Journal of Rail Transport Planning & Management 3 (2013) 68–77

Contents lists available at ScienceDirect

Journal of Rail Transport Planning & Management

journal homepage: www.elsevier.com/locate/jrtpm

Robust railway station planning: An interaction between routing, timetabling and platforming

Thijs Dewilde [a,*], Peter Sels [a,b,c], Dirk Cattrysse [a], Pieter Vansteenwegen [a]

[a] KU Leuven, University of Leuven, Centre for Industrial Management, Traffic & Infrastructure, Celestijnenlaan 300a, 3001 Leuven, Belgium
[b] Infrabel, Department of Network Access, Frankrijkstraat 91, 1070 Brussels, Belgium
[c] Logically Yours BVBA, Plankenbergstraat 112 bus L7, 2100 Deurne, Belgium

ARTICLE INFO

Article history:
Received 23 August 2013
Revised 4 November 2013
Accepted 10 November 2013
Available online 5 December 2013

Keywords:
Timetable robustness
Railway timetabling
Train routing
Railway stations

ABSTRACT

In this paper, we consider complex, busy stations whose limited capacity is one of the main reasons of delay propagation. Our goal is to improve, during the planning phase, the robustness of a complex station by fully exploiting the potential of the available capacity. The main feature of our approach is the interaction between routing decisions, timetabling and platform assignments. By altering one of these, slack can be created to allow improvements by the others as well. An objective function that maximizes the time span between any two trains is defined and the timing of the trains and the way how trains are routed to the platforms are optimized in the scope of this objective. By maximizing the spread of the trains, potential conflicts are avoided which is beneficial for – but not identical to – robustness. Using our approach, the robustness in the station zone of Brussels, Belgium's main railway bottleneck, can be improved by 8%. Next to that, the amount of knock-on delay arising due to conflicts within this area can be halved. This performance of our approach is confirmed by a second case study based on the station zone of Antwerp.

1. Introduction

When two trains come too close together on a shared infrastructure part, a conflict occurs with delay propagation as a consequence. Increasing the time span between these two trains works beneficial for the robustness of the system. Doing this for all pairs of trains, however, is impossible when dealing with highly used railway bottlenecks like complex, busy stations. In this paper, we improve the robustness of a railway system in bottlenecks and their neighborhood by focussing on the potential of the available capacity. Routing actions, timetable changes and platform deviations to improve the spreading of the trains are considered. Also the interaction between any of these changes is studied. Altering one item may require several other moves but can also create new possibilities for further improvement. Our goal is to improve the railway robustness which is a broad concept tackled by many authors. In Dewilde et al. (2011), our vision on robustness is introduced and motivated: "*A railway timetable that is robust minimizes the real total weighted travel time of the passengers, in case of frequently occurring small delays.*" Thus, focussing on the passengers, we want to decrease the travel times in practice, which include delays and missed transfers.

This paper results form a cooperation between the University of Leuven and the Belgian railway infrastructure manager Infrabel. Daily, the latter records a large amount of delay propagation in the zone of Brussels, the center of the Belgian railway network. This resulted in the following research question: *How can we improve the robustness in and around large railway stations?*

The idea behind this study and the first obtained results can be found in Dewilde et al. (2013). In the current paper, a new and significantly improved version of our approach is presented. Not only did we increase the level of detail to make the simulation (and the timetable) more realistic, we also applied our procedure to a new and intrinsic different case study. For this new case study, our findings are confirmed by the Belgian infrastructure manager when they simulated our results using their own commercial (microscopic) simulation tool LUKS (VIA Consulting & Development GmbH, 2013). In the algorithm itself, two new ways to improve the systems' robustness are introduced: changes in the order of trains and in the platform allocations are now being considered. Applying any of these changes can easily result in conflicts, i.e., overlapping blocking times. As a consequence, an extra loop that can cope with these infeasibilities is successfully introduced in the algorithm. It tries to solve the conflicts and looks further ahead to evaluate the potential of the initial change.

* Corresponding author. Tel.: +32 16 32 25 67.
E-mail address: Thijs.Dewilde@cib.kuleuven.be (T. Dewilde).

2210-9706/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.jrtpm.2013.11.002

1.1. Approach

In order to improve the robustness in a complex station zone, we focus on three aspects of the planning: the routing of the trains through the station zone, the timetable at the stations within this zone and the platform assignments. Clearly, there is an interaction between these three aspects. A modification of one aspect affects the set of possible achievable improvements of another one. For instance, a time shift of a train affects the achievable improvements of rerouting this train. Changing the platform assignment for a train without taking other actions, may not be beneficial for the system or, even worse, create conflicts which result in extra delay. The full strength of a (platform) change can only be explored whenever other, related timetable (and routing) adjustments are considered simultaneously. This is what we call the potential of a change. In this paper, this potential and the interaction between the different changes are investigated thoroughly.

In general, the construction of a railway timetable is done top-down. The schedule is created on a macroscopic level and microscopic checks are used to evaluate and repair the feasibility at individual stations (Huisman et al., 2005). The other way around, first building a plan for each station individually and combining all these (small) plans to one (large) timetable, the bottom-up approach, can be very cumbersome and non-efficient. Something in between this top-down and bottom-up approach can be practical especially in the case of star shaped networks. Using the nomenclature of the Theory of Constraints (drum-buffer-rope) (Goldratt, 1984), one may say that the center functions as drum and the lines towards the center can act as buffer. Adapting the other stations in the network to the new planning and hitting the drum at the optimal beat of the center, can improve the performance of the overall system. This is the idea behind our study.

In order to construct a robust timetable, we propose to develop a timetable for the center of the network. Next, this local timetable can be used as starting point to construct a timetable for the whole Belgian network. To extend the local timetable to a feasible global schedule, one can use a top-down approach with feedback loops or the technique of goal programming (Vansteenwegen and Van Oudheusden, 2006, 2007). Similar as in the top-down approach where some rough capacity constraints are considered to model the available capacity at stations and a feedback loop is added to perform microscopic checks, we start from an initially feasible schedule and have added time-window constraints for each train. When creating a national timetable, the timetable's attractiveness, or market suitability, as it is called by Lindner (2011), needs to be considered. In Lindner (2011), several examples to indicate shortcomings in the UIC Code 406 method to measure capacity (UIC, 2004) are presented. These examples point at some drawbacks in the way capacity is measured and at the consequences capacity restrictions can have on the attractiveness of the timetable, e.g., when trains on a single track are grouped per direction. However, using the method of retiming and reflowing (Sels et al., 2011a), one should be able to detect and solve such cases. The same holds for the planning of transfers in the new timetable. Based on a reflow action, important transfers can be detected and a retiming action can improve or restore the transfers if needed.

Thus, although changing the event times of some trains in a station may lead to conflicts at other stations, we do not sort this out in this paper. We are convinced that these conflicts are not impossible to solve and that a good schedule for the central bottleneck takes precedence. Nevertheless, this should be kept in mind and interpreting and extrapolating the obtained results should be done carefully.

Using our algorithm, the impact on the performance of infrastructure changes, like increasing the number of platforms in one station, can be evaluated as is done in Dewilde et al. (2013). In this paper, we consider the infrastructure as unchangeable. Although adding extra routing possibilities could avoid intersecting paths, which is one of the main reasons of delay propagation in the station zone of Brussels and thus would improve the routing solutions considerably, we only focus on the potential of the planning itself to improve the robustness of the system.

