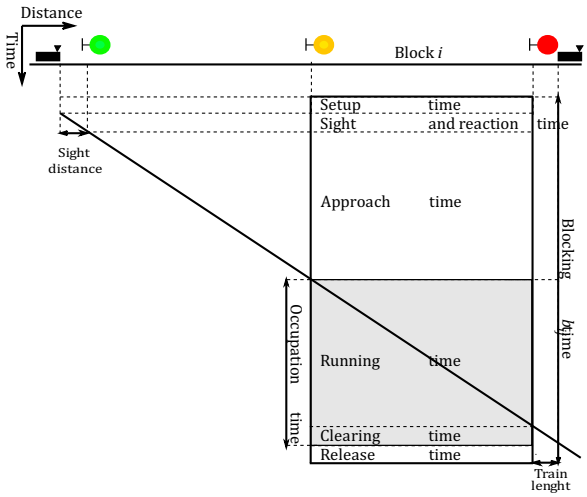

Fig. 4. Blocking time stairway.

As can be seen in Figure 4, the blocking time of a train relative to a given block section is composed of the following components: setup time t_{setup} [s] to set the route for the approaching train; sight distance l_{sight} [m] or sight time t_{sight} [s] of the train driver when approaching the previous block section (approach signal); reaction time $t_{reaction}$ [s] of the driver, usually equal to 1.5–2 seconds; approach time $t_{approach}$ [s] needed by the train to cross the previous block section; running time t_{block} [s] of the train to cross the block section; clearing time t_{clear} [s] needed by the train to clear the block section over its train length; release time $t_{release}$ [s] needed to release the route after the train clearance. After having computed all these terms the blocking time d_{ij} of the train t relative to block i is obtained as

$$d_{ti} = t_{setup,i} + t_{sight,ti} + t_{reaction,ti} + t_{approach,ti} + t_{block,ti} + t_{clear,ti} + t_{release,ti} \quad (1)$$

The input to this function are the infrastructure characteristics and running times of trains. In particular, the operational running times, either from the initial iteration that includes 5% of running time supplements or from macroTT, are used to produce the scheduled blocking time stairways. Note that the signaling system presented in Figure 4 represents a three-aspect two-block signaling system but different systems can be also modeled like four-aspect (U.K. signaling), the Dutch progressive speed signaling system, or the European Train Control System (ETCS) Level 1, 2, and 3.

Blocking times represent the main ingredient for the following functions, so we introduce it formally as $d_{ti} = (d^s_{ti}, d^e_{ti})$, where each blocking time d_{ti} of section i by train t is specified from the start d^s_{ti} to the end d^e_{ti} of the blocking time. Each train t has an attributed list of blocking times $D_t = \{d_{t1}, d_{t2}, \ldots, d_{tn}\}$, where n is the number of track sections along the route ρ^{micro}_t.

4.4 Minimum headway time computation

A minimum headway time is the time separation between two trains at certain positions that enable conflict-free operation of trains (Hansen and Pachl, 2014). The minimum headway is computed based on the blocking times of each train for every macroscopic point, and for each pair of consecutive trains. In particular, for each pair of trains we calculate a set of minimum headways considering all the possible interactions between them such as both trains leaving a station, both trains entering a station, or one entering and the other leaving.

We introduce the computation of the minimum headway at a timetable point $s \in S$. Let B_{ijs} be the set of blocks associated to conflicting routes (inbound or outbound) of train lines i and j in timetable point s, d^e_{il} be the end of blocking time d_{il}, and d^s_{jl} the start of blocking time d_{jl}. Assume that both trains have the same reference event (i.e., departure, arrival, or passing) time at s, for example, equal to 0. Then the minimum headway h_{ijs} from train line i to j in timetable point s is computed as

$$h_{ijs} = \max_{l \in B_{ijs}} \left(d^e_{il} - d^s_{jl} \right) \quad (2)$$

4.5 Conflict detection and resolution (CDR)

The *CDR* model consists of two algorithms: conflict detection (CD) and conflict resolution (CR). The aim of the CDR is to verify the feasibility of the macroscopic timetable and to locally resolve potential conflicts by analyzing the interaction between scheduled trains at the microscopic level. A track conflict occurs when two or more trains are scheduled to the same track section at overlapping periods of time. In other words, a track conflict is identified when the blocking times of two trains overlap fully or partially at a given track section. When a macroscopic timetable is available, we can test its feasibility at microscopic level using the CD procedure. This function takes as input the blocking time stairways produced for the operational running times. If there is an overlap between the blocking times of two different trains, this indicates a track conflict that must be solved. Specifically, track conflicts are solved by shifting trains in time until the blocking times do not overlap anymore. This shift initiates a change in the minimum headway between the trains. After all track conflicts have been

detected, it is necessary to recompute the corresponding minimum headways. These new headways may be given to the macroscopic timetabling model to iteratively adjust the macroscopic timetable until no conflicts are detected anymore. Therefore, conflict-freeness is tested comparing the interaction of scheduled blocking times for each pair of trains, that is, checking the possible blocking time overlaps between those two. The blocking time overlap $c_{ij\varphi}$ from train line i to j at corridor φ is computed similarly to the minimum headway times as

$$c_{ij\varphi} = \max_{l \in B_\varphi} \left(d^e_{il} - d^s_{jl} \right) \tag{3}$$

where B_φ is the set of conflicting blocks at corridor φ. If $c_{ij\varphi} > 0$ then a conflict exists. Usually, a corridor corresponds to a macroscopic arc. In this way, the whole network is analyzed by the conflict detection algorithm.

