Netw Spat Econ (2009) 9:63–84
DOI 10.1007/s11067-008-9088-1

An Advanced Real-Time Train Dispatching System for Minimizing the Propagation of Delays in a Dispatching Area Under Severe Disturbances

Andrea D'Ariano · Marco Pranzo

Published online: 15 November 2008

Abstract In highly utilized rail networks, as in the Netherlands, conflicts and subsequent train delays propagate considerably in time and space during operations. In order to realistically forecast and minimize delay propagation, there is a need to extend short-term traffic planning up to several hours. On the other hand, as the magnitude of the time horizon increases the problem becomes computationally intractable and hard to tackle. In this paper, we decompose a long time horizon into tractable intervals to be solved in cascade with the objective of improving punctuality. We use the ROMA dispatching system to pro-actively detect and globally solve conflicts on each time interval. The future evolution of railway traffic is predicted on the basis of the actual track occupation, the Dutch signaling system and dynamic train characteristics. Extensive computational tests are carried out on the railway dispatching area between Utrecht and Den Bosch.

Keywords DSS · Delay management · Short-term train scheduling · Temporal decomposition

A. D'Ariano (✉)
Department of Transport and Planning, Delft University of Technology,
Stevinweg 1, 2600 GA Delft, The Netherlands
e-mail: a.dariano@tudelft.nl

M. Pranzo
Dipartimento di Ingegneria dell'Informazione,
Università di Siena, via Roma 56, 53100 Siena, Italy
e-mail: pranzo@dii.unisi.it

1 Introduction

The continuous growth of railway traffic is increasing the pressure on European railway companies to provide a satisfactory level of service with limited changes to the existing infrastructure. Traditionally, issues arising in railway traffic management can be divided into at least two categories according to different time perspectives. Timetabling mainly consists of designing a robust schedule for a large traffic network, i.e., several dispatching areas, and for a large time horizon, i.e., up to a year, and its resolution can require even days of computation. On the other hand, real-time dispatching (or rescheduling) assumes the existence of a timetable and deals with delay management. This process is applied on each dispatching area to adjust the perturbed traffic flow and generate a feasible schedule of rail operations while respecting strict time limits of computation, i.e., up to few minutes.

In the railway literature, several papers deal with models and algorithms for the train scheduling problem. For an overview of the proposed approaches, we refer the reader to the comprehensive surveys of Assad (1980), Cordeau et al. (1998), Oh et al. (2004) and Törnquist (2005), and to the valuable contributions in Hansen et al. (2005). Among recent contributions related to the real-time dispatching problem, Jacobs (2004) has developed a detailed model based on the identification of possible route conflicts with high accuracy, using the blocking time theory (Pachl 2002), when the objective is to minimize additional running times. In presence of disturbances, an algorithm detects the infeasible train routes and solves each conflict locally based on train priorities. Mazzarello and Ottaviani (2007) describe the practical implementation of a real-time traffic management system that has been developed for the European project COMBINE, on a pilot site in The Netherlands. The aim is to test the feasibility of a completely automated system for conflict resolution and speed regulation. A detailed description of models and algorithms used within COMBINE is presented in Mascis et al. (2004) and Pacciarelli and Pranzo (2006). Rodriguez (2007) focuses on the resolution of train conflicts and proposes a constraint programming formulation for the compound routing and sequencing problem. Computational experiments show that a truncated branch and bound algorithm can find satisfactory solutions for a rail junction within 3 min of computation time. Törnquist and Persson (2007) present a model for rescheduling trains in a rail network with several merging and crossing points. The problem is formulated as a mixed-integer linear program and solved with commercial software packages. Four strategies are proposed for reducing the solution space based on restrictions on reordering and rerouting actions. Computational experiments are based on instances with a single delayed train.

Despite the effort devoted to developing sophisticated dispatching procedures, few decision support systems exist to help traffic controllers during operations. These systems are able to provide good solutions only for small instances or for simple perturbations. Hence, the real-time dispatching process is still mainly under the control of human dispatchers who usually do not

have precise information about the future evolution of train traffic and the chosen traffic control actions are often sub-optimal (Kauppi et al. 2006; van den Top 2006). One of the reasons for this is the complexity of finding a good compromise between the solution quality, time horizon (or time span) of the traffic prediction and computation times. In fact, in order to take informed operational decisions, the dispatchers should know the short-term consequences of their actions. An accurate prediction of the effects of delays and other disturbances require detailed modelling and reflecting the actual state of the network, both the dynamic behavior of running trains and the dispatching measures used to control traffic. Hence, the delay propagation is not predictable by traffic controllers, especially in case of complex rail networks, high density traffic and severe disturbances. For these reasons, there is a need for an automatic decision support tool to forecast delays propagation in the rail network. This tool should operate sufficiently in advance to correctly quantify the effects of different dispatching measures and would enable traffic controllers to perform frequent incremental changes to the actual timetable to accommodate changes in traffic patterns due to disturbances.