1.2. Definitions

In this paper, we use the term *station zone* to indicate the network that consists of a set of stations with their platforms and the tracks and switches that connect the incoming lines with the platforms and the outgoing lines. An overview of the station zone of Antwerp is given in Fig. 1. On the left hand side of this figure, an overview of the zone is shown and more details about the network are presented at the right hand side. Note that tunnels are represented by dashed lines.

The *(minimal) time span* between two trains is defined as the smallest delay that causes these trains to conflict in the station zone. By comparing the blocking times of the trains' common sections under undisturbed circumstances, the minimal time span between two trains can be calculated. When the minimal time span is nonnegative for all pairs of trains, the train schedule is considered *feasible* or *conflict-free*.

In this paper, three aspects of the planning within a station zone are considered. (i) The *train routing problem* is about finding a path for each train through the considered network, in our case a station zone. More precisely, given a set T of trains and the set R of all possible routes through the complete zone, a solution of the train routing problem is an allocation of exactly one route $r \in R$ to each train $t \in T$ in a conflict-free way. Solving the train routing problem, we assume that the arrival and departure time of each train and the assigned platforms are fixed. (ii) In what we call the *timetabling problem*, we keep the routing solution and the assigned platforms but allow variations in arrival and departure times. (iii) When focussing on the *platform assignment*, we reallocate a particular train in a particular station to a new platform. Consequently, a new route for this train is needed since a route is linked to a unique platform. In order to avoid conflicts and explore the *potential* of this reallocation, timetable changes are also allowed after a platform reallocation is made.

The time needed to make a certain journey according to the timetable, so under ideal circumstances, is what we call the *planned travel time*. In reality, disturbances will occur such that the time spent while traveling will be different from the planned travel time. Therefore, we define the *(average) real (total) travel time* as the total travel time that is needed to make the trip under these disturbed circumstances, so from the moment of planned departure until the effective arrival at the destination. Note that robustness is about the impact of disturbances on the system. For a passenger railway system, this comes down to bringing the passengers as fast as possible to their destination. In terms of the concepts that are introduced above, robustness is about minimizing the real travel time of the passengers. In Section 3, we use this to come up with a practically usable definition of robustness. For completeness, we also explain what we mean with the *nominal travel time*. The nominal travel time is the time needed to make a journey when the duration of each action (including transfers) equals the minimal necessary duration.

In the next section, an overview of the related literature is given. In Section 3, our vision on robustness is presented. The used objective function and the three modules that form our algorithm are tackled individually in Section 4. In Sections 5 and 6, the simulation model that is used to evaluate new found solutions and the case studies where this simulation model is applied on are

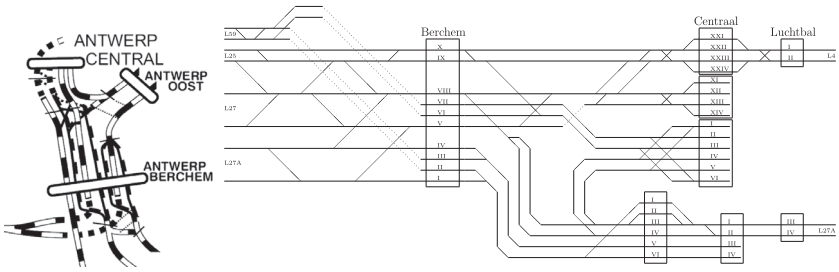

Fig. 1. An overview of the station zone of Antwerp.

introduced. After that, we present the results from our study and in the last section conclusions are drawn.

2. Literature

2.1. Train routing and platforming

In our algorithm, we consider the routing of the trains through the station zone twice. First, we solve the train routing problem for a given platform assignment and second, we investigate the impact of platform changes.

An extensive overview of the train routing problem at station level can be found in the survey paper of Lusby et al. (2011). In Dewilde et al. (2013), we elaborate on some related papers (Caimi et al., 2005, 2009, 2011; Zwaneveld et al., 1996, 2001) and on the comparison with our approach. The main differences between these papers and our work are the way robustness is tackled and the fact that they allow small timetable deviations to obtain a feasible train routing while we alternate between the routing and timetabling, keeping the train routing problem feasible at all times.

Where for the train routing problem, it is assumed that multiple routes exist to reach a certain platform, in the train platforming problem, one assumes that the switch zones bordering the stations are less complex such that the choice of the platform more or less determines the route to follow (Caprara et al., 2011; Cornelsen and Stefano, 2007). Often, the allocation of trains to the platforms of a complex station is considered for capacity purposes as is done in Sels et al. (2011b). In this paper, however, we start from a feasible platform assignment and try to reallocate trains to another platform to improve the robustness of the system. Doing so, a new route needs to be selected out of a set of candidate routes that lead to and from that platform.

In this paper, we restrict our study to a network consisting of some (main) stations that lie closely together such that disturbances at one station will inevitably cause conflicts at the other ones. Such a *station zone* with multiple stations which are lined up with multiple lines that merge, intersect and diverge, is also considered in Carey and Crawford (2007). However, unlike in our approach, they do not consider the routing through the network as well as the robustness of the schedule.

2.2. Robust timetabling

Robust timetabling is a topic that is covered by a lot of papers. Independent of how long the list of papers about robustness is, the list of interpretations of what timetable robustness is about is nearly as long (Cacchiani and Toth, 2012; Dewilde et al., 2011). An often applied and useful technique is the inclusion of timetable slack. This can be in the form of running and dwell time supplements to avoid or absorb delays or by increasing the buffer times to avoid delay propagation (Kroon et al., 2008a; Salido et al., 2012; Shafia et al., 2012). However, the assessment of the slowed down traffic (increased planned travel time) versus the gain in real travel time by less delay is hardly made. Schöbel and Kratz (2009) and Vansteenwegen and Van Oudheusden (2006, 2007) do consider both aspects. Both try to minimize the chance on a missed transfer and do consider the travel time at the same time. The former uses a bicriteria approach and minimizes both the planned travel time and the possibility of missing a transfer, while the latter computes the chance on a missed transfer in function of the size of the running time supplements and minimizes the real passengers' travel time accordingly. Also in Sels et al. (2013), the real (or expected) passengers' travel time is considered upon optimizing and evaluating a timetable. Another methodology that can be added to the list with papers considering the cost of robustness (here: increased planned travel time), is light robustness (Fischetti and Monaci, 2009). First, an initial timetable is created, e.g., by minimizing the planned travel time, and then a worsening in objective function value is allowed to improve the robustness by, e.g., including running time supplements. The concept of recoverable robustness, which considers recovery measures upon scheduling, has been used for multiple stages within the railway planning process: when creating a timetable (Cicerone et al., 2009; Liebchen et al., 2009), when solving the train platforming problem (Liebchen et al., 2009), and when scheduling the rolling stock (Cacchiani et al., 2008). Although the technique was originally developed to handle large disruptions, it is also used for timetable robustness against small disruptions.

In spite of all the effort put in creating a robust timetable, there will always be disturbances which require real-time interventions to solve conflicts. Then topics like delay management (Dollevoet et al., 2012; Meng and Zhou, 2011) and real-time rescheduling and rerouting (Corman et al., 2009, 2010; D'Ariano et al., 2008; Tamura et al., 2013) become useful.