For the modeling purposes of CD we used a compact but efficient algorithm:

1. Sort the start and end times of the blocking time intervals over shared blocks.
2. Go through the sorted end times and build up the list of conflict pairs by looking at the preceding start time.

Algorithm 4 for CD is presented in the following. First, we initialize the set for observed conflicts Γ. The CD algorithm progresses through the list of track sections and for each $b \in B$ it generates the set D_b that includes blocking times of trains that traverse the bth section. Then, D_b is sorted regarding the start and end times (d^s_{ti}, d^e_{ti}). For each pair of adjacent trains (t_i, t_{i+1}) the procedure checks the relation between the blocking time end of train t_i and blocking time start of train t_{i+1}, $d^e_t - d^s_{t+1}$. If this value is positive then a conflict exists. A conflict $\gamma \in \Gamma$ is described with a pair of conflicting trains t_1 and t_2, the corresponding track section b, and the total time in conflict, that is, the overlap $\eta \leftarrow d^e_t - d^s_{t+1}$; formally, $\gamma = (b, t_1, t_2, \eta)$.

Once all the conflicts have been determined, the CR procedure described in Algorithm 5 resolves existing conflicts between pairs of trains. The CR procedure (1) computes the maximum overlap, (2) determines the associated headway (pair of trains and corresponding macro point) to be updated, and (3) updates the headway time for the maximum overlap. Recall that headways were defined for each macroscopic point, while a conflict may be located somewhere between two macro points. Therefore, we also need to choose the corresponding macro point to assign the updated headway. Algorithm 5 gives a step-by-step description of the CR procedure.

Algorithm 5. The conflict resolution procedure

Input: tracks $a \in A$, conflicts $\gamma \in \Gamma$, $t \in T$, headways $h \in H$
Output: updated headway times H
Forall $a \in A$
  Forall pairs (t_1, t_2)
    Step 1. Initialize a subset, Γ_{sub}, of conflicts that correspond to a triplet (t_1, t_2, a)
    Step 2. Compute the maximum overlap for $\gamma \in \Gamma_{sub}$
    Step 3. Choose macro point s
    Step 4. Update $h_{t_1 t_2 s} \leftarrow h_{t_1 t_2 s} + c_{t_1 t_2 a}$
  End Forall
End Forall

In the first step, the procedure determines the subset of conflicts $\Gamma_{sub} \subset \Gamma$ that corresponds to a pair of conflicting trains (t_1, t_2) at a given arc $a \in A$. Then, the maximum overlap $c_{t_1 t_2 a}$ is determined using (3). Step 3 finds the macroscopic point s for which the headway should be updated. This choice has been made based on the geographical distance between the track section with the maximum overlap and the surrounding macroscopic points, that is, the closer point is selected. Finally, the relative headway $h_{t_1 t_2 s}$ is increased by $c_{t_1 t_2 a}$.

4.6 Capacity assessment

In this section we define the idea of infrastructure capacity assessment. Our approach for capacity assessment is based on the timetable compression method, which is common practice. Timetable compression is the process of shifting train paths to each other as much as possible, bringing them to the (time) distance of minimum headway times. The total time needed for operating such a compressed timetable is the capacity occupation. Capacity assessment consists of determining capacity occupation and capacity occupation rate (share of used capacity expressed in %). We briefly introduce the max-plus automata theory and then apply it to compute capacity occupation. Note that in this section we use a common max-plus algebra notation that may differ from the rest of the article. Our approach overcomes the current limitation of the UIC method and estimates the capacity for the station as a whole, and thus, includes all route dependencies in the station area. The capacity occupation $\mu(\varphi)$ of corridor φ can be obtained by

$$\mu(\varphi) = \sum_{\{(i,j) \in W_\varphi\}} h_{ij\varphi} \tag{4}$$

with W_φ the cyclic pattern of successive train pairs (i, j) in corridor φ, and $h_{ij\varphi}$ the minimum line headway. The minimum line headway is computed similarly to a local

minimum headway but with respect to all blocks on a corridor φ instead of a timetable point s. A corridor may be equal to a station area, an arc or comprise several adjacent arcs, $\varphi = \cup a_i$. We compute the capacity occupation for each corridor $\varphi \in \Phi$, applying an algorithm based on max-plus automata theory.

