Improving the robustness in railway station areas

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*

Abstract

In order to improve the robustness of a railway system in station areas, this paper introduces an iterative approach to successively optimize the train routing through station areas and to enhance this solution by applying some changes to the timetable in a tabu search environment. We present our vision on robustness and describe how this vision can be used in practice. By introducing the spread of the trains in the objective function for the route choice and timetabling module, we improve the robustness of a railway system. Using a discrete event simulation model, the performance of our algorithms is evaluated based on a case study for the Brussels' area. The computational results indicate an average improvement in robustness of 6.2% together with a decrease in delay propagation of about 25%. Furthermore, the effect of some measures like changing the train offer to further increase the robustness is evaluated and compared.

Keywords: Transportation, Robustness, Railway Timetabling, Train Routing, Bottleneck Scheduling, Mixed Integer Linear Programming

1. Introduction

Providing a good passenger service is an important task of a railway operator and an infrastructure manager. One of their main concerns is to bring all passengers from their origin to their destination in a travel time as close as possible to the published travel time. Although the prescribed schedule

*Corresponding author
Email address: Thijs.Dewilde@cib.kuleuven.be (Thijs Dewilde)

is conflict-free, there are always unavoidable disturbances that cause conflicts; definitely around large cities or, more generally, in the neighborhood of railway bottlenecks. In this paper, the timetable planning for and the train routing through bottlenecks is considered. In general, the assignment of routes through railway station areas happens after the timetable is created. This timetable is constructed based on a macroscopic model, while the routing of a train towards a platform is planned using a microscopic model considering all tracks and switches in the station area. Accounting for robustness while designing an appropriate routing is only considered in a small number of publications. Nevertheless, the impact of a robust routing solution is large as is indicated in [2, 20]. The results in this paper confirm this statement. Next to that, we show that applying our methodology outperforms the approach of Caimi et al. [2] for the case study of Brussels.
In this paper, the focus lies on the planning phase in which the schedule is to be improved. No real-time conflict resolution actions are considered. Starting from a given timetable, the question we want to address is: *How can we improve the robustness around large railway stations?* In order to answer this question, an objective function that maximizes the time span between any two trains is defined and the way trains are routed to the platforms is optimized in the scope of this objective. In a next step, small modifications to the timetable to further increase some time spans are considered. The results in this paper indicate that exploring the interaction between the routing and scheduling of trains in the surroundings of a bottleneck area, is promising for achieving an improved robustness. Increasing time spans is a commonly applied approach to make timetables more robust [2, 19]. However, large time spans do not necessarily imply robustness. We define robustness as a broader concept, as is explained in the discussion about robustness below. Nevertheless, it is clear that increasing a time span between two trains helps to avoid conflicts between these trains, which is beneficial for the robustness of a system.

1.1. Case study

The performance of our algorithm is tested on the compact and highly used network of the Brussels' area, the heart of the Belgian railway system. It has three major stations: South (19 through platforms), Central (6 platforms), and North (12 platforms) which are lined up on a 6-track line, see Figure 1 for a microscopic overview of the infrastructure. Next to the tracks towards the Central station and a shunt yard, each of the outer stations has

four in- and outbound orientations. Trains coming from all over the country run through Brussels and dwell at all three stations forming a crisscross of lines with many merging and intersecting actions in the switch zones between the stations as a consequence. In total there are 84 trains per hour during rush hour making the capacity utilization nearly saturated. Passenger flows, as computed in Sels et al. [31] or counted in 2009 [35], indicate that the Central station is the busiest station in Belgium with an average of more than 70000 passengers each day. The South and North station are ranked as, respectively, second and fourth most busiest stations of the country. Although it is the busiest station, capacity limitations force the planned dwell time in the Central station to be limited to one minute which is nearly always insufficient in practice. This results in dwell delays of at least half a minute on average.

Because of the high probability of delays and their large impact on the rest of the network, a robust planning for the Brussels' area is essential. In this paper, we present some methods to improve the robustness in Brussels, or more generally, in a complex and highly used station area. Therefore, we make the assumption that changes to the local planning do not harm the feasibility on the less occupied network outside the considered area or at other stations.

To improve the robustness of the railway system in Brussels even more, we suggest some measures like changing the stopping pattern, a reduction of the number of trains that runs through Brussels, or the creation of extra platforms for the Central station. Using our approach, the impact of these measures on robustness and the amount of knock-on delay can be estimated. The presented work is conducted in cooperation with the Belgian railway infrastructure manager Infrabel.

1.2. Definitions

Throughout this paper, we define the *station area* as the set of stations that lie within this area, together with their platforms and the tracks and switches connecting the incoming lines with the different platforms and the outgoing lines. An overview of the station area of Brussels is given in Figure 1. The dashed line indicates what we call a *route* through the station area. Next to the notion of a station area, we also define the *minimal time span*, or just time span, between two trains as the smallest delay that causes these trains to conflict in the station area. Trains ride from one block section to another block section and occupy these block sections for a certain (blocking) time. The interval during which a block section is idle between

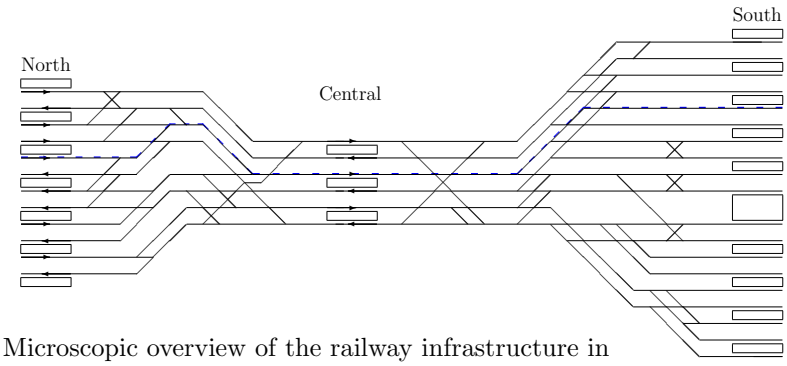

Figure 1: Microscopic overview of the railway infrastructure in Brussels. The dashed line represents a route through the station area

the passage of two trains is what we call a time span. The minimal time span between two trains is then the minimum of all these time spans over all common sections. We say that a routing solution or a train schedule is *feasible* or *conflict-free* if all the minimal time spans between all pairs of trains are nonnegative.

As mentioned above, two aspects of the planning within a station area are considered in this paper. The *train routing problem* is about finding a route for each train through the considered network, in our case a station area. In what we call the *timetabling problem*, we keep the routing solution but allow variations in arrival and departure times.

At last, the difference between the *planned travel time* and the *(average) (total) real travel time* is clarified. As the name states, the planned travel time is the duration a certain trip takes according to the planned timetable, so under ideal circumstances. In reality, when this trip is made, there can be a disturbance or conflict that extends the travel time. The (total) real travel time is defined as the total duration 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 we start counting from the planned departure time as we assume that passengers arrive at their station of departure before that moment. Since the real travel time differs each time a trip is made, we use the average real travel time instead but drop the term *average*.

The structure of this paper is as follows. In the next section, an overview of the related literature is given and the differences between our method and other approaches are discussed. Section 3 is about our vision on robustness.

In Section 4, our algorithm and the simulation module are presented and in Section 5, we provide the computational results. The paper ends with the conclusions and thoughts for further research.

2. Literature

2.1. Train routing problem

In general, the whole railway planning process consists of several steps [18, 23]. One of these is the timetabling step in which a timetable is created based on a macroscopic representation of the infrastructure, so partially neglecting capacity limitations at stations. In a next step, called the train routing or platforming problem, the track layout at microscopic level is considered and planners try to find conflict-free routes through station areas. An extensive overview of the train routing problem at station level can be found in the survey paper of Lusby et al. [23]. A classification of related contributions is given based on the level of planning (strategic, tactical and operational) and on the solution approach. Our paper, as well as the related papers [4, 24], belongs to the class of tactical level planning for which we use a conflict graph. Our approach distinguishes itself from the train platforming problem (TPP) [5] since for solving the train routing problem, we assume a rigid platform allocation. The situation in which multiple stations are lined up is studied in [6]. In that paper, an algorithm to come up with a conflict-free schedule for a series of stations is presented, but no robustness related topics or routing actions as we do are tackled. The radial structure of the British network motivates the approach of scheduling corridors largely separated. Conflicts between the network components are to be solved by iterations or adjusting the schedules. As the structure of the Belgian network is more or less radial with Brussels as midpoint, a similar approach is used in this paper.