2.3. Robustness evaluation

As mentioned in the previous section, there are many interpretations of the concept timetable robustness. The same holds for the way robustness is measured. In general, one can evaluate a timetable while optimizing, e.g., by including a robustness function in a mathematical model, or one can obtain a robustness score afterwards, e.g., through simulation. A popular way to include robustness while optimizing is by looking at the minimal time spans as is done in Caimi et al. (2005) and Vromans et al. (2006). With the use of simulation, multiple performance indicators can be evaluated simultaneously. Also more complex quality measures like the passenger disutility can be computed (Kanai et al., 2011; Takeuchi et al., 2007). Another, more theoretical technique to calculate the delay propagation, the percentage of missed connections or the settling time, is through the theory of max-plus algebra (Goverde, 2007). In Tassenoy et al. (2013), a practical approach to compare the robustness of two timetables is presented. Based on a macroscopic and a microscopic simulation, the propagation and

the recovery of delays are analyzed. In this paper, we use an objective function based on the minimal time spans to estimate the robustness during the optimization, see Section 4.1, and afterwards, we measure the real robustness, as is defined in the next section, by running a simulation model.

3. Robustness

To remove the discrepancy between the numerous existing definitions of robustness, we focus on what we think is the core of what robustness is about. According to Snelders et al. (2004), robustness is about offering a good quality of the passengers' service, even under disturbed circumstances. In the case of public transport, we are convinced that the level of passengers' service should be measured by the real total travel time of all the passengers. That is why we define railway robustness as follows: *A railway system that is robust minimizes the real total weighted travel time of the passengers, in case of frequently occurring, small disturbances.*

Considering the real total travel time of the passengers, as defined in Section 1.2, allows one to assess the use of timetable slack and account for important transfers. The *real total weighted travel time of the passengers* is measured by adding weights representing the passengers' perception of unnecessary travel time extensions like delays, waiting on a platform for the arrival of a connecting train, or consuming a running time supplement while dwelling (Dewilde et al., 2011, 2013). To obtain a fair comparison between different systems, which can have different nominal travel times due to, e.g., more or less stops, we normalize this real total weighted travel time by the nominal travel time of the corresponding system. Thus, for a system X, the *weighted travel time extension* (WTTE) is measured as follows:

$$\mathrm{WTTE}_X = \frac{(\text{real total weighted travel time})_X - (\text{nominal travel time})_X}{(\text{nominal travel time})_X}.$$

Now, the robustness of system X can be compared to the robustness of any reference system (REF) by using

$$1 + \frac{\mathrm{WTTE}_{REF} - \mathrm{WTTE}_X}{\mathrm{WTTE}_{REF}}. \tag{1}$$

A value of 1.10 or 110% means that system X is 10% more robust than the reference system.

4. Methodology

In this section, the outline of our algorithm is presented. A reference timetable is used as initial solution. This is a timetable that is used for many years in practice and is improved step by step in the course of the years. Also the routings and platform allocations are based on real life data corresponding to this timetable. The algorithm starts with calling the routing module, followed by the timetabling module. If the timetabling module did not find any improvement, the platforming module is executed. Otherwise, the routing module is resolved right after the timetabling module has finished. An overview of this framework is given in Algorithm 1.

Algorithm 1. Framework of our algorithm

input: infrastructure data and reference timetable
while number of consecutive non-improving iterations ≤ 5 **do**
  solve train routing problem
  apply the tabu search (timetabling module)
  if timetabling did not found improvements **do** start the platforming module

4.1. Objective function

Including robustness in the objective function would result in a complicated model since delay propagation computations are required to find the real total travel time. That is why we apply maximizing the spread of trains (time spans), a commonly used technique (Caimi et al., 2005; Kroon et al., 2008b; Salido et al., 2012), to indirectly improve the robustness. The impact on the robustness is measured afterwards.

Let T be the set of trains and let $B_{t,t'}$ (from Buffer time) be the minimal time span between trains t and t'. Define the cost

$$C_{t,t'} = \begin{cases} 15 & \text{if } B_{t,t'} = 0 \text{ (conflicting)}, \\ 1/B_{t,t'} & \text{if } B_{t,t'} < B^{\max} \text{ minutes}, \\ 0 & \text{otherwise}, \end{cases} \tag{2}$$

with $B^{\max}$ a model parameter denoting the shortest duration of a time span that is considered insensitive to conflicts. The cost C corresponding to a conflict is set equal to 15 because we use a precision of 0.1 minutes for computing B causing $C_{t,t'}$ to be smaller than 15 if $B_{t,t'} > 0$. C is used as cost in our *spreading objective function*:

$$\min \sum C_{t,t'}. \tag{3}$$

Using this function has several advantages. First of all, due to the usage of the reciprocals in (2), smaller time spans correspond to higher costs and there is a decreasing marginal effect of increasing a time span. Second, since the sum ranges over all time spans, one can assess the impact of each timetable change that causes increased and decreased time spans, and third, it provides a fast and easy way to evaluate a timetable.

Our spreading objective function is similar to Vromans' SSHR measure (Sum of the Shortest Headway Reciprocals) (Vromans et al., 2006). Using the sum of the reciprocals, the impact of heterogeneity on the reliability of a railway system can be measured since time spans at the beginning and end of a track section will differ due to speed differences. Where the applicability of the SSHR measure in Vromans et al. (2006) is restricted to track sections between stations, we consider the stations itself and use (3) as objective function instead of a reliability measure only.

4.2. Routing module

In the routing module, the train routing problem is solved to optimality using a mixed integer linear program. The objective function coefficients are computed using (2) prior to solving the mathematical model. Since the full details of this model are already presented in Dewilde et al. (2013), we do not give any further details here.

4.3. Timetabling module

When the train routing problem is solved, each train has a unique route which is chosen such that the total spread of the trains is as large as possible for the given event times. Within the timetabling module, one tries to improve the spreading objective value (3) even more by changing the arrival and departure times of some trains. Using a mathematical model requires the inclusion of the minimum time spans for all possible combinations of event times which would result in a large and complex model. As an alternative, we chose a tabu search heuristic to improve the timetable. As a consequence, the optimal timetable (or the optimal spreading value) as well as the optimality gap are not known.

Since we assume periodic timetables, we consider only one or two periods (hours) in our algorithm. We do not allow trains to be shifted to another period. That is why we imply a time-window for each train individually. Where needed, this allows us to restrict

the deviations in comparison with the original schedule or to keep two trains close together for important transfers.

The tabu search heuristic uses three neighborhood structures: shift, combined shift and order swap. When any of these neighborhood structures is called, it considers all candidate moves and returns the best one (steepest descent). In a first step, each of the neighborhood structures is explained and afterwards the details of the tabu search algorithm are given. In the following, we assume that the move under consideration considers an increment of $B_{t,t'}$, the time span between train t and t', with t the predecessor of t', notation $t \prec t'$. The size of a timetable deviation (shift) is indicated with δ and means that all arrival and departure times of the corresponding train are changed with δ.

4.3.1. Shift

A shift move, the most basic one, consists of the shift of one or more trains in time with $\delta_t = -\delta_{\min}, -\delta_{\min}+1, \ldots, -1$ or $\delta_{t'} = 1, 2, \ldots, \delta^{\max}$. $\delta_{\min}$ and $\delta^{\max}$ are model parameters whose values are given in Table 1. Remember that we assume that $t \prec t'$ such that these shifts increase $B_{t,t'}$ by advancing (postponing) train t (t') by δ_t ($\delta_{t'}$). If the shift (t, δ_t) (analogous for $(t', \delta_{t'})$ or any other train that gets shifted) would incur a conflict with another train, say $\tilde{t}$, or make $B_{\tilde{t},t} < B_{t,t'}, \tilde{t}$ also gets shifted with δ_t. Note that we only allow one of the two trains (t or t') to be shifted.

4.3.2. Combined shift

Where the shift operator only allows a shift in one direction with size δ, combined shift considers the combination of multiple shifts. It starts with a simple shift, then the candidate list is updated and a new shift is done. In total $\#\text{shift}^{\max}$ shift moves are allowed simultaneously. At the end, the best of all candidates returned by the shift operator, is selected. This is not necessarily the one in which $\#\text{shift}^{\max}$ individual shifts are made.