4.6.1 Basics of max-plus automata theory. Max-plus automata combines elements of the heaps-of-pieces theory and max-plus algebra and was introduced by Gaubert and Mairesse (1999). A max-plus algebra is a semiring over the union of real numbers and $\varepsilon = -\infty$, equipped with the two binary operations maximum ($\oplus$) and addition ($\otimes$). Let $R_{\max}$ be the set of real scalars and $-\infty$, then for $a, b \in R_{\max}$ the operations are defined as

$$a \oplus b = \max(a, b), a \otimes b = a + b$$

The element $\varepsilon = -\infty$ is the neutral element for $\oplus$ and absorbing for $\otimes$. The element $e = 0$ is the neutral element for $\otimes$. Properties of max-plus algebra are similar to conventional algebra. We refer to Goverde (2007) for more details on max-plus algebra with application to railways.

A max-plus automaton H is a triple (Q, R, M), where

- Q is a finite set of tasks, for example, all possible train routes,
- R is a finite set of resources, for example, block section or track detection section,
- M is a morphism $Q^* \to R_{\max}^{|R|x|R|}$ which is uniquely specified by the finite family of $|R| \times |R|$-dimensional matrices $M(l), l \in Q$. Also, Q^* denotes a set of chosen train (partial) routes over a given corridor from Q, $Q^* \subset Q$.

We define a timetable as an ordered sequence of tasks, $w = l_1 \ldots l_n$. Therefore,

$$M(w) = M(l_1 \ldots l_n) = M(l_1) \otimes \ldots \otimes M(l_n)$$

A task is called an elementary task if R-dimensional row vectors $s(l)$ and $f(l)$ exist such that $s(l) \leq f(l)$ and

$$M_{ij}(l) = \begin{cases} e, & \text{if } i = j, i \neq R(l) \\ f_j(l) - s_i(l), & \text{if } i, j \in R(l) \\ \varepsilon, & \text{otherwise} \end{cases} \quad (5)$$

Variables $s(l)$ and $f(l)$ represent the start and end time of task l, respectively. In the railway terms, task l is a (partial) route of a train line, while $s(l)$ and $f(l)$ correspond to occupation and release times of the ith block, d_i^s and d_i^e, respectively.

The *upper contour* $x(w)$ of a schedule w is defined as

$$x(w) = M(w) \otimes x(e)$$

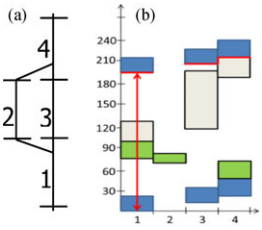

Fig. 5. (a) Example infrastructure and (b) capacity occupation for schedule *abc*.

where $x(e)$ is an R-dimensional vector corresponding to an empty schedule. A more extensive description of max-plus automata theory is given by Gaubert and Mairesse (1999) and Egmond (2000).

4.6.2 Application of max-plus automata to capacity occupation. The capacity occupation $\mu(w)$ of the schedule w is computed as

$$\mu(w) = \min(x(wa) - (f(a) - s(a))) \quad (6)$$

where schedule wa is a schedule for one cycle w and the first train service a that belongs to the next cycle, and $f(a) - s(a)$ is the blocking time stairway of the repeated train service a over all resources. This formulation corresponds to Equation (4). The capacity occupation rate $C(w)$ is defined as $C(w) = \frac{\mu(w)}{P} \times 100[\%]$, where P is the scheduled cycle period.

Let us summarize the capacity occupation model. First, we define a set of arbitrary railway sections ϕ. A section $\varphi \in \phi$ may represent a corridor or a station (i.e., macroscopic timetable point). A corridor is bounded by a pair of macroscopic timetable points, for example, $\varphi = (s_1, s_2, \ldots s_n)$. A station is treated similarly by accepting $\varphi = s$. Then, we determine a subset Q^* of train routes that are selected for train lines over section φ. Finally, the model computes the capacity occupation for each $\varphi \in \phi$ by using (6) and is represented with $\mu(\varphi)$.

4.6.3 Numerical example. Let us consider the following example for computing the capacity occupation in a station. Consider three trains a, b, c, schedule $w = abc$ and resources $r = 1, \ldots, 4$, as in Figure 5a. Train route a uses resources $[1, 3, 4]$, b uses $[4, 2, 1]$, and c uses $[1, 3, 4]$. The train blocking times are given as

Route	$s(r)$	$f(r)$
a	$[0, \varepsilon, 15, 25]$	$[25, \varepsilon, 35, 50]$
b	$[25, 15, \varepsilon, 0]$	$[50, 35, \varepsilon, 25]$
c	$[0, \varepsilon, 20, 90]$	$[30, \varepsilon, 100, 120]$