In this paper, we use a real-time dispatching system, called ROMA (Railway traffic Optimization by Means of Alternative graphs), to automatically recover disturbances. ROMA is able to automatically control traffic, evaluating the detailed effects of train reordering (D'Ariano et al. 2007a) and local rerouting (D'Ariano et al. 2006) actions, while taking into account minimum distance headways between consecutive trains and the corresponding variability of train dynamics (D'Ariano et al. 2007b; D'Ariano and Albrecht 2006).

We extend ROMA for the short-term prediction of train traffic in a dispatching area under strong disturbances. Specifically, the objective is to proactively evaluate the effects of train rescheduling actions for a time period of up to some hours within the same day. If the effects of standard delay management are deemed inadequate by the traffic controller, other drastic measures will be adopted, such as the use of emergency timetables or the cancellation of train routes.

The prediction of railway traffic can result into computationally intractable instances. In this context, Rodriguez (2000) has shown that a temporal decomposition and constraint programming approach can be used to compute suitable solutions in considerably less computation time compared to the global resolution of the problem. We also decompose the railway traffic into tractable time intervals to be solved in cascade. The coordination of each time interval is achieved by inserting the position and the speed of each train at the end of the time interval as an input constraint for the subsequent time interval and by also considering the constraints of the re-use of the same rolling stock for different train trips. The independent resolution of each hour of dispatching permit handling large time horizons within a linear increase of computation time. We compare this approach with a global formulation of the problem in order to evaluate the error due to the problem decomposition.

Our computational tests simulate the Utrecht - Den Bosch railway network, which is the control area of a human dispatcher. We study the short-term

consequences of train delays and temporary unavailability of some tracks. The effects of increasing timetable disturbances are analyzed carefully while considering several timetable hours. Furthermore, randomly generated blocked tracks are included in order to obstruct a main traffic direction and to cause a serious propagation of train delays. The disturbance propagation is measured in terms of punctuality for each successive timetable hour.

The paper is organized as follows: Section 2 defines the real-time train dispatching problem; Section 3 gives an introduction to ROMA while Section 4 describes the model extensions to tackle short-term traffic prediction. Section 5 presents the results of our computational experiments based on the dispatching area between Utrecht and Den Bosch.

2 Real-time train dispatching problem

A railway network consists of stations, links, block sections and signals. A block section is a track segment between two main signals and may host at most one train at a time. Signals and automatic train protection (ATP) control the train traffic by enforcing speed restrictions on running trains and imposing minimum safety separations between consecutive trains. Safety distance and time headways depend on the speeds of the consecutive trains, the braking rate of the second train, the train length of the first train and the signal spacing.

The passage of a train through a particular block section is called an *operation*. A *route* of a train is a sequence of operations to be traversed in a dispatching area during a *service*. Each operation requires a given *running time* which depends on the actual speed profile followed by the train while traversing the block section. A speed profile is furthermore constrained by rolling stock characteristics (maximum speed and acceleration and braking tables), physical infrastructure characteristics (maximum allowed speed and signaling system) and driver behaviors (braking and acceleration profiles when approaching variable signals aspects (D'Ariano et al. 2007b; D'Ariano and Albrecht 2006)). The minimum time separations among the running trains translates into a minimum *setup time* between the exit of a train from a block section and the entrance of the subsequent train into the same block section.

We consider a cyclic timetable which describes the movement of all trains running in the network during a given hour, specifying, for each train, planned arrival/passing times at a set of relevant points along its route (e.g., stations, junctions, and the exit point of the network). At stations, a train is not allowed to depart from a platform stop before its scheduled departure time and is considered late if arriving at the platform after its scheduled arrival time. At a platform stop, the scheduled waiting time of each train is called *dwell time*. Additional real-world constraints related to passenger satisfaction should also be considered, such as minimum transfer time between connected train services. This is the time required to allow passengers alight from one train, move to another platform and board another train. Constraints due to rolling

stock re-use must also be included while solving the problem. In fact, when dealing with short-term traffic predictions, a rolling stock can complete the round-trip of a train and be re-used as a new train, i.e., the rolling stock constraint. For this reason, railway timetables usually consider a *turning buffer time* between train routes, which is a time margin between the end of a train route and the start of a new train service using the same rolling stock. However, in case of severe disturbances, delays may propagate up to several hours and a train may be delayed because of the unavailability of its rolling stock.