Some words about the update of the candidate list. Each time a neighborhood structure is called, only the elements of a restricted candidate list (RCL) are considered. This is done to reduce the computation time. In general, the RCL consists of the pairs of trains whose time span belongs to a prescribed interval (which is updated within the tabu search framework, see below). This way, two trains with a time span larger than $B^{\max}$ will not be considered for a shift.

After a new solution is obtained within the combined shift framework, the RCL gets updated such that the set of candidate moves becomes related to one of the previous moves. This way, we avoid shifting simultaneously two trains which are far apart in time. More precisely, let M be the set of simultaneously shifted trains. Then the pair of trains (t, t') will be added to the RCL if $t \in M, t' \in M$ or if there exists a $\tilde{t} \in M$ such that $B_{\tilde{t},t}$ or $B_{t',\tilde{t}}$ is smaller than the current upper bound of the interval that determines the RCL within the tabu search, see (4) below. The latter allows to explore the potential of a prior shift; a first change can have created the possibility to increase the time span of another pair of trains.

Table 1
Parameter values.

Parameter	Value	Parameter	Value
$B^{\max}$ (min)	15	$\vartheta^{\max}$ (min)	5
$\delta_{\min}$ (min)	−5	B_{margin} (min)	2
$\delta^{\max}$ (min)	5	Shift-tabu tenure	5
$\#shift^{\max}$	10	Swap-tabu tenure	5
δ_{step} (min)	5	Stop criterium 1	∞ (30)
$\varepsilon_{\text{step}}$ (min)	0.5	Stop criterium 2	200 (5)

4.3.3. Order swap

The neighborhood structures above try to increase $B_{t,t'}$ by shifting t forward and/or postponing the events of t'. The idea behind an order swap move is to consider the possibilities of changing the order of two trains at their critical point, i.e., the place where the time span equals the minimal time span. Thus instead of $t \prec t'$, the shifts δ_t and $\delta_{t'}$ are such that $t + \delta_t \succ t' + \delta_{t'}$. Note that now $\delta_t \neq 0$ and $\delta_{t'} \neq 0$ is allowed. The outline of this move, which is new in comparison with Dewilde et al. (2013), is summarized in Algorithm 2.

Algorithm 2. Details for the order swap

input: RCL and tabu list
for all $(t, t') \in$ RCL **do**
  (i) determine $\underline{t}$ and $\bar{t}$
  for $(\underline{t}, t), (t, t')$ or $(t', \bar{t})$ **do**
    (ii) determine δ^{tot}
    (iii) **for** $\delta_t = 0..(\delta_{\text{step}})..\delta^{\text{tot}}$ and $\delta_{t'} = \delta_t - \delta^{\text{tot}}$ **do**
      (iv) shift t and t' or the whole extended set
      (v) perform the internal timetabling module and evaluate
return best found

An order swap consists of five steps.

(i) The first step is the selection of the swap pair. If (t, t') is the RCL-candidate, denote with $\underline{t}$ ($\bar{t}$) the train closest to t (t') such that $\underline{t} \prec t$ ($t' \prec \bar{t}$). Then, one by one, the order swap of the pairs $(\underline{t}, t)$, (t, t') and $(t', \bar{t})$ is considered. All three go through steps (ii)-(v). This is done because of the observation that sometimes $B_{t,t'}$ cannot be increased without involving the other trains surrounding these. For the ease of notation, we use (t, t') as swapping pair in the remainder of this paragraph.

(ii) In the second step, the size of the total shift, δ^{tot}, is determined. δ^{tot} is the smallest value for which $t + \delta^{\text{tot}} \succ t'$ and $B_{t+\delta^{\text{tot}},t'} \geqslant B_{t,t'}$.

(iii) The third step is about determining the values of δ_t and $\delta_{t'}$. This is a loop containing step (iv) and (v). It starts with $\delta_t = 0$ and $\delta_{t'} = -\delta^{\text{tot}}$ and adapts them in each iteration with δ_{step} such that $\delta_t + |\delta_{t'}| = \delta^{\text{tot}}$ at all times.

(iv) The fourth step is also a loop. In the first iteration only t and t' get shifted before the algorithm proceeds to step (v), but in the second iteration of the loop the set of shifted trains is extended. The selection criteria are analogue to the ones of the shift operator but the size of the shift is the smallest (in absolute values) that solves the conflict.

(v) When all selected trains get their shift, there can still be a conflict somewhere. In the first iteration of step (iv) this is expected but also in the second iteration this can occur, for example, when shift causes a train to leave its predetermined time-window. In case the new timetable is feasible, a large and dull change was made. So nothing guarantees a direct improvement but many new possibilities for improvement have arisen. That is why, step (v) is a recursive call to the timetabling module but with a limitation on its maximum number of moves and without order swaps itself.

The idea is as follows. Upon start of this *internal timetabling module*, feasibility is checked. Is the solution infeasible, the RCL will only contain the conflicting trains. Otherwise, the RCL is constructed as usual. Next, one tries to increase the minimal time span of each RCL-pair such that feasibility is restored or the solution improved. When the stopping criteria are met (the solution is still

infeasible after a number of iterations or a maximum number of moves are made), the internal timetabling module stops and returns a solution to the order swap.

As output, order swap returns the best, feasible solution found after executing the internal timetabling module. Using this internal call to the timetabling module, allows us to work with infeasible solutions and to evaluate the potential of a move, i.e., the new possibilities created by the order swap. This feature has enhanced the strength of our algorithm a lot and made order swaps attractive. Note that computational tests give no clear distinction between which iteration of step (iv) performs best. Thus both are kept in the algorithm.

4.3.4. Tabu search

Because the above neighborhood structures can incur a large threat to cycle, we opted for a tabu search structure to embed our neighborhoods. The heuristic contains four steps. Initially, all tabu lists are emptied and the variable ϑ, which is used to determine the restricted candidate list, is set to 0. The main steps are summarized in Algorithm 3.

Algorithm 3. Tabu search algorithm

input: reference timetable and routing
while stop criteria 1 and 2 are not met **do**
  for $\vartheta = 0..(\varepsilon_{\text{step}})..\vartheta^{\max}$ **do**
    (i) construct the RCL using (4) or with all pairs of conflicting trains
    (ii) impro ← shift
      if !impro **do** impro ← shiftcombi
      if !impro **do** impro ← order swap
    (iii) **if** impro **do**
      $\vartheta \leftarrow 0$, update tabu list
    (iv) Perform smallest ascent move and update tabu list

(i) The search starts with determining the restricted candidate list (RCL). In case the current solution is infeasible, the RCL contains all pairs of conflicting trains. Otherwise, the RCL consists of all pairs of trains whose minimal time span $B_{t,t'}$ lies within the interval that is bounded from below by the overall minimal time span plus ϑ and has $\varepsilon_{\text{step}}$ as width, with $\varepsilon_{\text{step}}$ a model parameter.

$$\text{RCL} = \left\{ (t,t') \in T \times T : B_{t,t'} \in \left[\min_{(\dot{t},\dot{t}) \in T^2} B_{\dot{t},\dot{t}} + \vartheta,\ \min_{(\dot{t},\dot{t}) \in T^2} B_{\dot{t},\dot{t}} + \vartheta + \varepsilon_{\text{step}} \right) \right\}. \tag{4}$$

(ii) When the RCL is constructed, the shift neighborhood is explored. If an improved timetable is found, the algorithm proceeds to step (iii) with this new solution. Otherwise, the combined shift neighborhood is explored. Here the same principle is repeated and if no solution is accepted, an order swap is considered. When this did not result in a better solution, the search returns to step (i) with $\vartheta \leftarrow \vartheta + \varepsilon_{\text{step}}$ if $\vartheta \leqslant \vartheta^{\max}$ or proceeds to step (iv) with the best non-improving solution that is found since the last move was made (smallest ascent).