Note that ε represents an unused resource. The corresponding matrices M for routes a, b, and c are defined

using Equation (5) as follows:

$$M(a) = \begin{bmatrix} 25 & \varepsilon & 35 & 50 \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 10 & \varepsilon & 20 & 35 \\ 0 & \varepsilon & 10 & 25 \end{bmatrix}$$

$$M(b) = \begin{bmatrix} 25 & 35 & \varepsilon & 0 \\ 35 & 20 & \varepsilon & 10 \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 50 & 35 & \varepsilon & 25 \end{bmatrix}$$

$$M(c) = \begin{bmatrix} 30 & \varepsilon & 100 & 120 \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 10 & \varepsilon & 80 & 100 \\ -60 & \varepsilon & 10 & 30 \end{bmatrix}$$

The matrix M for schedule ab is computed as

$$M\ (ab) = M(a) \otimes M\ (b) = \begin{bmatrix} 100 & 85 & 35 & 75 \\ 35 & 20 & \varepsilon & 10 \\ 85 & 70 & 20 & 35 \\ 75 & 60 & 10 & 25 \end{bmatrix}$$

Similarly, train c is added to the schedule in the same manner, that is, $M\ (abc) = M(ab) \otimes M(c)$. The upper contour of the schedule $abca$ is then computed as

$$x\ (abca) = M(abca) \otimes x\ (e) = \begin{bmatrix} 220 \\ 85 \\ 230 \\ 245 \end{bmatrix}$$

The capacity occupation for the scheduled services abc is then computed as

$$\mu\ (abc) = \min\left(x(abca) - \left(f(a) - s(a)\right)\right)$$

$$= \min\left(\begin{bmatrix} 220 \\ 85 \\ 230 \\ 245 \end{bmatrix} - \begin{bmatrix} 25 \\ \varepsilon \\ 20 \\ 25 \end{bmatrix}\right) = 195$$

Note that $85 - (-\infty) = +\infty$. If the cycle period equals $P = 600$seconds, then the capacity occupation rate is

$$C\ (abc) = \frac{\mu(abc)}{P}$$

$$cdot100\,[\%] = \frac{195}{600} \cdot 100\,[\%]$$

$$= 32.5\,[\%]$$

Figure 5b provides a graphical representation of the compressed schedule $w = abc$. The colored blocks represent the train occupation of the infrastructure, with one train movement depicted by the same color. Note that train a is added twice to determine the earliest possible departure of a train from the following period. The red line represents the capacity occupation for schedule w, $C(w)$. The white space (between the x-axis and red line) depicts unused capacity which might be used to add extra trains.

Fig. 6. Case study infrastructure with macroscopic (circles) and microscopic (squares) timetable points.

5 CASE STUDY

In the case study we focus on two elements. First, we show the applicability of each function within the microscopic model. Besides that, we demonstrate the developed timetabling framework with all functionalities of the microscopic module applied to a real railway network. We apply the macroscopic model from Bešinović et al. (2016). However, any other macroscopic model could be used (e.g., Sparing and Goverde, 2013).

We consider a real-life instance for train line services on the 80-km long corridor Utrecht (Ut)–Den Bosch (Ht)–Eindhoven (Ehv) (Figure 6), a highly utilized part of the railway network in the central Netherlands. The values present the number of tracks in stations or junctions and lines between depict the number of tracks between two timetable points. The microscopic infrastructure includes various topology—double, triple, and quadruple tracks. The microscopic graph M for the considered corridor includes around 1,000 nodes and 1,500 microscopic arcs considering infrastructure details like location of signals, switches, train detection points, the speed limits, slope gradients, and curves. For running time computations, a detailed train dynamics have been modeled. The network included 13 microscopic timetable points such as stations, stops, junctions, and bridges.

The original timetable on this network is periodic with half an hour pattern composed of 20 train lines, of which 12 are intercity (IC) and 8 are regional trains. Train lines originate and terminate at different stations along the corridor and have different stopping patterns. Regional trains stop at all stations, while ICs stop at limited stations.

5.1 Functionality of the microscopic model

We start by computing the minimum running times and the corresponding headway times, constructing the macroscopic network and aggregating the process times

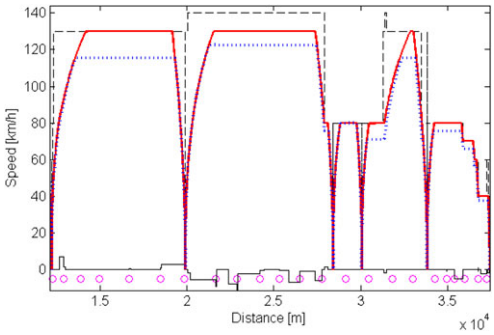

Fig. 7. Train trajectories for minimum running time (red solid line) and scheduled time supplements (blue dotted line). The maximum speed of the train is 130 km/h.

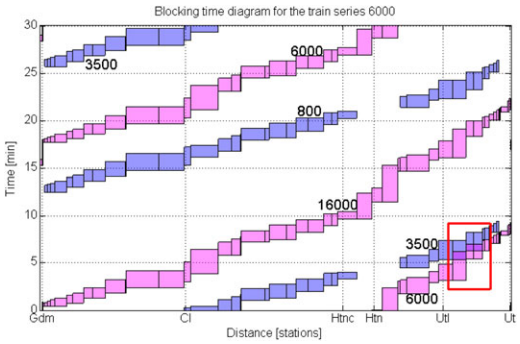

Fig. 8. Blocking time diagram the corridor Gdm–Ut.