Timetables are designed to satisfy all traffic regulations. However, in real-time, unexpected events occur that make the timetables infeasible. We define an *entrance perturbation* as entrance delays for trains entering the network and a *timetable disruption* as the modification of scheduled train speeds and routes, and dwell times within the network. Specifically, entrance delays are due to the delay propagation from previous dispatching areas, running time prolongation may occur because of headway conflicts between consecutive trains or technical failures, route changes are due to some block section being unavailable for a certain amount of time and dwell time perturbations are due to technical delays at station stops.

Real-time train dispatching copes with real-time infeasibilities by adjusting the timetable of each train, in terms of routing and timing, and by resequencing the trains at the entrance of each merging/crossing point. The railway traffic is predicted over a given time horizon, i.e., a given number of timetable hours. Its main goal is to minimize train delays while satisfying traffic regulations constraints and the compatibility with the real-time position of each train. The latter information enables the computation of the *release time* of each train, which is the minimum time at which the train can enter the network or reach the end of its current block section at the starting time of traffic prediction t_0. Following the notation of D'Ariano et al. (2007b), the *total delay* is the difference between the calculated train arrival time and the scheduled time at a relevant point in the network, and can be divided into two parts. The *primary delay* is caused by failures and disturbances and cannot be recovered by rescheduling train movements, except by exploiting available running time margins, i.e., trains traveling at maximum speed. The *consecutive delay* is caused by the interaction between trains running in the given time horizon of traffic prediction.

In our terminology, a *conflict* occurs when two or more trains claim the same block section simultaneously and one of the trains involved has to change its speed profile according to the constraints of the signaling system. The *blocking time* is the time interval in which a block section is exclusively allocated to a train and blocked for other trains (Pachl 2002). A *blocking time overlap* would arise if the minimum distance headway between two trains is not satisfied (D'Ariano et al. 2007b). A train speed profile is *acceptable* if this is compatible with the acceleration/braking rates usually performed by the train. A solution is therefore *feasible* if there are no conflicts between running trains, train distance headways are respected and each train has an acceptable speed profile.

The train dispatching problem can, therefore, be defined as follows: given a railway network, a set of train routes and passing/stopping times at each relevant point in the network, and the position and speed of each train at t_0 being fixed, find a conflict-free schedule with no blocking time overlaps, such that the selected train routes are not blocked, the speed profiles are acceptable, no train departs from a relevant point before its departure time, rolling stock constraints and connected train services are respected, and trains arrive at the relevant points with the smallest possible consecutive delay.

3 ROMA dispatching system

The ROMA dispatching system is designed as a decision support system for traffic controllers. Given a disturbed timetable, the real-time train dispatching problem is divided into three subproblems: (i) assigning a feasible route to each train in order to avoid blocked tracks, (ii) defining train orders and specifying the exact arrival and departure times at stations as well as at a set of relevant points in the network, such as junctions and passing points, and (iii) ensuring a minimum distance headway between trains while maintaining acceptable speed profiles. ROMA is able to solve the three subproblems automatically.

Figure 1 presents the ROMA architecture, which is composed of interrelated modules. Next, we describe the function of each module and how the three subproblems are solved.

The *load information* module periodically collects information from the field. Primary condition for calculating future train movement is the availability of a detailed and accurately updated data set. Specifically, running times, setup times and blocking times for each operation are computed in accordance with the actual speed and position of each train at its entrance of the network, the current infrastructure status (e.g. track layout, signals, speed limits), the timetable data and the rolling stock characteristics. After the completion of the loading phase, the other modules of ROMA are executed assuming that real-time variations of these data would not affect the principal validity of the rescheduling solution. In this paper, we adopt ROMA to predict railway traffic for several time horizons. The solutions obtained are thus applicable in real-

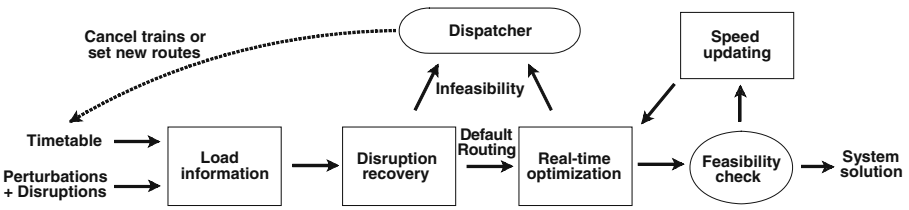

Fig. 1 ROMA dispatching system architecture

time only for those instances solved within 1 or 2 min, depending on the traffic conditions. However, the proposed dispatching system is a laboratory version tested on an off-line data set and does not include coupling to actual train monitoring data. A discussion of the necessary communication links between the running trains and the traffic control centers can be found in D'Ariano et al. (2007b).