(iii) and (iv) The algorithm reaches step (iii) when an improving move is made. If no such move is found within the whole ϑ-for loop, a smallest ascent move is performed in step (iv). In both cases, the tabu lists are updated accordingly. This happens as follows. For all trains that got shifted, a reverse shift (independent of the size of the shift) becomes shift-tabu. If the order of two trains is swapped, these two trains become swap-tabu. Thus after an order swap, multiple trains will be added to the shift-tabu list (due to the internal timetabling module) and one pair of trains added to the swap-tabu list. The tabu tenures are deterministic but list dependent. Note that a candidate move that is tabu will only be accepted whenever it improves the globally best found solution (aspiration). After the tabu list update, the algorithm returns to step (i) with $\vartheta = 0$ unless the maximum number of moves (stop criteria 1) or the maximum number of non-globally improving moves (stop criteria 2) is reached, in which case the algorithm terminates.

The selected values of the parameters are given in Section 7. A common used feature of a tabu search is a diversification loop. Here we do not explicitly have a diversification phase but use the platforming module and the routing module to diversify the search.

4.4. Platforming module

Until now, as well as in Dewilde et al. (2013), the platform assignment of all trains is assumed to be fixed. When the combination of the routing module and the timetabling module gets stuck in a local optimum, we drop this assumption and consider the impact of a platform change. This asks for three decisions: (i) which trains should get a new platform and at which station, (ii) which platforms are to be considered for a given train and (iii) since a route is platform dependent, a new route needs to be selected out of the set of candidate routes. Changing a platform for a train incurs a risk on conflicts. In order to resolve possible conflicts or explore the potential of a platform change, an internal run of the timetabling module is added to the platforming module in a similar way as done for an order swap.

To keep the platforming move relevant and limit the computation time, we do not consider all possible platforming combinations but restrict the search to the relevant ones. In the remainder of this paragraph, we provide a short overview of the criteria that are used. We skip the details as these criteria mainly affect the computation time but hardly the solution quality, which we consider to be much more important.

First, only these trains for which we may expect a possible improvement are selected; only those $t \in T$ for which there exists a $t' \in T$ such that $B_{t,t'} \leqslant \min_{(\dot{t},\dot{t}) \in T^2} B_{\dot{t},\dot{t}} + B_{\text{margin}}$, with B_{margin} a model parameter, are eligible. Next, a selected candidate should be moved from a busy platform to a calmer one or the number of intersecting routes causing time spans smaller than $B^{\max}$ should be decreased. This is implemented because a busy platform limits the shift possibilities and preliminary results showed that trains that intersect at switch yards neighboring stations are more delay sensitive than those intersecting elsewhere. Next to the new platforms satisfying these criteria, a train's original platform is always considered as a candidate (new) platform. In this case, only a route change is applied. The train routing problem will not consider this (new) route if it gives rise to too small time spans or conflicts. That is why, applying the internal timetabling module to explore the potential of this new route can (and often will) be improving.

After deciding upon the set of candidate platforms for a train, all possible routes connecting that train's inbound and outbound line with any of the new platforms are selected. A dominance rule, similar to the one used for solving the train routing problem (Dewilde et al., 2013), is applied here to limit the number of candidate new routes. This also includes a restriction on the amount of (latent) conflicts of the new route in comparison with the old one.

If all these conditions are met, the combination of a train, a platform and a route is a candidate for a platform change and its potential impact will be checked using the internal timetabling module. This is done for all possible trains at all possible stations and at the

end the best platform change is selected if it leads to an improved solution. Then, the algorithm returns to the routing module.

5. Simulation

At the end of our algorithm, we have a solution (route, timetable and platform assignment) for which none of the three modules can find an improved version anymore. In comparison to the initial (reference) timetable, the spreading of the trains (our objective function) is improved. To test if and how much the performance has changed, we apply a self-developed simulation model on the station zone under consideration. Since several assumptions are made when creating this simulation model, the output needs to be interpreted with care. Nevertheless, the model is appropriate to evaluate the performance of our algorithm because all instances are valuated using the same assumptions such that a fair comparison is guaranteed.

In our discrete event driven simulation model, events are handled synchronously and stochastic influences are represented by input delays. Analogue to Fischetti et al. (2009) and Takeuchi et al. (2007), we use a simulation model to validate the improvements of the new schedule independent of our optimization algorithm. Since this simulation model has a secondary role with respect to our algorithm some assumptions can be made to reduce its complexity.

During each run of the simulation model, initial or external disturbances are represented by delays drawn from the exponential distribution of which the parameters are determined using real data. For the results in this paper, we gave half of the trains a delay upon entering the system and half of the trains a dwell delay. In Dewilde et al. (2013), we show that the improvement in robustness is stable (standard deviation of 1%) when varying the delay parameters. Note that we only consider small delays, i.e., those frequently occurring during the daily practice, and do not consider large disruptions.

Trains enter the system at their inbound line or at the platform of departure in case of a reutilization. In the first case, we apply a minimum headway of three minutes. Then, the trains travel from signal to signal at a predetermined speed that equals the real allowed maxima within the station zone. We assume that the time lost by slowing down or speeding up can be approximated by a constant and only affects the travel time through the first (last) block section after (before) a stop. The size of this constant is based on the difference between the expected travel time (when traveling at maximum allowed speed) and the scheduled travel time for that type of train through the corresponding sections. Note that we approximate the type of rolling stock (and thus the specific acceleration characteristics) by the type of train. This way, the simulated travel time through the first (last) section after (before) a stop will be similar to the planned travel time through that section.

Next to the assumptions about the speed profiles, we also made some simplifications concerning the blocking times. In Pachl (2008), the intervals that are part of the total blocking time are being described. In our simulation model, however, we only distinguish three different blocking time subintervals: the travel time through the section, the clearing time, and thirdly, an interval of constant size representing, among others, the signal processing. The first one is based on the ratio between the length of the block section(s) and the speed of the train together with the penalties for speeding up or slowing down (if applicable). The clearing time consists of the time needed for the tail of the train to leave the block section and is a function of the length of the train and its speed. The last subinterval captures the time needed for setting the signals and aligning the switches but also needed to release the section after the passage of a train. The minimum headway time is set to 3 minutes, which is a commonly used threshold for trains on a common line.

Events are handled chronologically and conflicts are not predicted in advance but only detected when a train approaches an already reserved block section. Conflicts are solved one by one on a first-come, first-serve basis meaning that, once detected, a conflict is solved immediately by postponing the next event of the approaching train until the estimated time the corresponding block section becomes available again. If multiple events become active simultaneously on a shared resource, extra priority rules, which are derived from practice, apply. This way, high-speed trains get priority on local commuter trains and slightly delayed trains may precede punctual trains in the event list. No real-time rerouting actions or cancelations of trains are allowed.

Because of the limited capacity, no running time supplements are planned on the lines between the stations. Nevertheless, dwell time supplements that are included in the reference timetable and running time supplements on the incoming or outgoing lines are kept constant within our algorithm. These supplements and the slack time that originates from rounding the arrival and departure times to minutes are considered as is done in practice and can result in extra waiting time at one of the stations.