Some of the main contributors concerning the train routing problem on a tactical level are Zwaneveld et al. In [42] and [41], the routing method that is used within DONS, the planning tool used for the Dutch railway system, is presented. First, all possible routes connecting the inbound track with a platform and the outbound track are computed. Then, a preprocessing phase is executed to reduce the number of candidate routes for each train. Finally, the remaining train routing problem is modeled as an independent set problem. The objective is to find a conflict-free combination of routes such that each train gets exactly one route assigned. This approach is enhanced with

some robustness features, using aggregated routes, by Kroon and Maroti [20]. The work of Caimi [2, 3, 4] has similarities with the Dutch approach. In [3], a time index to enable small changes to the timetable is inserted in the model. This increases the complexity of the model which is then solved with a fixed point iteration heuristic [2]. The output consists of a routing for each train and a (modified) timetable. In [4], the independent set approach is replaced by a resource tree to enable computing feasible routings, eventually for slightly modified timetables, in a few seconds.

A difference between our approach and the work of Zwaneveld et al. [42, 41] is that we have included a robustness-oriented objective function in our model. Also in [2], an objective function to improve the robustness is used. Although the idea of their approach, maximizing the size of the minimal disturbance that causes knock-on delays, is similar to ours, we show at the end of Section 4.2 that only considering the smallest time spans as they do, is not sufficient for obtaining a robust solution for our case study. That is why we include all minimal time spans in the form of a weighted sum ranging over all time spans. As a consequence, a mathematical model can be used to efficiently compute the optimal routing solution. When only the smallest minimal time slots are considered in the objective function, Caimi et al. found that using a mathematical model becomes too time consuming. Therefore, a fixed point iteration heuristic is used to come up with a routing solution in which the smallest time slots are increased considerably in comparison to the initial solution [2].
In Zwaneveld et al. [42, 41], as well as in Caimi et al. [3, 4], small timetable modifications are allowed to obtain a feasible train routing. We use a different approach of alternating between routing and timetabling, both aiming to increase the robustness.

2.2. Robust timetabling

As is indicated by Cacchiani and Toth in their survey on robust train timetabling [1], there is a large set of papers about timetable robustness in which each author captures the notion of robustness in their own way [19, 21, 22, 30]. Undoubtedly, providing a timetable with a certain amount of supplements and buffers to avoid or absorb delays and to avoid knock-on delays, respectively, helps to improve the robustness [19], but there is a limit to the amount added; once a timetable becomes too slow, it will not be practical anymore. Improving the scheduling of transfers whilst limiting or minimiz-

ing the planned (passenger) travel time is also essential to obtain a robust timetable [30, 37, 38]. However, it has little to no use to improve a transfer that is not chosen by any passenger. Thinking about the passengers' perspective, robustness should be about minimal delays, good transfers, short planned travel times, shortly said, anything that can improve the passengers' service [22, 31].

Considering passengers' service, or passengers' satisfaction as they call it, is also done in [33]. But unlike we do, they consider robustness as opposed to the passengers' satisfaction because of the longer travel times the first implies [33]. The proposed approach computes the required buffer times to achieve the desired level of robustness, based on the stochastic behavior of disturbances with respect to the possibility of delay propagation.

The concept of recoverable robustness [21] and recover to optimality [16] considers restoring feasibility in case of disruptions. They aim at minimizing the planned travel time plus a weighted delay cost (recoverable robustness) or at minimizing the necessary changes to restore feasibility (recover to optimality). Liebchen et al. [21] include the planned travel time and possible delays in the objective function, but the delay cost is purely based on the worst case values.

Some techniques to obtain a robust timetable based on a nominal timetable are compared in [14]. One of these techniques is the principle of Light Robustness [13]. The authors call their approach robustness training. The idea is to allow a worsening in the objective function value of the nominal timetable and to use this freedom to improve the robustness. This way, a solution somewhere between a robust optimization solution and a timetable resulting from stochastic programming is obtained.

In [28], four methods to improve the timetable robustness are presented: these four are adding time supplements, decreasing capacity consumption by adding buffer times to the headways, avoiding heterogeneity and decreasing the average speed. All of these incur a certain price of robustness in the sense that a higher level of robustness goes hand in hand with a larger travel time. The robustness is measured using the total delay but no assessment between the increase in planned travel time (the price of robustness) and the gain in real travel time of having less delays is made. Although the effectiveness of these methods in reducing the total delay is undoubtable, none of these methods are applicable in our case study due to a hard requirement of scheduling a given train set on an infrastructure with fixed capacity. We show that by modifying the schedule and the routing of some of the trains

in a better way, the performance can be improved in the considered area without increasing the planned travel time.
The work of Vansteenwegen and Van Oudheusden [37, 38] is about investigating optimal transfer planning for passengers while taking robustness into account. By planning a running time supplement on the feeder train's trip towards the transfer station, it is shown that the overall passenger travel time can be decreased. Based on the findings presented in [12], Sels et al. [32] came up with an integer programming model to compute a robust railway timetable for periodic trains in only a few hours. The way to deduce the passenger flows is presented in [31].
Even if a timetable is robust, disturbances still can cause conflicts which need to be solved in real-time. This real-time rescheduling belongs to the operational level of planning and is gaining more and more attention. As we focus in this paper on the tactical level planning, we restrict ourselves to a short overview discussing the most related trends in operational planning. Results concerning delay management with a rolling horizon on a single track network can be found in [27] and the references contained therein. Extending the network to lines between stations [7, 11] or considering real-time rescheduling or rerouting in railway bottlenecks [8, 9, 10, 26] has also been the subject of several studies.

2.3. Robustness evaluation

A robustness definition has to be practical in the sense that railway companies are willing to use it. This means that striving for robustness should not make the timetable too slow, and a robustness definition has to be practical to work with such that the robustness can be assessed and compared. Thus, besides a definition of robustness, an appropriate way to measure and compare timetable robustness is needed [28]. As dividing delay in primary delay and knock-on delay is still a very difficult exercise, it is hard to use anything related to this to measure robustness. Also the percentage of maintained transfers has shortcomings as this does not account for the delays arising due to, for example, synchronization actions.
An interesting method to evaluate robustness is described in [17]. Using the theory of max-plus algebra, delay propagation computations can be done easily. Next to that, the stability (existence of timetable slack), realizability and recoverability (resilience) can be evaluated. The robustness of a schedule is measured using the amount of propagated delay. Although this technique has led to the development of a simulation package called PETER, the ab-

sence of the ability to incorporate stochastic influences in the model forms a restriction for the use of this approach.
In [36], Monte Carlo simulation is used to estimate the robustness of a schedule. To evaluate the robustness, the authors measure the sensitivity to delays of the passenger disutility which is influenced by the travel time, congestion rate, number of transfers, and the waiting time. The used approach is tested by comparing the robustness of two schedules: a reference and an (ad hoc) improved one. Just like [36] and [22] with passengers' service, we share the same vision on robustness and apply a similar technique to measure it as is explained in the next section.