The *disruption recovery* module checks if there are unavailable block sections in the network, rendering some route infeasible. This activity corresponds to the resolution of subproblem (i). This module discards infeasible routes from the list of prioritized routes, and assigns each train the feasible route with highest priority. If no feasible route is available for a train, the system requests external support by the human dispatcher.

The *real-time optimization* module is the decisional kernel of the traffic management system which is responsible for detecting and solving train conflicts while minimizing the propagation of delays. Given a feasible route for each train, this module defines a train scheduling that corresponds to determine the starting time of each operation. The conflict detection and resolution problem (i.e. subproblem (ii)) can be formulated as a job shop scheduling problem with no-store and other additional constraints (Mascis et al. 2004). Furthermore, Mascis and Pacciarelli (2002) showed that alternative graph is a suitable model for the job shop problem and several real-world constraints can be easily modeled by it. The real-time optimization module uses the alternative graph formulation of D'Ariano et al. (2007a) in which we formulate the train scheduling problem in first instance as a fixed-speed model. This formulation requires that a routing for each train is given and a fixed traversing time for each block section is known in advance, except for possible additional waiting time needed to solve conflicts. In case of conflicts, a passing order must be defined between the trains. A conflict-free schedule is obtained by solving an associated complex job shop problem. The main value of the alternative graph is the detailed representation of the network topology at the level of railway signal aspects and operational rules. Moreover, other constraints relevant to the railway practice can be included into the model, such as the ones described in Section 2.

Since the resolution of train conflicts has direct impact on the level of punctuality, in this paper we use two scheduling algorithms based on the alternative graph formulation:

- *Branch and Bound* (BB): This is an exhaustive algorithm that explores all the reordering alternatives and chooses the one minimizing the maximum consecutive delay. Here we consider a truncated branch and bound, introduced in D'Ariano et al. (2007a), which returns a near-optimal schedule for practical size problems within a short computation time.
- *First Come First Served* (FCFS): This is a well-known dispatching rule which gives precedence to the train arriving first at a block section. This rule requires no dispatching action since trains pass at merging or crossing

points on the basis of their actual order of arrival and not necessarily as planned in the timetable.

If the real-time optimization module is unable to find a conflict-free schedule, the dispatcher has to carry out other types of timetable modifications such as introducing new train routes, applying short-turning of trains in case of track blockage or even canceling train services at some stations.

In ROMA, the solution to subproblem (ii) presents deterministic blocking and waiting times and thus does not explicitly model the dynamic consequences of braking and subsequent acceleration imposed to avoid conflicts in the network. According to the Dutch signaling system, this means that the traversing time of each train on each block section is computed assuming green signal aspects. D'Ariano et al. (2007b) therefore present a variable speed model, based on an alternative graph formulation of the blocking time theory, that is able to solve subproblem (iii). The *feasibility check* module verifies whether or not the schedule is compatible with the actual train dynamics and signal aspects. For the Dutch railways, the latter case corresponds to the infeasible situation in which a train traverses a block section at yellow signal aspect without changing its speed profile, i.e. an overlap of blocking times in our model. The *speed updating* module therefore adjusts the train speed profile according to typical driver behaviors and to the dynamics of the rolling stock. Feasibility checks and speed updates are performed until an overlap-free schedule with acceptable speed profiles is obtained. As a result of this, the blocking times are then correctly recomputed. This dynamic control therefore coordinates the speed of successive trains on open tracks, secures the time windows at junction/crossing points and synchronizes the trains arriving at stations. In our computational experiments, we test the following ROMA configurations:

- *Iterative scheduling strategy* (D'Ariano et al. 2007b): The real-time optimization, feasibility check and speed updating modules are performed iteratively. At each iteration, the real-time optimization module solves subproblem (ii), using BB or FCFS, and calls the feasibility check and speed updating modules for the resolution of subproblem (iii). The speed profile of the "first" unacceptable train is updated and another run of the real-time optimization module is performed considering an acceptable speed profile for this train. The iterative procedure is executed a finite number of times.
- *Single scheduling strategy*: This is a simplification of the iterative scheduling strategy. The subproblems (i) and (ii) are solved in cascade. The real-time optimization module solves subproblem (ii), using BB or FCFS, and calls the feasibility check and speed updating modules until admissible speed profiles are calculated for every train. Following this strategy, the sequencing obtained by the real-time optimization module is therefore not modified during the resolution of subproblem (iii).