For every performance measure, the average over 10000 simulation runs is taken as result. Together with the *spreading objective value*, this forms the rows of the result tables (Tables 2 and 4 in Section 7). First, we get the *robustness* value as computed by (1). Next, the *real total weighted travel time* (see Section 3) and the amount of *knock-on delay per hour* are given. The latter is the sum of all the delay that have arisen due to conflicts detected by the simulation model. Finally, the percentage of trains that entered without delay but leaves the system with delay (*newly delayed*) and those that left with more delay than they had upon arrival (*extra delayed*) are given.

As said above, the results from our simulation model need to be interpreted while keeping the assumptions in mind. Although some simplification is made, we do take the interaction between trains into account and apply rules from practice when solving conflicts. Introducing more details in the simulation model and necessarily also within the entire algorithm would complicate the computations a lot with only a moderate gain in accuracy as a result. The question then remains how external effects like, for example, the driver's behavior, can bias the results. Recognizing the drawbacks of our model, Infrabel has confirmed our results using their microscopic, asynchronous simulation package LUKS (VIA Consulting & Development GmbH, 2013). They extended the study area such that it contains an extra stopping station for each train. The improvements to the system they noticed, were of the same order of magnitude as the results we present in Section 7.

6. Case studies

Initially, the presented approach is used to investigate improvements in the performance on the network connecting the three

Table 2
Computational results for the case of Brussels.

	Reference	Routing (%)	Timetabling (%)	Platforming (%)
Spreading objective value	100%	71.9	20.6	**13.4**
Robustness	100%	100.6	105.2	**108.1**
Real total weighted travel time	10.6 min	99.7	97.6	**96.3**
Knock-on delay per hour	35 min	93.4	64	**49.3**
Newly delayed	8.5%	7.6	5.4	**3.9**
Extra delayed	33.6%	30	21.8	**17.1**

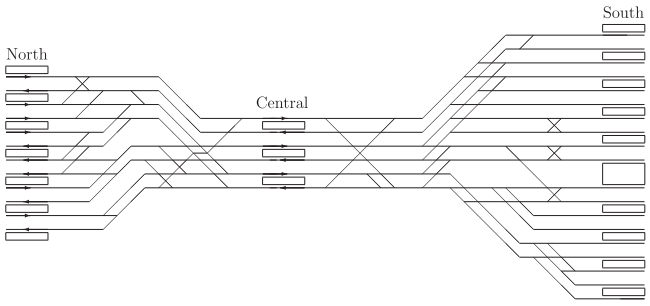

Fig. 2. Microscopic overview of the railway infrastructure in Brussels.

major stations in the zone of Brussels, Belgium's capital. The Brussels area truly is an interesting case study to evaluate our approach as it contains three of the country's four busiest stations regarding passenger numbers (Sels et al., 2011a). It includes the largest station with respect to the number of platforms and a true, physical bottleneck since 19 (through) platforms are connected through a 6-track tunnel, containing the busiest station, with the 12 platforms of the station on the other side. Fig. 2 gives a microscopic overview of the infrastructure connecting the three stations. According to the timetable, the travel time between the stations is 3 minutes. Next to the tracks towards the Central station and a shunt yard, each of the outer stations has four in- and outbound orientations such that trains run from all over the country towards Brussels, forming a crisscross of lines with many intersecting routes in the station zone as a consequence. In total there are up to 90 trains per (peak) hour making the capacity utilization nearly saturated. All of this makes that a small delay in the centrally located Brussels area easily spreads out in space and time. Thus, it is expected that an optimized local schedule and an appropriate routing through the station zone will help to improve the performance on the whole railway network.

To show the general applicability of our approach, we also consider a second case study that is significantly different from the first one. In the station zone of Antwerp, see Fig. 1, the two major stations are connected through three corridors allowing trains to arrive at three different levels in the Central station. The zone of Antwerp is an interesting case study because of the high capacity usage by a heterogeneous fleet of trains; international trains mix up with slow and fast local trains and with freight trains. Other challenges are that all but four platforms in the Central station are dead end tracks and that, in comparison with the study for Brussels, there are more restrictions on the combinations between inbound and outbound lines and the allocated platforms.

Currently, no list of important transfers within the zone of Brussels is available and as a consequence, no transfers are guaranteed in practice. Due to the high frequency of trains in each direction, the extra waiting time caused by a missed transfer remains limited. This is in contrast with the situation in Antwerp. As most of the lines terminate or originate there, transfers are important. Thus, next to the limited routing and platforming possibilities, also within the timetabling module extra constraints apply.

7. Results

In Dewilde et al. (2013), the first results of our study for the case of Brussels can be found. In the current paper, however, the computational results obtained with an extended and significantly better version of our algorithm are presented. Next to that, the general applicability of our methodology is illustrated by the new case study, the station zone of Antwerp. Since the algorithm contains several parameters, we start by giving their values which are determined based on (simple) computational experiments, see Table 1. The used stop criteria are the maximum number of moves (*stop criterium 1*) which is unbounded for the general timetabling module but limited to 30 for the internal timetabling module, and the maximum number of consecutive non-improving moves (*stop criterium 2*) equal to 200 and 5 for the general and internal timetabling module, respectively.

The algorithm is coded in C++, uses Cplex 12.5, and runs on a 2009-DELL Optiplex 760 with Intel(R) Core(TM) 2 Duo 3.00 GHz, 4.00 GB RAM, 64-bit operating system. Since the computation times for both case studies are of the same order of magnitude, we only report these for the case study of Antwerp. Where the routing module lasts less than 5 seconds, the nested structure of the timetabling module in combination with the potential checks make the computation times for the first iteration to be around 840 seconds but for the next iterations this decreases to about 145 seconds. Together with the platforming module which lasts between 275 and 850 seconds, the total computation time for the whole algorithm was 6700 seconds. In total, 6 platform changes and 11 calls to the timetabling module were made for the case study of Antwerp. For Brussels, similar results apply.

Considering the different neighborhood structures, one can say that restricting the timetabling module to the shift neighborhood is very fast (less than 5 seconds) but not effective. In combination with the combined shift neighborhood, a significant improvement is found (reduction of nearly 25% in the objective function value), but order swaps are needed to find the best solutions.

7.1. Brussels

Now, the computational results for the case study of the station zone of Brussels can be given. These results are summarized in Table 2 and 3. For the explanation of the rows in Table 2, we refer to Section 5. The columns represent the result of, respectively, the *reference* system, the reference timetable with improved *routing*, the solution obtained after the iterative procedure of routing and *timetabling* has reached a local optimum, and finally, the best found solution when also platform changes are allowed (*Platforming*).

In Table 3, the amount of knock-on delay that emerges due to conflicts in a station or on the switch zones between two stations is given. Based on the infrastructure from Fig. 2, five subareas are identified: the three stations (North, Central, and South) and the two switch zones connecting these. Each entry in the column *Reference* is the amount of knock-on delay, in minutes, that emerges in each of the corresponding zones. The other columns represent the fraction of the knock-on delay in the *Reference* situation that emerges in the corresponding zone for that case. Thus the 99.2% for the North station in the *Routing* column means that, on average, the amount of knock-on delay that is caused by conflicts within the North station, is equal to 99.2% of the 7.2 min in the *Reference* situation.

From the results in Table 2, we see that, except for the spreading objective value, changing the routing does not seem to have a large impact. This is because changing the routing only affects the time spans at the switch zones between the stations and, although the

Table 3
Knock-on delay within the zone of Brussels.