Table 1
Capacity occupation at corridors

Corridor	*Time (s)*	*Rate (%)*	*No. of resources*
Ut–Ht	1,892	52.6	110
Ht–Ut	1,924	53.4	104
Ehv–Ht	1,320	36.7	90
Ht–Ehv	1,372	38.1	91

(Algorithm 1). Solving the equations for running time is performed over distance with computational accuracy set to 10^{-5} m, while $\epsilon_{abs} = 1$ second. The accuracy of other models is 1 second. The average computation (CPU) time for the minimum speed profile for one train line was 1 second, while for the operational speed profile was 4 seconds. Generating macroscopic network resulted in seven macroscopic timetable points (important stations and junctions) and six macroscopic arcs. A total of 1,000 headway times was computed in 8 seconds. In the later iterations, a limited number of headways are usually updated, so the CPU time then is well under 1 second. The CPU times for conflict detection for the whole network and capacity assessment per corridor (or station) are on average 3 and 1 second, respectively. Finally, network transformations, micro to macro and vice versa, take under 1 second as well. For testing purposes, we applied a macroscopic timetable model as in Bešinović et al. (2016) to generate a macroTT. Once a macroTT is obtained, the microscopic models evaluated its feasibility and stability. First, a microTT is generated by identifying the operational train trajectories corresponding to the scheduled running times (Algorithms, 2 and 3). The output of the Algorithm 3 for one train line is illustrated in Figure 7 and depicts the distance-speed diagram for the local train 6000 (blue dotted line) running over the corridor Ht–Ut. Such a trajectory corresponds to the scheduled running time where time supplements are exploited by cruising at a speed lower than the time-optimal speed profile, that is, computed for the minimum running time (red solid line). The circles represent line-side signals, the black solid line indicates gradients, and the black dashed line is the static speed limit.

The newly produced blocking times are used in the CDR model to detect possible conflicts between trains. The corridor included 600 track sections. Figure 8 gives the (partial) output of the blocking time computation for the different train services operating between Gdm and Ut. The diagram shows only the infrastructure that train 6000 uses, to clearly visualize actual conflicts between trains. The red box depicts a conflict of train services 6000 and 3500 between Utrecht Lunetten (Utl) and Ut. The minimum headway $h^{dd}_{6000,3500,Ut}$ between these two trains originally was 150 seconds while the maximum overlap of their conflicting blocking times (three in total) is max(48, 38, 38) = 48 seconds. The track conflict is therefore resolved by shifting the train over an extent equal to the overlap. In this case, the minimum headway increases by 48 seconds, resulting in a new headway time $h^{dd}_{6000,3500,Ut} = 198$ seconds, so that the blocking times are touching but not overlapping. This new headway is sent to the macroscopic model together with the other updated headways and running times, for reproducing a new macroscopic timetable.

The capacity occupation for a given microTT is computed by applying the max-plus automata method. The capacity occupation for all corridors and stations is given in Tables 1 and 2, respectively. In addition, the last column in both tables shows the total number of resources used by all routes, which defines the size of matrix M (cf. Equation 5), and thus the complexity of the computation. We describe here the capacity occupation for station Ht, which consists of six station tracks including four platform tracks. Fourteen trains operate each 30 minutes through Ht, which use in total 69 different infrastructure resources. Figure 9 shows the station layout and the output of the capacity assessment. The

Table 2
Capacity occupation at stations

Station	*Time (s)*	*Rate (%)*	*No. of resources*
Btl	870	24.2	85
Ehv	930	25.8	37
Gdm	954	26.5	48
Ht	1,539	42.8	69
Htn	900	25.0	24
Ut	844	23.4	58
Vga	934	25.9	34

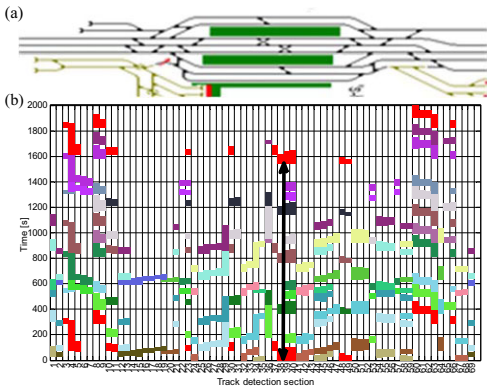

Fig. 9. Station Den Bosch: (a) station layout and (b) capacity occupation.

x-axis reports all the track detection sections belonging to the station. Note that their sequence does not follow a topological order. The *y*-axis denotes time, and the blocks show for each track detection section when they are used by a train service. The different colors of the blocking times correspond to distinct train routes through a station. In red we highlight the first train service of the next timetable period. We found that the capacity occupation time of station Ht is 1,539 seconds (25.6 minutes) and the rate is 42.8% in a timetable period of 60 minutes. This means that the timetable locally contains 2,061 seconds (57.2%) of time allowances. By comparing these values with those suggested by the UIC 406 Code, that is, a minimum of 50%, it is concluded that Ht has an acceptable amount of time allowances, and therefore satisfies the stability norms.