4 Short-term traffic prediction

We now extend ROMA to forecast the propagation of train delays up to several hours, which corresponds to a short-term train dispatching problem. This is a large-scale decision problem with several decision variables and constraints. A common operations research practice to solve such a large decision problem is to decompose it into a series of smaller problems that progressively arrive at the final solution providing a good solution to the entire problem (Ovacik and Uzsoy 1997; Ahuja et al. 1993; Pardalos and Resende 2002). Specifically, decomposition approaches have been proposed successfully to solve complex job shop scheduling problems, such as machine decomposition (Adams et al. 1988) and temporal decomposition (Chambers et al. 1991; Zeng et al. 1998).

Section 4.1 describes the alternative graph formulation adopted to solve the problem. Section 4.2 then introduces two resolution procedures. The first is a global approach based on the resolution of a very big alternative graph, while the second consists of decomposing the time horizon of traffic prediction in intervals and adopting an alternative graph for solving each interval. In Section 4.3, an explanatory example of the procedures is also given, including a graphical representation of the alternative graphs.

4.1 Alternative graph formulation

Given an operation h, we denote with $\sigma(h)$ the operation which follows h on its route. A *timing* specifies the starting time t_h of each operation. Denoting with f_h the running time of operation h, the running time constraint for operation h imposes that $t_{\sigma(h)} \geq t_h + f_h$, while the setup time constraint is respected for operation h if $t_k \geq t_{\sigma(h)} + f_{hk}$ holds the next operation on the same block section (k). A timing t_h is feasible if for every operation h the running and setup times constraints are satisfied. In addition, a feasible timing corresponds to a conflict-free and deadlock-free schedule.

An alternative graph is a triple $\mathcal{G} = (N, F, A)$, where N is a set of nodes, F is a set of fixed arcs, and A is a set of pairs of alternative arcs. Each arc (h, k), either fixed or alternative, has a length w_{hk}. A *selection* S is a set of alternative arcs obtained choosing at most one arc from each pair. The selection S is *consistent* if the graph $\mathcal{G}(S) = (N, F \cup S)$ contains no positive length cycles. The selection S is *complete* if exactly one arc from each pair is selected. An empty selection is indicated as $S^{\emptyset}$ and a partial selection as S'.

In our formulation, an operation h is associated to a *node* h of the alternative graph. Two additional dummy nodes 0 and n represent the start and end of the schedule, and each train route is modeled with a chain of fixed arcs from 0 to n. A fixed arc (h, k) corresponds to the precedence relation $t_k \geq t_h + w_{hk}$. Fixed arcs model feasible timings for the routes, i.e., for each operation h of a route there is a fixed arc $(h, \sigma(h))$ with length $w_{h\sigma(h)} = f_h$, resulting from its assigned speed profile. However, fixed arcs are used also to impose other constraints (e.g. release times, rolling stock constraints, connected train services) and to

compute the consecutive delay of a schedule. Alternative arcs are used to avoid conflicts between trains. For each alternative pair $((\sigma(h), k), (\sigma(k), h))$, operations h and k are associated with the entrance of two trains in same block section. The length of the alternative arcs is given by the setup times, i.e., ensuring minimum distance headways between consecutive trains. A more detailed description of the alternative graph formulation of railway traffic management constraints can be found in D'Ariano et al. (2006, 2007a, b) and Mascis et al. (2004).

4.2 Resolution procedures

This section describes the approaches adopted to solve the short-term train dispatching problem. After all the necessary information have been elaborated by the load information and disruption recovery modules, we generate an alternative graph $\mathcal{G} = (N, F, A)$ representing all the trains running during the entire time horizon. We then divide the time horizon in m time intervals. Clearly, when dealing with a single time interval ($m = 1$), no decomposition is performed and the approach corresponds to a global resolution. On the other hand, for $m > 1$ we adopt a temporal decomposition approach. In what follows, we explain the two approaches with reference to the algorithmic description of Fig. 2.

The global resolution approach directly solves $\mathcal{G} = (N, F, A)$ using one of the scheduling strategy of Section 3. In this case ($m = 1$), τ is the length of the entire time horizon and a solution in the time interval $[0, \tau]$ is denoted as $\mathcal{G}(S)$, where S is a complete consistent selection for $\mathcal{G} = (N, F, A)$.