3. Robustness

The definition we have presented in [12], also in joint cooperation with the Belgian railway infrastructure manager Infrabel, removes the discrepancy between the existing definitions: the inclusion of timetable slack is assessed by considering its marginal effect, passenger streams are used as weights such that, e.g., important transfers can be detected, and the total passengers' travel time including delays, transfer times and necessary waiting is being used to measure the quality of the passengers' service. A good quality of this passengers' service, even under disturbed circumstances, is what robustness is about [34]. That is why we, together with Infrabel, define railway robustness as follows: *A railway system that is robust minimizes the total weighted real travel time of the passengers, in case of frequently occurring, small disturbances.*
Since the total real travel time of a passenger, as defined in Section 1, is easily measured, we have a constructive definition for robustness. Weights are added to represent the passengers' perception of wasted travel time. Wasted travel time can be seen as anything that prolongs the journey time unnecessarily: delays (weight 3), supplements that result in extra unnecessary dwell time (weight 2) and transfers. For transfers this goes in the form of waiting on the platform until the connecting train arrives and is weighted by a factor 2. The size of the weights originates from extensive studies about the value of (waiting) time in public transportation [25, 40].
The total weighted real travel time for all passengers is computed using the weights as described above. In Section 5.2, we compare different measures with different nominal travel times, e.g., due to changes in the stopping pattern. For the fairness of comparison, we normalize the total weighted real

travel time by subtracting the nominal travel time of all passengers from it and dividing this result by the same nominal travel time. This ratio gives a measure for the *weighted travel time extension* (WTTE). The weighted travel time extension equals

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

To evaluate the robustness of system X, it is hard to know when the maximum level of robustness is achieved. Hence in this paper, robustness related results are presented in comparison to a reference situation (REF) such that the gain in robustness can be computed. Therefore the following formula is used:

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

If (1) results in a value of, e.g., 1.10 or 110%, this means that railway system X is 10% more robust than the reference system according to our definition of robustness.
Note that the visions on robustness as formulated in [19, 21, 22, 30] are all captured in our definition.

4. Algorithms

In this section, the route choice module to solve the train routing problem, the timetabling module, and the simulation module to evaluate the performance at a railway bottleneck are introduced. Since both the timetabling module and the route choice module affect the minimal time span between two trains, defined as the minimal delay that causes these trains to conflict, an iterative procedure is used to combine both modules. Starting from an initial timetable, the route choice module is solved and based on its results the timetabling algorithm is run. This procedure is repeated as long as improvements are obtained.

4.1. Objective function

The objective function of the route choice module and the timetabling module is similar to the objective function from Caimi et al [2]. In their paper, the authors try to maximize the weighted sum of the four smallest minimal time spans between any two trains, while in this paper, we sum over all minimal time spans. Unlike the weights in the objective function of

Caimi et al. [2], our weights are non-linear and provide a *natural* assessment of the costs and benefits of altering some time spans. In the next section, we compare both approaches and show that ours is more appropriate for the Brussels' area.

In the following, we use the term *trainroute* to indicate the combination of a train (t) and its route through the station area (r_t), e.g., (t, r_t) is a trainroute. The minimal time span between two trains depends on both aspects of a trainroute: the train and its route. Let $B_{(t,r_t),(t',r_{t'})}$ (from Buffer time) be the minimal time span between trains t and t' with their routes r_t and $r_{t'}$, respectively, and let the weight corresponding to a time span be

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

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

$$\min \sum W_{(t,r_t),(t',r_{t'})}. \tag{3}$$

In this way, a smaller time separation corresponds to a higher cost and the marginal effect of increasing a time span is decreasing. Another advantage is that, when increasing a time span, the objective function weighs up the increased and decreased time spans in a natural way.

Our objective function is similar to the SSHR measure (sum of the shortest headway reciprocals) that is introduced in [39]. In that paper, the impact of heterogeneity on the reliability of a railway system is investigated. Using the sum of the reciprocals, one is able to account for speed differences since these lead to different time spans at the beginning and end of a common section. Where the applicability of the SSHR measure is restricted to track sections between stations, we also consider the stations itself and use (3) as the objective function instead of a reliability measure only as is done in [39]. Function (3) is used as the objective function in the route choice module as well as in the timetabling phase. It is important to notice that, although we use (3) to improve the robustness, this is not how the robustness itself is measured (see Section 3).

4.2. Route choice module

The train routing problem [23] is about finding a path for each train through the considered network, in our case a station area. More precisely, given a set T of trains and the set R of all possible routes through a station, 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. In this step, we assume that the arrival and departure time of each train are unchangeable and throughout the entire algorithm we keep the assigned platforms fixed. Often, the train routing problem is solved only for feasibility [3, 42], leaving several opportunities unconsidered. In this paper, feasibility is not enough since robustness is our goal. Including robustness in the objective function would result in a complicated model since delay propagation computations are required to evaluate the robustness as is explained in Section 3. That is why the spreading of trains is used here to indirectly improve the robustness. The impact on the robustness is measured afterwards using the simulation model from Section 4.4. To solve the train routing problem, our route choice module consists of three steps.

Step 1: In the first step, the set R of all possible routes is determined. With a route, we mean a sequence of (platform-)tracks from one end of the station area to the other end. An example of a route is highlighted in Figure 1. Each route connects the three stations and alternative routes (between the same platforms) exist. Comparing the routes connecting the same entry or exit point with the same platform, some routes can already be pruned. This is explained as follows. Denote with N_r the set of switches along route $r \in R$. If $N_r \subset N_{r'}$ with r and r' two routes between the same in-/outbound track and a platform, then route r' can be removed from R since replacing r' by r in any solution of the train routing problem will result in a better routing; all the switches on route r are also part of route r' which might have some extra switches, causing the minimal time span between a train following r and any other train to be at least as large as when route r' would be followed. Notice that it is assumed that no overtaking of trains takes place in a station area.

Step 2: The second step of the route choice module starts with assigning to each train all possible, remaining routes connecting that train's inbound or outbound track with its platform. Name this set R_t ($t \in T$). Now each $r \in R_t$ connects that train's platform in the North station with the platform in the South station and passes along the assigned platform of the Central station. Regarding the infrastructure of the Brussels' area (Figure 1), it can

be seen that in general R_t consists of multiple routes.
When the set R_t for each $t \in T$ is created, a preprocessing phase follows to reduce the number of candidate routes per train. Where in the previous step, routes could only be pruned based on their set of switches, now timetable information can be used to compute $B_{(t,r),(t',r')}$, with trainroutes $(t,r) \in T \times R_t$ and $(t',r') \in T \times R_{t'}$. When trainroute (t,r) conflicts with trainroute (t',r'), the minimal time span between these two, is set to 0. For all routes of train $t \in T$, say $r, r' \in R_t$, holds that $B_{(t,r),(t,r')} = 0$ since each train can only have one assigned route. In [42], some dominance rules are used to prune candidate trainroutes based on their set of conflicting trainroutes. If for a certain trainroute (t,r), the set of its conflicting trainroutes contains that of another route for the same train, say (t,r'), then according to Zwaneveld [42], (t,r) is dominated by (t,r') and can be pruned. Unlike in [42], we are not satisfied with feasibility, i.e., the ability to assign a route to each train in a conflict-free way, thus some modifications to the dominance rules are needed. A trainroute (t,r) can only be dominated by another trainroute for the same train, say (t,r'), if the minimal time span with each other trainroute is at least as large using (t,r') as when (t,r) is used. Therefore, define the set $K_{(t,r)}$ as follows:

$$K_{(t,r)} = \left\{ (t',r') \in T \times R_{t'} \; : \; \begin{array}{l} B_{(t,r),(t',r')} = 0, \\ \forall (\bar{t},\bar{r}) \in T \times R_{\bar{t}} : B_{(t,r),(\bar{t},\bar{r})} \leq B_{(t',r'),(\bar{t},\bar{r})}. \end{array} \right\} \tag{4}$$

Thus all $(t',r') \in K_{(t,r)}$ conflict with (t,r) and the minimal time span between (t',r') and any other trainroute is at least as large as the minimal time span between (t,r) and that other trainroute. In the following lemma, it is shown that trainroute (t,r) can be pruned, i.e., removed from $T \times R_t$, whenever $K_{(t,r)}$ is not empty.