5.2 Testing the developed framework

To show the suitability of the microscopic models within the developed framework, we used the macroscopic timetabling model described in Bešinović et al. (2016). We present the computational results and the computed timetable, including the achieved values for the performance measures, that is, feasibility and stability.

Table 3
Characteristics of the macroscopic timetable after each iteration

Iteration	*No. of conflicting train pairs*	*Overlap time (s)*
1	6	160
2	4	130
3	3	98
4	5	110
5	3	65
6	1	8
7	0	0

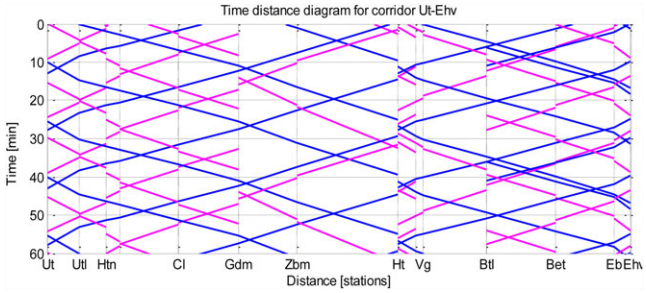

Fig. 10. Time-distance diagram for corridor Ut-Ehv.

Table 3 presents the microscopic conflicts in the macroscopic timetable at the end of each iteration. The number of conflicting train pairs equals the number of headways that has been updated at the microscopic level. Overlap time is the sum over all maximum conflicts between two trains $\sum c_{t_1 t_2 a}$. In the first iteration, there are six conflicts that add up to 160 seconds of overlapping blocking times. In the second iteration, only four conflicts remain with a total overlap time of 130 seconds. In the subsequent iterations, all conflicts are resolved. It can be seen from the table that the approach can solve all conflicts successfully within several iterations, gradually reducing the number and size of total overlaps. However, resolving conflicts in one iteration may produce some new conflicts in the following iteration. But the algorithm converges to a timetable which is completely feasible both macroscopically and microscopically. The observed computation time for obtaining the feasible and stable timetable was about 14 minutes, with on average 2 minutes per iteration.

Figure 10 shows a time–distance diagram of the computed hourly timetable for the corridor Ut–Ehv. The vertical axis shows time in minutes downwards. The horizontal axis shows distance with the station positions indicated. The blue lines are IC trains, the magenta lines are local trains. Note that the sections Btl–Ehv and Htn–Htnc have four tracks where trains may cross each other. Figure 11 shows the corresponding blocking time diagram for the route of intercity train line

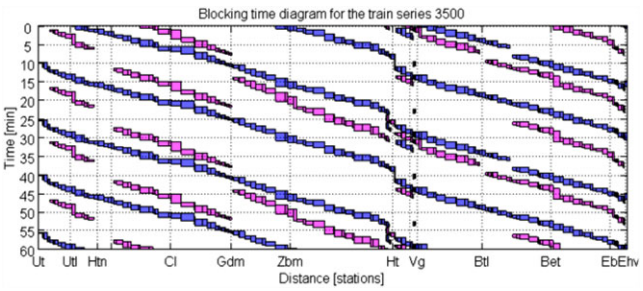

Fig. 11. Blocking time diagram the corridor Ut–Ehv.

3500. Note that only the blocking times are shown for the trains running on the same tracks as train line 3500. The gaps in the blocking time stairways for some trains correspond to running on parallel tracks in stations or the four-track lines between Htn–Htnc and Btl–Ehv. Around Ht also some blocking times are visible corresponding to crossing trains from/to different corridors.

The optimized timetable shows periodic passenger trains with regular 15-minute services of both IC and local trains where two similar train lines follow the same route. Hence, effectively 15-minute train services are realized instead of two separate 30-minute train lines.

The blocking time diagram shows no overlapping blocking times and hence asserts that the timetable is conflict-free. Moreover, the timetable is robust which is illustrated by the buffer times (white space) between the train paths.

Finally, the obtained capacity occupation rates are below the recommended stability values of 65% for mixed traffic corridors in daily periods and 50% for stations defined by the UIC, which were the constraints of the timetabling algorithms. Corridor Ut–Ht is the heaviest used with the capacity occupation rate of 57.8%. Therefore, we may conclude that the produced timetable is also stable.