We now describe the temporal decomposition scheme for dispatching trains when the traffic prediction is enlarged up to several hours and the alternative graph $\mathcal{G} = (N, F, A)$ becomes difficult to solve due to the huge problem size. The large instance is thus divided into m tractable alternative graphs to be solved successively. In this case, we suppose that each graph is extended over a time interval of length τ.

The temporal decomposition approach may require the splitting of some train routes over two time intervals. The resolution of successive time intervals thus requires coordination constraints for the split train routes (i.e., the computation of feasible positions and speeds for those train routes). For example, if a train runs at maximum speed in a given time interval and has to brake at the entrance of the successive time interval there may be problems due to minimum required space headway and time for braking. It follows that the resolution of successive time intervals must be coordinated in order to avoid infeasibilities.

We solve the problem of coordinating m time intervals, enforcing some precedence relation constraints between consecutive time intervals, i.e., the coordination constraints. Given an i-th time interval, we use the solutions obtained for the previous time intervals to generate the graph $\mathcal{G}^i(S^{\emptyset})$. Specifically, we set the release time of each train in $\mathcal{G}^i(S^{\emptyset})$ according to the rolling stock re-use constraints. The "SetDecompositionConstraints" procedure of Fig. 2

MainMethod
input:
a temporal length of each time interval τ
a $m\tau$-time horizon instance $\mathcal{G}(N, F, A)$
begin
generate the first time interval instance $\mathcal{G}^1(S^{\emptyset})$
compute a solution $\mathcal{G}^1(S)$ using the iterative or single scheduling strategy
if $m > 1$
for all i from $i = 2$ up to $i = m$
generate the i-th time interval instance $\mathcal{G}^i(S^{\emptyset})$
SetDecompositionConstraints($\mathcal{G}^{i-1}(S)$,$\mathcal{G}^i(S^{\emptyset})$)
compute a solution $\mathcal{G}^i(S)$ using the iterative or single scheduling strategy
end

SetDecompositionConstraints
input:
the $(i-1)$-th time interval (solved) instance $\mathcal{G}^{i-1}(S)$
the i-th time interval (unsolved) instance $\mathcal{G}^i(S^{\emptyset})$
begin
let $Z_i = \emptyset$ be the list of trains running in both $[(i-2)\tau, (i-1)\tau]$ and $[(i-1)\tau, i\tau]$
initialize the set of alternative arcs $X_i = \emptyset$
for all running trains in $\mathcal{G}^{i-1}(S)$
if the current train T_j is also in $\mathcal{G}^i(S)$
insert T_j in Z_i
find the starting time t_{T_j} of the first operation of T_j in the i-th time interval
starting from t_{T_j}, calculate the minimum time $t^*_{T_j}$ at which T_j can stop
insert in X_i all the alternative pairs of T_j in $\mathcal{G}^i(S^{\emptyset})$ until $t^*_{T_j}$
for all alternative pairs of X_i
if one arc (h, k) of the current pair is in the selection of $\mathcal{G}^{i-1}(S)$
select the arc (h, k) also in $\mathcal{G}^i(S')$
else select the current pair in $\mathcal{G}^i(S')$ giving precedence to the trains in Z_i
end

Fig. 2 Algorithmic description of the proposed resolution procedures

then computes the required timing constraints between $\mathcal{G}^{i-1}(S)$ and $\mathcal{G}^i(S^{\emptyset})$ as follows. For each train T_j spanning the two time intervals $[(i-2)\tau, (i-1)\tau]$ and $[(i-1)\tau, i\tau]$, we have to find the starting time t_{T_j} of its first operation in the i-th time interval. Starting from t_{T_j}, we calculate the minimum time $t^*_{T_j}$ at which T_j can stop its run. Precisely, $t^*_{T_j}$ is computed starting from the scheduled speed and position of T_j at t_{T_j} while respecting the signaling system, infrastructure and rolling stock characteristics (see D'Ariano et al. 2007b for a description of the adopted driver behaviors). Before the resolution of the i-th time horizon, we thus set the precedence relations between T_j and the