Lemma: If $K_{(t,r)} \neq \emptyset$ for a trainroute $(t,r) \in T \times R_t$, then (t,r) is dominated by each element of $K_{(t,r)}$ or (t,r) can never be part of a feasible routing solution.

proof: Take an arbitrary $(t',r') \in K_{(t,r)}$. If $t = t'$ such that $r, r' \in R_t$, then from (4), it follows that if (t,r') conflicts with a $(\bar{t},\bar{r}) \in T \times R_{\bar{t}}$, then (t,r) also conflicts with that trainroute. Moreover, the weights in the spreading objective function (2) for any feasible solution using (t,r) will never be smaller than when using route r' instead of r for train t. This means that

Table 1: Minimal time span and the set K for each trainroute

	(t_1, r_1)	(t_1, r_1')	(t_2, r_2)	(t_2, r_2')
(t_1, r_1)	-	0	2	3
(t_1, r_1')	0	-	3	3
(t_2, r_2)	2	3	-	0
(t_2, r_2')	3	3	0	-
K	$\{(t_1, r_1')\}$	$\emptyset$	$\{(t_2, r_2')\}$	$\emptyset$

(t, r) is dominated by (t, r') since the latter always gives at least as good solutions. If $t \neq t'$ then for all $\bar{r} \in R_{t'}$ holds that $B_{(t,r),(t',\bar{r})} \leq B_{(t',r'),(t',\bar{r})} = 0$. This means that (t, r) conflicts with all possible routes for train t', thus (t, r) can never be part of a feasible solution. □

An example will illustrate this lemma. Let t_1 and t_2 be two trains, $r_1, r_1' \in R_{t_1}$ and $r_2, r_2' \in R_{t_2}$. The minimal time span between any two of these trainroutes and the sets K are given in Table 1. From this table, we see that the minimal time span between (t_1, r_1) and (t_1, r_1') is 0 since both are trainroutes of the same train, and that $B_{(t_1,r_1),(t_2,r_2')} = 3$. For each of the trainroutes, the set K can be computed using (4). As $K_{(t_1,r_1)}$ and $K_{(t_2,r_2)}$ are nonempty, these trainroutes are dominated and can be removed from the set of trainroutes. If (t_1, r_1) is pruned first, then the corresponding row and column in Table 1 must be removed causing $K_{(t_2,r_2')}$ to become nonempty. As a consequence (t_2, r_2) and (t_2, r_2') dominate each other. In this case, only one of the two is pruned causing the other to become non-dominated.
This example shows that a set of trainroutes has no unique maximal subset of non-dominated trainroutes. Analogous to the discussion in [42], it can be shown that the cardinality of a maximal subset of non-dominated trainroutes is independent of the dominance order. One can also show that the dominance rule is transitive and that pruning a dominated trainroute does not affect the dominance of another distinct trainroute.
During test runs, an average of 85% of all the trainroutes were pruned using this lemma. In total, the pruning in step 1 and the dominance rule of this lemma, reduced the number of trainroutes with 95%. As a result, only 141 trainroutes remained for an instance with 84 trains.

Step 3: The last step of the route choice module is to solve the train

routing problem on the set of non-dominated trainroutes. Therefore, define for all trainroutes $(t, r) \in T \times R_t$

$$x_{(t,r)} = \begin{cases} 1 & \text{if route } r \text{ is selected for train } t, \\ 0 & \text{otherwise.} \end{cases}$$

We also define the continuous variable $y_{(t,r),(t',r')}$, bounded by 0 and 1, as the variable that is 1 if both $x_{(t,r)}$ and $x_{(t',r')} = 1$. Now, the train routing problem can be solved with a linear mathematical model.

$$\text{Minimize} \sum_{t \in T} \sum_{t' \in T} \sum_{r \in R_t} \sum_{r' \in R_{t'}} W_{(t,r),(t',r')} \cdot y_{(t,r),(t',r')} \tag{5}$$

subject to

$$y_{(t,r),(t',r')} - x_{(t,r)} - x_{(t',r')} \geq -1 \qquad \forall (t,r) \in T \times R_t, (t',r') \in T \times R_{t'}, \tag{6}$$

$$\sum_{r \in R_t} x_{(t,r)} = 1 \qquad \forall t \in T, \tag{7}$$

$$x_{(t,r)} + \sum_{\substack{r' \in R_{t'}: \\ B_{(t,r),(t',r')}=0}} x_{(t',r')} \leq 1 \qquad \forall (t,r) \in T \times R_t, t' \in T \setminus \{t\}, \tag{8}$$

$$0 \leq y_{(t,r),(t',r')} \leq 1 \qquad \forall (t,r) \in T \times R_t, (t',r') \in T \times R_{t'}, \tag{9}$$

$$x_{(t,r)} \in \{0,1\} \qquad \forall (t,r) \in T \times R_t. \tag{10}$$

The spreading objective function (5) uses the weights from (2) and minimizes the sum of all the weights. Due to the decreasing impact of larger time spans, priority is given to avoiding small time spans. Constraint set (6) gives a lower bound on the continuous decision variable y and keeps the model linear. Since the objective is to minimize a nonnegative cost times the variable y, there is an incentive to set $y_{(t,r),(t',r')} = 0$. When both $x_{(t,r)}$ and $x_{(t',r')}$ equal 1, constraint set (6) forces $y_{(t,r),(t',r')}$ to be larger than or equal to 1. In (7) we make sure that all trains get exactly one route assigned. For each conflicting couple of trainroutes a clique constraint of the form $x_{(t,r)} + x_{(t',r')} \leq 1$ should be inserted. In our model, these clique inequalities are replaced by the stronger clique inequalities (8) which all are generated a priori. We refer to Zwaneveld et al. [41] for more details about this constraint. Finally, (9) and (10) give the bounds on the variables.

Running this model on the set of 141 non-dominated trainroutes (i.e. combinations of a train and a candidate route for that particular train) for the test case of step 2 resulted in about 20000 decision variables and more than 31000 constraints. Note that the solver is able to prune many variables and constraints right away because of the scarcity of the conflict matrix. As a result, solving this model required only 5 seconds on a DELL Optiplex 760 with Intel(R) Core(TM) 2 Duo 3.00 GHz, 4.00 GB RAM, 64-bit operating system running Cplex 12.4. Note that without the preprocessing phase, the solution for this instance cannot be found within one hour.

The mathematical model (5)-(10) can be adapted to cope with the objective of Caimi et al. [2] to maximize the four smallest minimal time spans. Using the same assumption as before, no timetable changes are allowed and the whole set of minimal time spans can be computed in advance, the adapted model can be solved for our case study using Cplex. Although the preprocessing rules from step 1 and step 2 still apply and building the model does not require significantly more computational effort, the computation time the solver needs to find the optimal solution increases considerably from 5 seconds to more than 1400 seconds. The largest portion of this is spent on finding an initial solution or due to an (extra) internal preprocessing step of Cplex.
Due to the nearly saturated infrastructure and the fact that the cost of a time span in our objective function is inversely proportional to its size, no improvement in the size of the smallest minimal time spans could be obtained using the objective of [2] instead of (3). Furthermore, our objective function tries to increase all time spans, thus also the time spans between one of the trains that forms the smallest minimal time span and any other train. As a consequence, the propagation of delay will, in general, be smaller for our solution than for the solution of Caimi et al. This allows us to conclude that our objective function (3) outperforms the one of Caimi et al. [2] for the case study of Brussels.

The outcome of the mathematical model (5)-(10) gives us the optimal routing for the current timetable. For a modified timetable, however, this routing can become suboptimal. That is why, in our methodology, the route choice module alternates with the timetabling module.

Algorithm 1 Outline of the tabu search algorithm of the timetabling module

while number of consecutive globally non-improving moves $\leq \text{iter}^{\max}$ **do**

1. Build up the restricted candidate list using ϑ
2. Perform a full neighborhood search of the shift neighborhood
3. **if** improving move found **do** update tabu list and make the move **else**
   $\vartheta \leftarrow \vartheta + \varepsilon_{\text{step}}$
   if $\vartheta > B^{\max}$ **do** perform a smallest ascent move and $\vartheta \leftarrow 0$

4.3. Timetabling module

In this step, we want to improve the timetable using the same objective function as in the route choice module. The same time discretization of 0.1 minutes is used. Since using a mathematical model to solve the timetabling problem requires the inclusion of all the minimal time span computations, which would result in a large and complex model, we chose a heuristic to improve the timetable. As a consequence, no optimal timetable for a certain routing solution is known. The timetable from the previous iteration is used as start solution in a tabu search heuristic [15] that, one by one, tries to increase the smallest minimal time span by shifting some trains in time. If this is not possible or would result in an increase in objective function value (3), the second smallest minimal time span is considered. Notice that the arrival and departure times for all trains are bounded by a time window. During our computations, we required that all trains had their planned arrival and departure between 7 and 8 AM. This choice is motivated by the fact that the considered timetable is periodic with a period of one hour. A smaller time window can be implied for each train individually. For example, the time window for an international high speed train can be chosen such that no timetable deviations are allowed for this train.