6 PRACTICAL REFLECTION OF THE DEVELOPED MICROSCOPIC MODEL

The developed framework has been evaluated by experts from the infrastructure managers Network Rail (the United Kingdom) and Trafikverket (Sweden). Here we give a summary regarding the functionality of our microscopic model. The applied time precision of one second is highly appreciated, as it leads to minimizing the unused capacity and unrealizable running times. Also, its relevance is supported by the current efforts in this direction in the United Kingdom. They emphasized the ability of the model to compute highly detailed running and blocking times taking into account all route details at the track section level (speed restrictions, signaling, and gradients). The implementation of the new conflict detection and resolution algorithms that accurately assess the timetable feasibility gives valuable transparency to timetable planners. The importance of capacity occupation and stability norms was also stressed, but they also pointed out the need to standardize and configure the norms to reflect local (national) capacity standards. The overall comment is that "The implemented functionality to timetable planning was reviewed as highly valuable and an advance on current practice." This is also confirmed by the infrastructure manager ProRail and main railway undertaking NS from the Netherlands. The microscopic models are currently applied in a pilot project by ProRail and NS to evaluate the Dutch timetables at the national network level.

7 CONCLUSION

In this article, we have provided a methodology and new microscopic models for supporting the timetable design as well as the network and data transformations to manage communications between microscopic and macroscopic models. The main focus was on the microscopic models for computing reliable running and minimum headway times for the macroscopic model, as well as analyzing the feasibility and stability of the macroscopic timetables at the microscopic level. Operational running times are calculated by integrating the Newton's motion formula and a fast bisection model that introduces cruising phases at lower speeds to cover the supplement times imposed by the timetable. Accurate headway computation is based on the blocking time theory. In this way, we could generate train process times in short time, even for very dense railway traffic. The timetable feasibility was checked by an efficient conflict detection model based on the blocking time theory, which automatically recomputes new minimum headway times if a conflict arises. The capacity assessment is realized by the new application of max-plus automata following the compression method. Our method allowed computing capacity occupation in stations as well as at corridors. If the capacity occupation rate satisfies technical thresholds the timetable is considered as stable.

The microscopic models were also integrated in an innovative timetabling framework to develop timetables that are operationally feasible and stable. The framework is completely general and based on the iterative interaction among macroscopic and microscopic models. Due to its modular development, the macroscopic model can be any optimization model for timetable computation.

We applied the microscopic models at a real Dutch railway network. Small computation times make us confident that the models could also be used on more complex instances, although no microscopic infrastructure data for other networks is available to us yet. In practice, microscopic models could be interfaced with existing timetabling tools to rapidly obtain good quality timetables that are also conflict-free and able to effectively absorb delays. We believe that presented microscopic models have a great potential for improving real-life applications for railway planning.

ACKNOWLEDGMENTS

This research is funded by the European FP7 project Optimal Networks for Train Integration Management across Europe (ON-TIME) under Grant Agreement number FP7- SCP1-GA-2011-285243. We thank the Dutch infrastructure manager ProRail and the Netherlands Railways (NS) for the data provided.

REFERENCES

Albrecht, A. R., Howlett, P. G., Pudney, P. J. & Vu, X. (2013), Energy-efficient train control: from local convexity to global optimization and uniqueness, *Automatica*, **49**(10), 3072–8.

Armstrong, J., Preston, J. & Hood, I. (2015), Evaluating Capacity Utilisation and Its Upper Limits at Railway Nodes, CASPT 2015, Rotterdam, The Netherlands.

Bešinović, N., Goverde, R. M. P., Quaglietta, E. & Roberti, R. (2016), An integrated micro-macro approach to robust railway timetabling, *Transportation Research Part B: Methodological, 87, 14–32*.

Bešinović, N., Quaglietta, E. & Goverde, R. M. P. (2013), A simulation-based optimization approach for the calibration of dynamic train speed profiles, *Journal of Rail Transport Planning and Management*, **3**(4), 126–36.

Bešinović, N., Quaglietta, E. & Goverde, R. M. P. (2014), Supporting tools for automated timetable planning, in C. A. Brebbia, N. Tomii, P. Tzieropoulos, and J. M. Mera (eds.), *Computers in Railways XIV*, WIT Press, Southampton, pp. 565–76.

Brünger, O. & Dahlhaus, E. (2014), Running time estimation, in I. A. Hansen and J. Pachl (eds.), *Railway Timetabling and Operations*, Eurailpress, Hamburg.

Butcher, J. C. (2013), *Numerical Methods for Ordinary Differential Equations*, Wiley, London.

Cacchiani, V., Huisman, D., Kidd, M., Kroon, L., Toth, P., Veelenturf, L. & Wagenaar, J. (2014), An overview of recovery models and algorithms for real-time railway rescheduling, *Transportation Research Part B*, **63**, 15–37.

Cacchiani, V. & Toth, P. (2012), Nominal and robust train timetabling problems, *European Journal of Operational Research*, **219**(3), 727–37.

Caimi, G., Fuchsberger, M., Laumanns, M. & Schüpbach, K. (2011), A multi-level framework for generating train schedules in highly utilised networks, *Public Transport*, **3**(1), 3–24.