other running trains until $t^*_{T_j}$. A subset of alternative pairs X_i is created to insert the timing constraints in $\mathcal{G}^i(S^{\emptyset})$. Specifically, all alternative pairs of X_i between two trains running in $[(i-2)\tau, (i-1)\tau]$ are selected in $\mathcal{G}^i(S^{\emptyset})$ as in the selection of $\mathcal{G}^{i-1}(S)$. The remaining unselected alternative pairs of X_i are selected in $\mathcal{G}^i(S^{\emptyset})$ giving precedence to the trains running in both time intervals. Finally, the resulting partially selected graph $\mathcal{G}^i(S')$ is solved. Successively, the alternative graph of the next time interval is generated and a new run of the "SetDecompositionConstraints" procedure is performed. This iterative procedure terminates when the m-th time interval is solved. In general, we define the set of all alternative pairs constrained between consecutive time intervals as $X = \sum X_i$ with $i = 2, ..., m$. This decomposition procedure enables our dispatching system to compute locally feasible schedules corresponding to a globally feasible solution.

4.3 Illustrative example

Consider the small rail network reported in the upper part of Fig. 3, in which we have four running trains (T_A, T_B, T_C and T_D), and a time horizon of length 2τ. For these four trains we assume that T_A completes its route within the time interval $[0, \tau]$, while T_B spans the time interval $[0, 2\tau]$. The other two trains, T_C and T_D, travel in $[\tau, 2\tau]$. Furthermore, T_A and T_C are scheduled to be coupled at a platform track station outside the railway area considered. In other words, T_A runs completely over its network route and, after a turning buffer time, its rolling stock is re-used by T_C. For simplicity, in this example we do not specify the running and setup times of the involved trains, and we only show the location of the most relevant block signals.

The alternative graph shown in the lower part of Fig. 3 represents the global formulation approach, i.e., all the running trains are modeled in the graph. We denote a node with the pair (train, block section) of the associated operation, except for the dummy nodes. The number on each node corresponds to

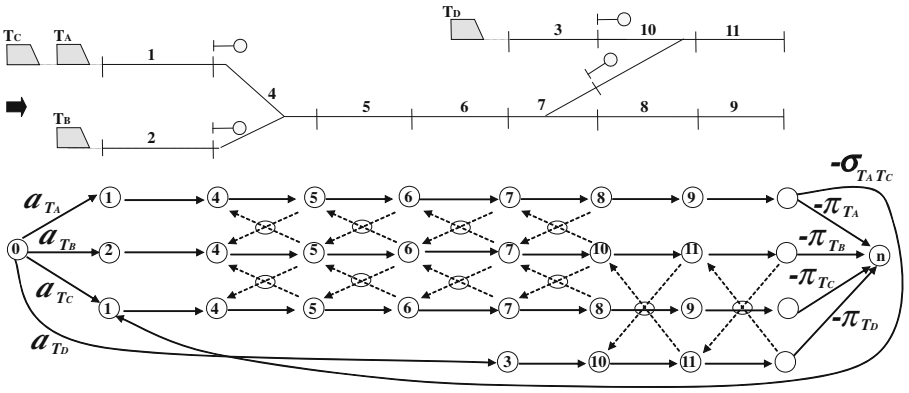

Fig. 3 Alternative graph formulation of a small network with four trains

the associated block section, while the train is shown on the corresponding entrance arc. The four fixed arcs departing from node 0 model the release time of each train ($\alpha_{T_A}, \alpha_{T_B}, \alpha_{T_C}$ and α_{T_D}), whereas the arcs entering node n model the objective function, with length $-\pi_{T_A}$, $-\pi_{T_B}$, $-\pi_{T_C}$ and $-\pi_{T_D}$, respectively. In general, a train T_y arriving in the dummy node n at a time greater than π_{T_y} will be late. The fixed arc connecting T_A with T_C is used to represent the rolling stock constraint between these two trains and the corresponding turning buffer time has length $\sigma_{T_A T_C}$. There are ten alternative pairs, which are depicted by dashed arcs.

We now show how the small example of Fig. 3 can be temporally decomposed into two instances. Let τ be the length of the time interval of each instance. In the alternative graph of Fig. 4(a), we show the alternative graph formulation of $[0,\tau]$. Two trains, T_A and T_B, and four alternative pairs are depicted in this graph.

A dispatching solution to $[0,\tau]$ leads to the graph shown in Fig. 4(b), in which T_A precedes T_B on the block sections 4, 5, 6 and 7. Specifically, only the selected alternative arcs are depicted in Fig. 4(b). Let t_{T_B} be the time at which the first operation of T_B starts in $[\tau,2\tau]$, corresponding to the entrance of block section 5, and let $t^*_{T_B}$ be the minimum time required to stop T_B in $[\tau,2\tau]$. In order to solve $[\tau,2\tau]$ after the resolution of $[0,\tau]$, T_B must not change its speed profile at least until $t^*_{T_B}$.