Our algorithm has one neighborhood operator: shifting one or more trains in time. During test runs, the need for a better guiding of the heuristic became clear as frequently the algorithm got stuck due to cycling. That is why we opted for a tabu search heuristic. An outline of the tabu search algorithm is given in Algorithm 1. The idea is as follows. As long as the maximum number of consecutive globally non-improving moves ($\text{iter}^{\max}$) is not reached the algorithm executes the following steps. First it constructs a restricted candidate list (RCL) which is a list of train pairs 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.

$$\text{RCL} = \left\{ (t,t') \in T \times T \,:\, B_{t,t'} \in \left[\min_{\tilde{t},\bar{t}} B_{\tilde{t},\bar{t}} + \vartheta \,,\, \min_{\tilde{t},\bar{t}} B_{\tilde{t},\bar{t}} + \vartheta + \varepsilon_{\text{step}} \right) \right\}.$$

Notice that since each train has a unique route in the timetabling module, we can simplify the notation of the time spans by dropping the routing indices. In the following, we assume t precedes t' when denoting (t,t').

In the second step, a shift move is considered for each RCL-candidate and the best found move is selected as candidate move. A shift move consists of the shift of one or more trains in time with $\delta_t = -\delta^{\max}, -\delta^{\max}+1, \ldots, -1$ or $\delta_{t'} = 1, 2, \ldots, \delta^{\max}$. Since we assume that t precedes t', a shift move increases $B_{t,t'}$. 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 $\hat{t}$, or make $B_{\hat{t},t} < B_{t,t'}$, $\hat{t}$ also gets shifted with δ_t. Notice that we only allow one of the two trains (t or t') to be shifted and that shifts that are tabu are not considered except when they are globally improving (aspiration).

In step 3, the result from the shift operator is evaluated. If the best found move was improving and not tabu (or tabu but globally improving), the move is made, ϑ is set back to 0, and the tabu lists are updated. This means that, for all shifted trains, a reverse shift becomes forbidden for a fixed number of iterations equal to *tabutime*. If the best found move was not improving, the variable ϑ is increased with $\varepsilon_{\text{step}}$ which affects the RCL in the next iteration. If $\vartheta > B^{\max}$, with $B^{\max}$ the smallest size for a time span to be considered insensitive to conflicts, no improvements will be found by increasing ϑ and a smallest ascent move is made. This means that the solution which is the best found, non-tabu solution over the last iterations, is selected as next solution to continue the algorithm.

To determine the optimal values of the parameters, a large set of experiments are conducted in which the value of the parameters could vary and based on the objective function value (primary concern) and the computation time (secondary importance) the following values are selected. $B^{\max} = 15$ minutes, meaning that two trains that have a minimal time span of 15 minutes or larger are considered to be insensitive to propagate delay to one another. The width of the RCL-interval ($\varepsilon_{\text{step}}$) is set to 0.5 minutes and the *tabutime* = 10 iterations. The maximum shift size ($\delta^{\max}$) equals 5 minutes and the parameter determining the stop criterion (iter$^{\max}$) gets 200.

Note that we do not explicitly consider changes in train ordering and that a single shift move does not allow them. Due to the iterative procedure of the route choice module and the timetabling module, however, order changes can be possible as a result of multiple consecutive changes in routing and timetable. Nevertheless, we assume that this does not lead to problems on the outside of the considered area. The same holds for any change in event times. The structure of the underlying timetable will be similar to the reference timetable that serves as input for our algorithm. The reference timetable was used for many years in practice and was improved step by step over the years. As a consequence, we assume that no change in event times will cause large problems outside the considered area.
Since the shift operator is created in such a way that a timetable shift to increase the time span $B_{t,t'}$ will never cause other time spans to become smaller than $B_{t,t'}$, no conflicts can arise by applying timetable shifts. This means that the routing solution of iteration i remains feasible throughout the whole tabu search. This allows us to use the objective function value obtained by the timetabling module as an upper bound for solving the train routing problem in iteration $i + 1$. If a (local) optimum is reached for both the route choice module and the timetabling, computations are stopped. For all test cases, a local optimum was reached in less than five iterations. Once this happened, we consider the planning of trains in the station area to be optimized and the new system can be evaluated in detail by the simulation model, described in the next section.

4.4. Simulation module

To assess the robustness of the new train system in the station area, a discrete event simulation model is developed. Our simulation model utilizes detailed infrastructure data and real-time rescheduling or rerouting is not considered. The results are computed based on 10000 simulation runs. Next to the average knock-on delay and the percentage of delayed trains, the robustness value of the system is computed.
Delays are allocated to some of the trains to represent an initial disturbance. Together with the route choice solution, the timetabling solution and the infrastructure details, this serves as input for the simulation module. Two types of delays are considered: delays upon arrival at the border of the considered area and dwell delays at any of the three stations. We denote the input delay with a four-tuple representing the delay upon arrival, the dwell delay at the South, Central and North station, respectively. Each of these

delays is drawn from the exponential distribution with the –for that train– measured real average delay ($\bar{D}$) as parameter or is equal to a predetermined value. The former we denote with E from exponential, the latter with $P^{(\text{size})}$ from predetermined. In both cases, the number of delayed trains is added as index. Let n be the number of trains, then $\left(E_{n/2}, 0, P_{n/2}^{(0.5)}, 0\right)$ means that half of the trains gets a delay upon arrival which is drawn from the exponential distribution and half of the trains gets a dwell delay at the Central station which is equal to 0.5 minutes for all these trains.
If two trains request the same infrastructure simultaneously, priority rules based on the real practice are applied to resolve this. According to these rules, priority is given to trains based on their commercial type, e.g., international trains are handled before local trains. If applicable, we restrict the number of trains within the bottleneck area by prioritizing trains that are about to leave the system compared to trains that want to enter the bottleneck area and are waiting in a –so called– buffer or compensation zone. Based on the observation that a delayed train can be overtaken by another train outside the considered area, e.g., at a previous station, we allow deviations for the planned arrival sequence at the border of the system.
In [29], it is argued that no connections are scheduled in the bottleneck area of Schiphol in the Netherlands. Doing so, synchronization actions that slow down the traffic through the bottleneck are avoided. In Brussels, there is a high frequency of trains in each direction such that the extra waiting time caused by a missed transfer remains limited. As a consequence, no transfers are scheduled and no transfer data for the Brussels' area is recorded. For our case study, we use the same principle and do not consider connections in the timetabling and the simulation module. It should be noted, however, that a list of guaranteed transfers could easily be taken into account in the timetabling module. If one of the two trains involved in a transfer is considered for a shift that would make this transfer infeasible, the other train involved in the transfer should also be shifted.