	Reference (min)	Routing (%)	Timetabling (%)	Platforming (%)
North	7.2	99.2	**67.6**	75
North-Central	4.8	71.1	46.3	**28.9**
Central	16	100.1	65.1	**54.4**
Central-South	4	79	58.8	**29.2**
South	2.9	99.6	85.1	**19.9**
Knock-on delay per hour	35	93.4	64	**49.3**

Table 4
Computational results for the case of Antwerp.

	Reference	Routing (%)	Timetabling (%)	Platforming (%)
Spreading objective value	100%	95.8	19.5	**16.5**
Robustness	100%	100	106	**108**
Real total weighted travel time	14.8 min	100	96.4	**95.1**
Knock-on delay per hour	70.1 min	100.4	71.4	**60.5**
Newly delayed	13.6%	13.5	9.3	**8.3**
Extra delayed	49.1%	49	33.9	**30**

Table 5
Knock-on delay within the zone of Antwerp.

	Reference (min)	Routing (%)	Timetabling (%)	Platforming (%)
border-Berchem	**2.6**	101.6	147.6	122.8
Berchem	28.4	100.5	65.8	**55.6**
Berchem-Central	15.6	100.6	68.1	**58.2**
Central	20.7	100.3	71.9	**63.4**
Central-border	2.8	98.7	74.3	**46**
Knock-on delay per hour	70.1	100.4	71.4	**60.5**

reduction in delay propagation varies between 20% and 30% there, this only represents a fraction of the total amount of knock-on delay. When routing is combined with timetabling (column *Timetabling*), a considerable reduction in delay propagation at all zones is achieved. This is reflected in a 5% improvement in robustness. The results in the last column show that the extra freedom given by allowing platform changes enables a robustness improvement of more than 8%. Concerning the real total weighted travel time itself, a reduction of nearly 4% is achieved by applying our algorithm. Although the total amount of knock-on delay is halved, we know from Table 3 that, locally, it can be reduced to 20% of the initial amount. Even in the bottleneck itself, the Central station, about half of the knock-on delay can be avoided by maximizing our spreading objective function.

7.2. Antwerp

Analogues to Table 2 and 3, the results for the case study of the station zone of Antwerp are presented in Tables 4 and 5. In Table 5, the station zone of Antwerp is divided into five areas; the two large stations (Berchem and Central) and the switch areas bordering these stations. The area Central-border contains the station Luchtbal. Note that about 30% of the considered trains passes this station but only 10% stops there while for the station Berchem this is 95% and 88%, respectively.

Unlike in Brussels, the specific layout of the infrastructure in Antwerp does limit the routing possibilities considerably. As a result, the routing module alone does not lead to improvement. Since the Central station is a (partial) terminus, reutilizations are common. This implies that larger free time slots at a platform are required to apply a platform change then in the case of a through station. Hence, the impact of adding the platforming module is moderate compared to the effect of the timetabling module. Nevertheless, we are convinced that significant improvements can be obtained by changing the platform allocation on a larger scale in combination with altering the reutilizations, i.e., assigning a train to another line after it has terminated.

The large amount of knock-on delay at the Central station and the surrounding areas, as well as the larger values in the row *extra delayed*, are another consequence of the terminus. Where arrivals at the other stations occur on a first-come, first-serve basis and the platform occupation times remain limited, in the Central station, a train might have to wait before entering the station because of another train that blocks its platform. Furthermore, all ingoing and outgoing routes, except for the limited amount of *through trains*, use the same infrastructure. Thus, also the total occupation time of the block sections bordering the Central station will be larger than for the other stations.

Comparing the results for Brussels and Antwerp, we see the same trends; a huge decrease in spreading objective function value, a robustness increase of about 8% together with a reduction of about 4–5% in real travel time. The difference in knock-on delay per hour is due to the above mentioned differences in setting but in both cases the improvement per zone is more or less stable, except for Brussels' Central station (nearly saturated capacity usage) and the border zones (first meeting place).

8. Conclusions

In this paper, a new way to improve the robustness in railway bottlenecks is presented. The objective of our methodology is to increase the minimal time span between any two trains in a station zone. Our algorithm is able to fully exploit the potential of an integrated optimization of routing decisions, train sequences and schedules, and platform allocations. Since our improvement modules do not suggest infrastructure changes, the cost for implementing the new planning is very small. Simulation results show that we are able to reduce the passengers' travel time within the considered area, which is an important quality measure, with 4 to 5%. Other results predict that by using our algorithm a gain in robustness of about 8% can be achieved. This improvement is due to a decrease in knock-on delay of 40–50% and a considerable reduction in the number of trains that are hindered during their passage through the bottleneck. The generality of our approach is shown by a second case study, with completely different settings, for which the algorithm returns similar results.

Some thoughts for further research are: including the passenger flows and potential delays into the improvement modules or considering more case study specific features like improving the reutilizations timings or the platform reallocation possibilities. A large potential for improving the robustness can be found in considering infrastructure changes. By specific changes in the layout of a switch yard, new routes can be created which allow for further improvements of the routing solutions.

Acknowledgement

Research partially funded by a Ph.D. grant of the Flemish Agency for Innovation by Science and Technology (IWT).

References

Cacchiani, V., Caprara, A., Galli, L., Kroon, L., Maróti, G., and Toth. P., 2008. Recoverable robustness for railway rolling stock planning. In: Fischetti, M., Widmayer, P. (Eds.), Proceedings of the 8th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems, volume 14 of OpenAccess Series in Informatics (OASIcs), Dagstuhl, Germany.

Cacchiani, V., Toth, P., 2012. Nominal and robust train timetabling problems. Eur. J. Oper. Res. 219, 727–737.

Caimi, G., Burkolter, D., Herrmann, T., 2005. Finding delay-tolerant train routings through stations. In: Fleuren, H., Hertog, D., Kort, P. (Eds.), *operat. res. proceed.*, vol. 2004. Springer–Verlag, pp. 136–143.

Caimi, G., Burkolter, D., Herrmann, T., Chudak, F., Laumanns, M., 2009. Design of a railway scheduling model for dense services. Networks Spatial Econ. 9, 25–46.

Caimi, G., Chudak, F., Fuchsberger, M., Laumanns, M., Zenklusen, R., 2011. A new resource-constrained multicommodity flow model for conflict-free train routing and scheduling. Transp. Sci. 45, 212–227.