Castillo, E., Gallego, I., Sánchez-Cambronero, S., Menéndez, J. M., Rivas, A., Nogal, M. & Grande, Z. (2015), An alternate double-single track proposal for high speed peripheral railway lines, *Computer-Aided Civil and Infrastructure Engineering*, **30**, 181–201.

D'Ariano, A., Pranzo, M. & Hansen, I. A. (2007), Conflict resolution and train speed coordination for solving real-time timetable perturbations, *IEEE Transactions on Intelligent Transportation Systems*, **8**(2), 208–22.

De Fabris, S., Longo, G., Medeossi, G. & Pesenti, R. (2013), Application and validation of a timetabling algorithm to a large Italian network, *RailCopenhagen 2013*, Copenhagen, May 13–15, 2013.

Egmond, R. J. (2000), An algebraic approach for scheduling train movements, CASPT 2000, Berlin, June 21–23, 2000.

Gaubert, S. & Mairesse, J. (1999), Modeling and analysis of timed Petri nets using heaps of pieces, *IEEE Transactions on Automatic Control*, **44**(4), 683–97.

Gille, A., Klemenz, M. & Siefer, T. (2008), Applying multiscaling analysis to detect capacity resources in railway networks, *Computers in Railways XI*, WIT Press, Southampton, pp. 595–604.

Goverde, R. M. P. (2007), Railway timetable stability analysis using max-plus system theory, *Transportation Research Part B*, **41**(2), 179–201.

Goverde, R. M. P., Bešinović, N., Binder, A., Cacchiani, V., Quaglietta, E., Roberti, R. & Toth, P. (2016), A three-level framework for performance-based railway timetabling, *Transportation Research Part C*: Emerging Technologies, 67, 62–83.

Goverde, R. M. P. & Hansen, I. A. (2013), Performance indicators for railway timetables, in *Proceedings of the IEEE International Conference on Intelligent Rail Transportation (ICIRT 2013)*, Beijing, August 30 September 1, 2013, pp. 301–6.

Hansen, I. A. & Pachl, J. (2014), *Railway Timetabling and Operations*, Eurailpress, Hamburg.

Kroon, L., Huisman, D., Abbink, E., Fioole, P. J., Fischetti, M., Maroti, G., Schrijver, A., Steenbeek, A. & Ybema, R. (2009), The new Dutch timetable: the OR revolution, *Interfaces*, **39**(1), 6–17.

Landex, A. & Jensen, L. W. (2013), Measures for track complexity and robustness of operation at stations, *Journal of Rail Transport Planning & Management*, **3**(1–2), 22–35.

Lindner, T. (2011), Applicability of the analytical UIC Code 406 compression method for evaluating line and station capacity, *Journal of Rail Transport Planning and Management*, **1**(1), 49–57.

Nash, A. & Huerlimann, D. (2004), Railroad simulation using OpenTrack, in J. Allan, C. A. Brebbia, R. J. Hill, G. Sciutto, and S. Sone (eds.), *Computers in Railways IX*, WIT Press, Southampton, pp. 45–54.

ON-TIME (2016), Available at: www.ontime-project.eu, accessed January 2016.

Peng, F., Kang, S., Li, X., Ouyang, Y., Somani, K. & Acharya, D. (2011), A heuristic approach to the railroad track maintenance scheduling problem, *Computer-Aided Civil and Infrastructure Engineering*, **26**, 129–45.

Quaglietta, E. (2014), A simulation-based approach for the optimal design of signalling block layout in railway networks, *Simulation Modelling Practice and Theory*, **46**, 4–24.

RailML (2015), Available at: http://www.railml.org, accessed June 2015.

Scheepmaker, G. M. & Goverde, R. M. P. (2015), The interplay between energy-efficient train control and scheduled running time supplements, *Journal of Rail Transport Planning & Management*, 5(4), 225–39, http://dx.doi.org/10.1016/j.jrtpm.2015.10.003.

Schlechte, T., Borndorfer, R., Erol, B., Graffagnino, T. & Swarat, E. (2011), Micro-macro transformation of railway networks, *Journal of Rail Transport Planning and Management*, **1**, 38–48.

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*, **84**, 124–56.

Siefer, T. & Radtke, A. (2006), Evaluation of delay propagation, in *Proceedings of VIIth World Congress on Railway Research*, Montreal.

Tsiflakos, K. & Owen, D. B. (1993), A decision support tool for the railway industry based on computer graphics and intuitive modelling techniques, *Computer-Aided Civil and Infrastructure Engineering*, **8**, 105–18.

UIC (2013), *Code 406: Capacity*, International Union of Railways, Paris, 2nd edn.

Xie, W., Ouyang, Y. & Somani, K. (2014), Optimizing location and capacity for multiple types of locomotive maintenance shops, *Computer-Aided Civil and Infrastructure Engineering*, 31(3), 163–l75, http://dx.doi.org/10.1111/mice.12114/.