5. Experimental results

5.1. Impact of the route choice module and timetabling module

In Table 2 and 3, the most relevant results are given for two different delay scenarios. In both delay scenarios, half of the trains gets an exponential delay upon arrival. To obtain the results from Table 2, also a dwell delay at

the Central station of half a minute is given to half of the trains. In delay scenario $\left(E_{n/2}, E_{n/2}, 0, E_{n/2}\right)$ (Table 3), this dwell delay is replaced by an exponential dwell delay at the two outer stations.
The rows in the tables correspond to the instances solved. The row *reference* corresponds to the results obtained for the formerly used routing and timetable for which spreading was not an issue. This timetable, which is used as input timetable for the route choice module and the timetabling module, is used as benchmark. The impact of only applying the route choice module (timetabling module), so without the timetable (route choice) optimization, is summarized in the row *route choice* (*timetable*). The *iterative* row contains the results obtained after the iterative procedure of the route choice module and the timetabling module has reached a local optimum. This happened after three iterations of the route choice module and timetabling module. At the end 48 out of 84 trains were shifted with 2 minutes on average.
The first two columns present the values of the *spreading* objective function (3) and the computation *time* (in seconds) needed to compute the specified timetable and routing. The algorithm is run on a DELL Optiplex 760 with Intel(R) Core(TM) 2 Duo 3.00 GHz, 4.00 GB RAM, 64-bit operating system running Cplex 12.4 and is coded in C++. The percentages in the first column are relative with respect to the reference situation which is set equal to 100%. Since these values are independent of the delay scenario, these columns are the same in Table 2 and 3. Next, the results for the *robustness* (1) and the average weighted real *travel time* per passenger, see Section 3, are given. The entry in the first row of the column travel time is the average weighted real travel time per passenger, the other rows contain relative values. Further, the average amount of *knock-on* delay per hour is shown. The entry in the reference row represents the size of the total knock-on delay in this situation. The 93.6% in the row route choice in Table 2 corresponds to 57.4 (= 93.6% of 61.3) minutes of knock-on delay. Finally, the percentage of trains that are confronted with knock-on delay during their passage through Brussels (*extra delayed*) are presented. Each value in this column must be interpreted as an absolute term.
Comparing the first rows in Table 2 and 3, it can be seen that the route choice module alone as well as the timetabling module alone improves the current situation but that the impact on the results is limited. By combining the route choice module with the timetabling module in an iterative way, better results are achieved. For all criteria, the entries in the iterative row are better than those in the route choice or timetable row and considerably

Table 2: Simulation results for delay scenario $\left(E_{n/2}, 0, P_{n/2}^{(0.5)}, 0\right)$

	spreading	time (s)	robustness	travel time	knock-on	extra delayed
reference	100%	-	100%	11.6 min	61.3 min	45.2%
route choice	79.4%	4.7	101.1%	99.4%	93.6%	41%
timetable	63.2%	6.4	101.3%	99.3%	92.1%	40.8%
iterative	**38.7%**	**30**	**105.8%**	**97%**	**75.1%**	**33.6%**

Table 3: Simulation results for delay scenario $\left(E_{n/2}, E_{n/2}, 0, E_{n/2}\right)$

	spreading	time (s)	robustness	travel time	knock-on	extra delayed
reference	100%	-	100%	12.4 min	68.3 min	47.6%
route choice	79.4%	4.7	101%	99.5%	93.8%	43.1%
timetable	63.2%	6.4	100.7%	99.6%	94.5%	44.1%
iterative	**38.7%**	**30**	**105.3%**	**97.1%**	**75.7%**	**35.8%**

Table 4: Summary of the robustness results for different delay scenarios

scenario	$\left(E_{n/2}, 0, P_{n/2}^{(0.5)}, 0\right)$	$\left(E_{n/2}, 0, P_{n/2}^{(1)}, 0\right)$	$\left(E_{n/2}, 0, E_{n/2}, 0\right)$	$\left(E_{n/2}, 0, P_{3n/4}^{(0.5)}, 0\right)$	$\left(E_{n/2}, 0, P_{3n/4}^{(1)}, 0\right)$	$\left(E_{n/2}, 0, E_{3n/4}, 0\right)$
robustness	105.8%	106.8%	106.8%	106.4%	107.4%	107.3%
scenario	$\left(E_{n/3}, 0, P_{n/2}^{(0.5)}, 0\right)$	$\left(E_{n/2}, 0, P_{n/2}^{(0.5)}, 0\right)$	$\left(E_{3n/4}, 0, P_{n/2}^{(0.5)}, 0\right)$	$\left(E_{n/3}, 0, P_{3n/4}^{(1)}, 0\right)$	$\left(E_{n/2}, 0, P_{3n/4}^{(1)}, 0\right)$	$\left(E_{3n/4}, 0, P_{3n/4}^{(1)}, 0\right)$
robustness	106.5%	105.8%	105.4%	108.6%	107.4%	106.1%
scenario	$\left(P_{n/3}^{(\tilde{D})}, 0, P_{n/2}^{(0.5)}, 0\right)$	$\left(P_{n/2}^{(\tilde{D})}, 0, P_{n/2}^{(0.5)}, 0\right)$	$\left(P_{3n/4}^{(\tilde{D})}, 0, P_{n/2}^{(0.5)}, 0\right)$	$\left(P_{n/3}^{(\tilde{D})}, 0, E_{n/2}, 0\right)$	$\left(P_{n/2}^{(\tilde{D})}, 0, E_{n/2}, 0\right)$	$\left(P_{3n/4}^{(\tilde{D})}, 0, E_{n/2}, 0\right)$
robustness	106.6%	105.5%	104.5%	107.7%%	106.1%	104.8%
scenario	$(E_{n/3}, E_{n/2}, 0, E_{n/2})$	$(E_{n/2}, E_{n/2}, 0, E_{n/2})$	$(E_{3n/4}, E_{n/2}, 0, E_{n/2})$	$(E_{n/3}, E_{n/3}, E_{n/3}, E_{n/3})$	$(E_{n/2}, E_{n/3}, E_{n/3}, E_{n/3})$	$(E_{3n/4}, E_{n/3}, E_{n/3}, E_{n/3})$
robustness	106.3%	105.3%	104.3%	107.1%	105.9%	105.4%

Table 5: The amount of knock-on delay (in minutes) for each subarea in delay scenario $\left(E_{n/2}, 0, P_{n/2}^{(0.5)}, 0\right)$

	North	North-Central	Central	Central-South	South	total
reference	15.5	6	18.5	11.2	10.1	61.3
route choice	15.4	4.3	18.3	9.2	10.1	57.3
timetable	15.6	5.3	15.7	9.8	10	56.4
iterative	**13.5**	**3.4**	**13.4**	**7**	**8.7**	**46**

improve the results of the reference situation. The improvement of more than 60% in the spreading objective value is remarkable. Also the effect on the robustness of using the algorithms from Section 4 is significant. Next, the results indicate that the total weighted real travel time decreases with about 3%; the amount of knock-on delay has decreased with nearly 25%; while the number of extra delayed trains is reduced with 25.7% ($= 1 - 33.6/45.2$) and 24.8%, respectively.

In Table 4, the obtained *robustness* values for the iterative timetable are given for different delay *scenarios*. In the delay scenarios on the top row, half of the trains gets an exponential delay upon arrival while different dwell delays are considered. The two rows in the middle of the table contain delay scenarios with different numbers of trains that get an exponential or predetermined delay upon arrival at the system's border, and the lowest part considers the impact of dwell delays at the outer stations. From the results in Table 4, we see that the average gain in robustness is about 6.2% with a standard deviation of 1%. Considering the fact that neither the timetabling, nor the route choice module uses passenger flows or optimizes with respect to delays directly, which both are criteria that determine the robustness, the strength of these methods is even more striking.

Another important conclusion based on the evaluation of all these delay scenarios is that the obtained results are very similar. The standard deviation of 1.04% supports the conclusion that the performance of the obtained iterative timetable, measured through the robustness, is rather stable and independent of the delay scenario.

In Table 5, the amount of knock-on delay that emerges due to conflicts in a station or on the switch zones between two stations is given for the same delay scenario as in Table 2. Based on the infrastructure from Figure 1, five subareas are identified: the three stations (*North*, *Central*, and *South*) and the two switch zones connecting these. Each entry is the amount of knock-on delay, in minutes, that emerges in each of the corresponding zones. The entries for each subarea sum to the *total* amount of knock-on delay per hour. The largest numbers in Table 5 correspond to the stations. For the two outer stations, North and South, this is because the merging of the incoming paths happens there. The knock-on delay that arises normally on the incoming lines is now added to the outer stations since this is where the trains first meet in our case study. It is not surprising that the Central station is a large source of knock-on delay as the available capacity is nearly saturated there. Although the amount of knock-on delay is smaller for the switch zones, their impact should not be underestimated. A large number of merging and intersecting actions takes place in these switch zones and thus there is a large threat on conflicts.