Caprara, A., Galli, L., Toth, P., 2011. Solution of the train platforming problem. Transp. Sci. 45, 246–257.
Carey, M., Crawford, I., 2007. Scheduling trains on a network of busy complex stations. Transp. Res. Part B: Methodol. 41, 159–178.
Cicerone, S., D'Angelo, G., Di Stefano, G., Frigioni, D., Navarra, A., Schachtebeck, M., Schöbel, A., 2009. Recoverable robustness in shunting and timetabling. In: Ahuja, R.K., Möhring, R.H., Zaroliagis, C.D. (Eds.), Robust and Online Large-Scale Optimization, volume 5868 of Lecture Notes in Computer Science. Springer, Berlin Heidelberg, pp. 28–60.
Corman, F., Goverde, R.M.P., D'Ariano, A., 2009. Rescheduling dense train traffic over complex station interlocking areas. In: Ahuja, R.K., Möhring, R.H., Zaroliagis, C.D. (Eds.), Robust and Online Large-Scale Optimization, Lecture Notes in Computer Science, vol. 5868. Springer, Berlin Heidelberg, pp. 369–386.
Corman, F., D'Ariano, A., Pacciarelli, D., Pranzo, M., 2010. A tabu search algorithm for rerouting trains during rail operations. Transp. Res. Part B: Methodol. 44, 175–192.
Cornelsen, S., Stefano, G., 2007. Platform assignment. In: Geraets, F., Kroon, L., Schöbel, A., Wagner, D., Zaroliagis, C.D. (Eds.), Algorithmic Methods for Railway Optimization, Lecture Notes in Computer Science, vol. 4359. Springer, Berlin Heidelberg, pp. 233–245.
D'Ariano, A., Corman, F., Pacciarelli, D., Pranzo, M., 2008. Reordering and local rerouting strategies to manage train traffic in real time. Transp. Sci. 42, 405–419.
Dewilde, T., Sels, P., Cattrysse, D., Vansteenwegen, P., 2011. Defining robustness of a railway timetable. In: Proceedings of 4th International Seminar on Railway Operations Modelling and Analysis (RailRome).
Dewilde, T., Sels, P., Cattrysse, D., Vansteenwegen, P., in press. Improving the robustness in railway station areas. Eur. J. Oper. Res., doi: 10.1016/j.ejor.2013.10.062.
Dollevoet, T., Huisman, D., Schmidt, M., Schöbel, A., 2012. Delay management with rerouting of passengers. Transp. Sci. 46, 74–89.
Fischetti, M., Monaci, M., 2009. Light robustness. In: Ahuja, R.K., Möhring, R.H., Zaroliagis, C.D. (Eds.), Robust and Online Large-Scale Optimization, volume 5868 of Lecture Notes in Computer Science. Springer, Berlin Heidelberg, pp. 61–84.
Fischetti, M., Salvagnin, D., Zanette, A., 2009. Fast approaches to improve the robustness of a railway timetable. Transp. Sci. 43, 321–335.
Goldratt, E.M., 1984. The Goal: A Process of Ongoing Improvement. North River Press.
Goverde, R.M.P., 2007. Railway timetable stability analysis using max-plus system theory. Transp. Res. Part B: Methodol. 41, 179–201.
Huisman, D., Kroon, L.G., Lentink, R.M., Vromans, M.J.C.M., 2005. Operations research in passenger railway transportation. Stat. Neerlandica 59, 467–497.
Kanai, S., Shiina, K., Harada, S., Tomii, N., 2011. An optimal delay management algorithm from passengers' viewpoints considering the whole railway network. J. Rail Transp. Plann. Manage. 1, 25–37.
Kroon, L., Huisman, D., Maróti, G., 2008a. Optimisation models for railway timetabling. In: Hansen, I.A., Pachl, J. (Eds.), Railway Timetable & Traffic: Analysis, Modelling and Simulation. Eurailpress, Hamburg, pp. 135–154.
Kroon, L., Maróti, G., Retel Helmrich, M., Vromans, M., Dekker, R., 2008b. Stochastic improvement of cyclic railway timetables. Transp. Res. Part B: Methodol. 42, 553–570.
Liebchen, C., Lubbecke, M., Möhring, R., Stiller, S., 2009. The concept of recoverable robustness, linear programming recovery, and railway applications. In: Ahuja, R.K., Möhring, R.H., Zaroliagis, C.D. (Eds.), Robust and Online Large-Scale Optimization, volume 5868 of Lecture Notes in Computer Science. Springer, Berlin Heidelberg, pp. 1–27.
Lindner, T., 2011. Applicability of the analytical UIC Code 406 compression method for evaluating line and station capacity. J. Rail Transp. Plann. Manage. 1, 49–57.
Lusby, R.M., Larsen, J., Ehrgott, M., Ryan, D., 2011. Railway track allocation: models and methods. OR Spectr. 33, 843–883.
Meng, L., Zhou, X., 2011. Robust single-track train dispatching model under a dynamic and stochastic environment: A scenario-based rolling horizon solution approach. Transp. Res. Part B: Methodol. 45, 1080–1102.
Pachl, J., 2008. Timetable design principles. In: Hansen, I.A., Pachl, J. (Eds.), Railway Timetable & Traffic: Analysis, Modelling and Simulation. Eurailpress, Hamburg, pp. 9–42.
Salido, M.A., Barber, F., Ingolotti, L., 2012. Robustness for a single railway line: Analytical and simulation methods. Exp. Syst. Appl. 39, 13305–13327.
Schöbel, A., Kratz, A., 2009. A bicriteria approach for robust timetabling. In: Ahuja, R.K., Möhring, R.H., Zaroliagis, C.D. (Eds.), Robust and Online Large-Scale Optimization, volume 5868 of Lecture Notes in Computer Science. Springer, Berlin Heidelberg, pp. 119–144.
Sels, P., Dewilde, T., Cattrysse, D., Vansteenwegen, P., 2011a. Deriving all passenger flows in a railway network from ticket sales data. In: Proceedings of 4th International Seminar on Railway Operations Modelling and Analysis (RailRome).
Sels, P., Waquet, B., Dewilde, T., Cattrysse, D., Vansteenwegen, P., 2011b. Calculation of realistic railway station capacity by platforming feasibility checks. In: Proceedings of 2nd International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS).
Sels, P., Dewilde, T., Cattrysse, D., Vansteenwegen, P., 2013. Expected passenger travel time for train schedule evaluation and optimization. In: Proceedings of 5th International Seminar on Railway Operations Modelling and Analysis (RailCopenhagen).
Shafia, M.A., Aghaee, M.P., Sadjadi, S.J., Jamili, A., 2012. Robust train timetabling problem: Mathematical model and branch and bound algorithm. IEEE Trans. Intell. Transp. Syst. 13, 307–317.
Snelders, M., Immers, B., Wilmink, I., 2004. De begrippen betrouwbaarheid en robuustheid nader verklaard (in Dutch). In: Colloquium Vervoersplanologisch Speurwerk, Zeist, The Netherlands.
Takeuchi, Y., Tomii, N., Hirai, C., 2007. Evaluation method of robustness for train schedules. Q. Rep. Railway Tech. Res. Inst. 48, 197–201.
Tamura, K., Tomii, N., Sato, K., 2013. An optimal rescheduling algorithm from passengers' viewpoint based on mixed integer programming formulation. In: Proceedings of 5th International Seminar on Railway Operations Modelling and Analysis (RailCopenhagen).
Tassenoy, S., Maricau, E., Scheerlinck, K., Büker, T., Verboven, S., 2013. Evaluating the robustness of a railway timetable: a practical approach. In: Proceedings of 5th International Seminar on Railway Operations Modelling and Analysis (RailCopenhagen).
UIC (Union Internationale des Chemins de Fer). 2004. UIC Code 406 – Capacity.
Vansteenwegen, P., Van Oudheusden, D., 2006. Developing railway timetables which guarantee a better service. Eur. J. Oper. Res. 173, 337–350.
Vansteenwegen, P., Van Oudheusden, D., 2007. Decreasing the passenger waiting time for an intercity rail network. Transp. Res. Part B: Methodol. 41, 478–492.
VIA Consulting & Development GmbH. 2013. Luks - analysis of lines and junctions. http://www.via-con.de/development/luks/, October.
Vromans, M.J.C.M., Dekker, R., Kroon, L.G., 2006. Reliability and heterogeneity of railway services. Eur. J. Oper. Res. 172, 647–665.
Zwaneveld, P.J., Kroon, L.G., Romeijn, H.E., Salomon, M., Dauzère-pérès, S., Van Hoesel, S.P.M., Ambergen, H.W., 1996. Routing trains through railway stations: model formulation and algorithms. Transp. Sci. 30, 181–194.
Zwaneveld, P.J., Kroon, L.G., van Hoesel, S.P.M., 2001. Routing trains through a railway station based on a node packing model. Eur. J. Oper. Res. 128, 14–33.