The results in Table 5 confirm the reasoning that only changing the routing of trains through a station does not directly help to avoid conflicts in that station, but it does help to reduce the amount of knock-on delay in the switch zones: A reduction of 27.7% and 18.2% for the zones North-Central

and Central-South, respectively, is achieved. Although the timetabling module only alters the arrival and departure times in the stations, the usage of the spreading objective function (3) ensures improvement in nearly all zones. Comparing the reference and iterative rows, it can be seen that, due to the iterative procedure of the algorithms from Section 4, the combination of the route choice and timetabling module helps to reduce the effect of conflicts in each of the zones considerably.

5.2. Improvement measures to evaluate

The Belgian railway infrastructure manager Infrabel requested us to suggest some potential measures to improve the robustness. That is why, next to improving the current situation with the algorithms from Section 4, three specific measures to further enforce the robustness are investigated. One of the major problems in the Brussels' area is that nearly all trains that run through Brussels dwell at one of the six platforms of the Central station. The three measures we suggest are chosen because each of them helps to decrease the capacity usage or to increase the available capacity in the Central station. After the measures are taken, our algorithm is used to reschedule (improve) the system. The idea is to make a "*what if*"-study by estimating the impact of this measure. We do not make claims about optimality, e.g., whether the number and the selection of removed trains (measure 1) is the best possible or not.

The first measure consists of removing some trains from the system. In our timetable, 84 trains run through Brussels between 7 and 8 AM. With this measure, we want to estimate the impact of reducing the train offer by 4 (measure 1_a), 6 (1_b) and 8 (1_c) trains. The removed trains are selected based on their (low) occupation rate and the passengers on any of these trains are, without an extra cost, assigned to alternative trains.

The second measure keeps the number of trains constant but adapts the stopping pattern. More specific, the trains that are supposed to dwell at platform 1 and 2 in the Central station become non-stopping. We assume that harmed passengers transfer to other trains that do stop at the Central station and experience a travel time extension of 3 minutes. The other passengers on a non-stopping train gain one minute in travel time because of the avoided stop. Changing the stopping pattern in this way results in a gain in the size of the total headway buffer for platform 1 and 2.

The third measure focuses on the infrastructure in the Central station. The idea is to increase the number of platforms from 6 to 10 by doubling the

Table 6: Simulation results for delay scenario $\left(E_{n/2}, 0, P_{n/2}^{(0.5)}, 0\right)$

	spreading	time (s)	robustness	travel time	knock-on	extra delayed
iterative	38.7%	30	105.8%	97%	75.1%	33.6%
measure 1_a	32.3%	26.4	108.9%	95.5%	63.6%	30%
measure 1_b	28.9%	27.1	112.1%	93.8%	55.3%	27.2%
measure 1_c	27.3%	19.4	113.1%	93.2%	51.7%	26.1%
measure 2	37.5%	19	103.3%	101.6%	74%	32.5%
measure 3	35.7%	23.5	108.2%	95.8%	68.5%	31.1%

Table 7: Simulation results for delay scenario $\left(E_{n/2}, E_{n/2}, 0, E_{n/2}\right)$

	spreading	time (s)	robustness	travel time	knock-on	extra delayed
iterative	38.7%	30	105.3%	97.1%	75.7%	35.8%
measure 1_a	32.3%	26.4	108.7%	95.3%	63.7%	32.1%
measure 1_b	28.9%	27.1	110.4%	94.4%	56.4%	29.3%
measure 1_c	27.3%	19.4	111.3%	93.7%	51.7%	28%
measure 2	37.5%	19	103.6%	101.3%	74.7%	34.7%
measure 3	35.7%	23.5	106.5%	96.5%	72.7%	34%

Table 8: The amount of knock-on delay (in minutes) for each subarea in delay scenario $\left(E_{n/2}, 0, P_{n/2}^{(0.5)}, 0\right)$

	North	North-Central	Central	Central-South	South	total
iterative	13.5	3.4	13.4	7	8.7	46
measure 1_a	12.3	3	11.1	5.7	6.9	38.9
measure 1_b	11.5	2.7	10	4.5	5.2	33.9
measure 1_c	10.9	2.4	8.8	4.4	5.2	31.7
measure 2	14.9	3.6	11.8	6.5	8.6	45.3
measure 3	15.1	4.3	6.5	7.2	8.7	41.9

two outer platforms at each side. As a consequence, a train that is currently planned to dwell at platform 1, can now be assigned to platform 1 or 1'. If the trains are ordered chronologically, the odd numbered trains get assigned to the old platform and the even numbered ones to the new platform. No other changes than the ones specified are being made for any of the three measures and the methodology for Section 4 is used to improve the timetables after the changes due to the measures are applied.

The impact of these measures is shown in Table 6 and 7. The results are ob-

tained using the methodology of Section 4 applied on the reference timetable that is modified by the measure itself. Since removing some trains from the system leads to a decrease in capacity usage for the whole area while the other two measures only affect the Central station, it is not surprising that the first measure outperforms the others. Table 8 clearly confirms this, where measure 1 reduces the amount of knock-on delay for each subarea, the other two measures perform more or less similar to the iterative timetable, except for the Central station where measure 3 outperforms the other two measures. Since measure 3 affects four of the six tracks in the Central station and measure 2 only two, it is not surprising that the third measure leads to (locally) better results than measure 2.
Dividing the knock-on delay per hour by the number of trains, the average amount of knock-on delay per train per hour is obtained. Doing this for the results in Table 6, we get that for the iterative situation the average knock-on delay per train per hour equals 0.89%(= 75.1%/84) of the total delay in the reference situation while for measure 1 (a, b and c) this becomes 0.80%(= 63.6%/80), 0.71%, and 0.68%, respectively. Thus, having fewer trains not only leads to less knock-on delay in total and thus also a smaller weighted real travel time, but also causes less knock-on delay per train. We emphasize once more that we do not claim to have found the optimal set of trains or its size, but just use this measure to show that removing some trains from the system helps to improve the robustness. The decreasing marginal impact of going from 4 to 6 or from 6 to 8 removed trains, however, shows that this improvement is limited.

In contrast with the other rows, the weighted real travel time has increased for measure 2. This is due to an increase in nominal travel time as a consequence of the required transfers for the passengers that want to end up in the Central station. Thus the advantage for the passengers who gain by avoiding the stop is smaller than the disadvantage for the passengers with the Central station as destination. However, considering the other performance indicators, we see that the gain in buffer time due to this measure affects the amount of knock-on delay and the percentage of extra delayed trains positively. Thus, we can conclude that this measure turns out to be good for the punctuality of the trains but not when considering the passengers' real travel time.

6. Conclusion

In this paper, a new way to improve the robustness in railway bottlenecks is introduced. Based on a route choice module for solving the train routing problem and a timetabling module, we significantly increase the minimal time span between any two trains in a station area. As the modifications suggested by the presented algorithm require only timetable changes and route changes within the station, the cost for the railway operator and the railway infrastructure manager is very small. Moreover, this paper presents a way to improve the perception of the passengers' real travel time, which is an important quality measure. Computational results for the Brussels' area, the largest bottleneck in the Belgian railway network, predict that by using the iterative procedure of the route choice module and the timetabling module an average gain in robustness of 6.2% (with a standard deviation of 1.04%) can be achieved. This improvement is due to a decrease in knock-on delay of about 25% for some delay scenarios and a reduction in the number of trains that are hindered during their passage through the Brussels' area of, again, 25%.
The methodology that is introduced in this paper is also used to assess the impact of three possible measures. From our study, we can conclude that applying our algorithm to the system with a slightly reduced train offer, results in an increased level of robustness up to 13.1%, a remarkable decrease in knock-on delay of nearly 50%, and a reduction in the number of extra delayed trains of more than 70%.
Some thoughts for further research are: including the passenger flows and potential delays in the route choice module and the timetabling module to further increase the performance of the algorithms considering robustness. Some other ideas are to allow platform changes or small modifications to the timetable during the route choice module or to explicitly incorporate changes in the order of trains during the timetable optimization phase without losing feasibility.

Acknowledgements

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

References

[1] Cacchiani, V., Toth, P., 2012. Nominal and robust train timetabling problems. European Journal of Operational Research 219, 727–737.

[2] Caimi, G., Burkolter, D., Herrmann, T., 2005. Finding delay-tolerant train routings through stations. In: Fleuren, H., Hertog, D., Kort, P. (Eds.), Operations Research Proceedings. Vol. 2004. Springer-Verlag, pp. 136–143.

[3] Caimi, G., Burkolter, D., Herrmann, T., Chudak, F., Laumanns, M., 2009. Design of a railway scheduling model for dense services. Networks and spatial Economics 9, 25–46.

[4] 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. Transportation Science 45, 212–227.

[5] Caprara, A., Kroon, L., Monaci, M., Peeters, M., Toth, P., 2007. Passenger railway optimization. In: Barnhart, C., Laporte, G. (Eds.), Handbooks in Operations Research and Management Science. Vol. 14. Elsevier, pp. 129–187.

[6] Carey, M., Crawford, I., 2007. Scheduling trains on a network of busy complex stations. Transportation Research Part B: Methodological 41, 159–178.

[7] Corman, F., D'Ariano, A., Pacciarelli, D., Pranzo, M., 2010. A tabu search algorithm for rerouting trains during rail operations. Transportation Research Part B: Methodological 44, 175–192.

[8] Corman, F., Goverde, R., D'Ariano, A., 2009. Rescheduling dense train traffic over complex station interlocking areas. In: Ahuja, R., Möhring, R., Zaroliagis, C. (Eds.), Robust and Online Large-Scale Optimization. Vol. 5868 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 369–386.

[9] D'Ariano, A., Corman, F., Pacciarelli, D., Pranzo, M., 2008. Reordering and local rerouting strategies to manage train traffic in real time. Transportation Science 42, 405–419.

[10] D'Ariano, A., Pacciarelli, D., Pranzo, M., 2007. A branch and bound algorithm for scheduling trains in a railway network. European Journal of Operations Research 183, 643–657.

[11] D'Ariano, A., Pacciarelli, D., Pranzo, M., 2008. Assessment of flexi-

ble timetables in real-time traffic management of a railway bottleneck. Transportation Research Part C: Emerging Technologies 16, 232–245.

[12] 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).

[13] Fischetti, M., Monaci, M., 2009. Light robustness. In: Ahuja, R., Möhring, R., Zaroliagis, C. (Eds.), Robust and Online Large-Scale Optimization. Vol. 5868 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 61–84.

[14] Fischetti, M., Salvagnin, D., Zanette, A., 2009. Fast approaches to improve the robustness of a railway timetable. Transportation Science 43, 321–335.

[15] Gendreau, M., 2003. An introduction to tabu search. In: Glover, F., Kochenberger, G., Hillier, F. (Eds.), Handbook of Metaheuristics. Vol. 57 of International Series in Operations Research & Management Science. Springer New York, pp. 37–54.

[16] Goerigk, M., Schöbel, A., 2010. An empirical analysis of robustness concepts for timetabling. In: Erlebach, T., Lübbecke, M. (Eds.), Proceedings of ATMOS2010. Vol. 14 of OpenAccess Series in Informatics (OASIcs). Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, pp. 100–113.

[17] Goverde, R., 2007. Railway timetable stability analysis using max-plus system theory. Transportation Research Part B: Methodological 41, 179–201.

[18] Huisman, D., Kroon, L., Lentink, R., Vromans, M., 2005. Operations research in passenger railway transportation. Statistica Neerlandica 59, 467–497.

[19] Kroon, L., Huisman, D., Maroti, G., 2008. Optimisation models for railway timetabling. Railway Timetable & Traffic: Analysis, Modelling and Simulation. Eurailpress, Hamburg, pp. 135–154.

[20] Kroon, L., Maroti, G., 2008. Robust train routing. Technical report TR-0123, ARRIVAL.

[21] Liebchen, C., Lubbecke, M., Möhring, R., Stiller, S., 2009. The concept of recoverable robustness, linear programming recovery, and railway applications. In: Ahuja, R., Möhring, R., Zaroliagis, C. (Eds.), Robust and Online Large-Scale Optimization. Vol. 5868 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 1–27.

[22] Liebchen, C., Schachtebeck, M., Schöbel, A., Stiller, S., Prigge, A., 2010.

Computing delay resistant railway timetables. Computers & Operations Research 37, 857–868.

[23] Lusby, R., Larsen, J., Ehrgott, M., Ryan, D., 2011. Railway track allocation: models and methods. OR Spectrum 33, 843–883.

[24] Lusby, R., Larsen, J., Ryan, D., Ehrgott, M., 2011. Routing trains through railway junctions: A new set-packing appraoch. Transportation Science 45, 228–245.

[25] Mackie, P., Jara-Díaz, S., Fowkes, A., 2001. The value of travel time savings in evaluation. Transportation Research Part E: Logistics and Transportation Review 37, 91–106.

[26] Mazzarello, M., Ottaviani, E., 2007. A traffic management system for real-time traffic optimisation in railways. Transportation Research Part B: Methodological 41, 246–274.

[27] Meng, L., Zhou, X., 2011. Robust single-track train dispatching model under a dynamic and stochastic environment: A scenario-based rolling horizon solution approach. Transportation Research Part B: Methodological 45, 1080–1102.

[28] Salido, M., Barber, F., Ingolotti, L., 2012. Robustness for a single railway line: Analytical and simulation methods. Expert Systems with Applications 39, 13305–13327.

[29] Schaafsma, A., Bartholomeus, M., 2007. Dynamic traffic management in the schiphol bottleneck. In: Proceedings of 2nd International Seminar on Railway Operations Modelling and Analysis (RailHannover).

[30] Schöbel, A., Kratz, A., 2009. A bicriteria approach for robust timetabling. In: Ahuja, R., Möhring, R., Zaroliagis, C. (Eds.), Robust and Online Large-Scale Optimization. Vol. 5868 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 119–144.

[31] Sels, P., Dewilde, T., Cattrysse, D., Vansteenwegen, P., 2011. 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).

[32] Sels, P., Dewilde, T., Vansteenwegen, P., Cattrysse, D., 2011. Automated, passenger time optimal, robust timetabling, using integer programming. In: Proceedings of 1st International Workshop on High-speed and Intercity Railways (IWHIR 2011).

[33] Shafia, M., Aghaee, M., Sadjadi, S., Jamili, A., 2012. Robust train timetabling problem: Mathematical model and branch and bound algorithm. IEEE Transactions on Intelligent Transportation Systems 13,

307–317.

[34] Snelders, M., Immers, B., Wilmink, I., 2004. De begrippen betrouwbaarheid en robuustheid nader verklaard (in Dutch). In: Colloquium Vervoersplanologisch Speurwerk. Zeist, The Netherlands.

[35] Stynen, S., August 2010. Reizigerstellingen 2009 (online). http://www.treintrambus.be/actueel/blog/1216-opstapcijfers.html.

[36] Takeuchi, Y., Tomii, N., Hirai, C., 2007. Evaluation method of robustness for train schedules. Quarterly Report of Railway Technical Research Institute 48, 197–201.

[37] Vansteenwegen, P., Van Oudheusden, D., 2006. Developing railway timetables which guarantee a better service. European Journal of Operational Research 173, 337–350.

[38] Vansteenwegen, P., Van Oudheusden, D., 2007. Decreasing the passenger waiting time for an intercity rail network. Transportation Research Part B: Methodological 41, 478–492.

[39] Vromans, M., Dekker, R., Kroon, L., 2006. Reliability and heterogeneity of railway services. European Journal of Operational Research 172, 647–665.

[40] Wardman, M., 2004. Public transport values of time. Transport Policy 11, 363–377.

[41] Zwaneveld, P., Kroon, L., Romeijn, H., Salomon, M., Dauzère-pérès, S., Van Hoesel, S., Ambergen, H., 1996. Routing trains through railway stations: Model formulation and algorithms. Transportation Science 30, 181–194.

[42] Zwaneveld, P., Kroon, L., van Hoesel, S., 2001. Routing trains through a railway station based on a node packing model. European Journal of Operations Research 128, 14–33.