In Fig. 4(c), we depict the alternative graph of $[\tau,2\tau]$. Since T_B has coordination constraints with T_C until $t^*_{T_B}$, this must be the first train passing on block sections 4 and 5, corresponding to the selection of two alternative arcs in the graph of $[\tau,2\tau]$. The set X therefore contains the two corresponding alternative pairs. To compute a complete selection for this graph, four alternative pairs must be selected. In this way, we ensure that the solution to $[\tau,2\tau]$ is compatible with the solution to $[0,\tau]$. In order to include the rolling stock constraint

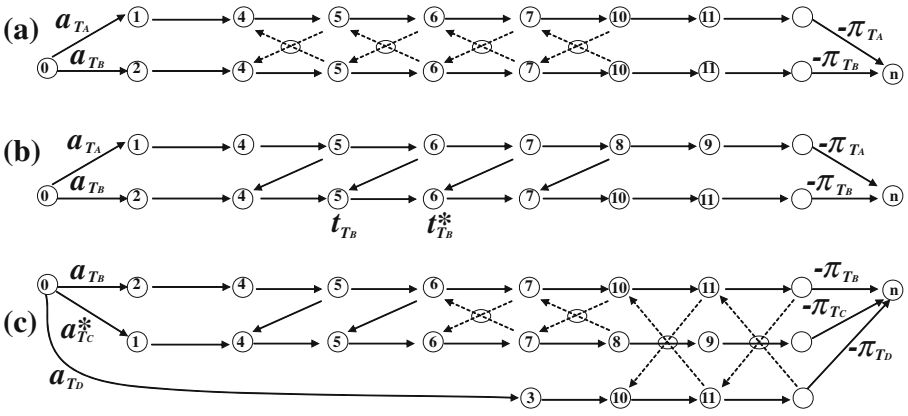

Fig. 4 Decomposition phases: (a) formulation of time interval $[0,\tau]$; (b) solution to time interval $[0,\tau]$; (c) formulation of time interval $[\tau,2\tau]$

between T_A and T_C, we also have to model a new release time $\alpha^*_{T_C}$, which must include the original release time α_{T_C} plus the delay of the rolling stock used by T_A, if positive. The solution obtained combining the local solution to the three graphs of Fig. 4 may be sub-optimal, even if every graph is solved to optimality. This is clearly due to the presence of the coordination constraints between consecutive time intervals, which are needed to compute a feasible global solution.

5 Computational experiments

This section presents our experiments on a large sample of practical size instances. We consider the dispatching area of the route Utrecht-Den Bosch, a bottleneck of the Dutch railway network. We study the network simulating disturbed traffic conditions, i.e., entrance delays, dwell time perturbations and blocked tracks. Dispatching algorithms are implemented in C++ language and executed on a laptop equipped with a 1.6 GHz Pentium M processor.

5.1 Test case description

Our computational tests simulate the area of control of a human dispatcher. The railway area under study is shown in Fig. 5, and consists of 191 block sections and 21 platforms. This railway includes the Den Bosch station and the line connecting Utrecht (Ut) to Den Bosch (Ht), which is around 50 km long. There are two main tracks, divided into one long corridor for each traffic direction, a dedicated stop for freight trains (Ozbm) and seven intermediate passenger stations: Utrecht Lunetten (Utl), Houten (Htn), Houten Castellum (Hc), Culemborg (Cl), Geldermalsen (Gdm), Zaltbommel (Zbm) and Den Bosch (Ht). Each traffic direction has nine entrances: Utrecht (Ut), Dordrecht (Ddr), Geldermalsen Yard, Nijmegen (Nm), Betuweroute, Oss (Oss), Den Bosch Yard, Eindhoven (Ehv) and Tilburg (Tl). Two extensions of the network, which are still under construction, are also modeled (block sections: 96, 98, 128, 129 and from 131 to 140).

We consider a 2007 timetable variant which is hourly, cyclic and extended to the entire railway area. During peak hours, 26 passengers and freight trains in both directions are scheduled for the area around Geldermalsen. A more complex situation occurs at Den Bosch station, where up to 40 trains are scheduled each hour. The infrastructure offers a few possibilities for train reordering and rerouting. For each train a default route and a set of local rerouting options are given. Rerouting options can be applied along corridors or within a station, where a train may be allowed to stop at different nearby platforms.

We incorporate in the model all the constraints described in Section 2. Specifically, we evaluate the effects of constraints due to minimum transfer time between connected train services. Rolling stock connections are provided

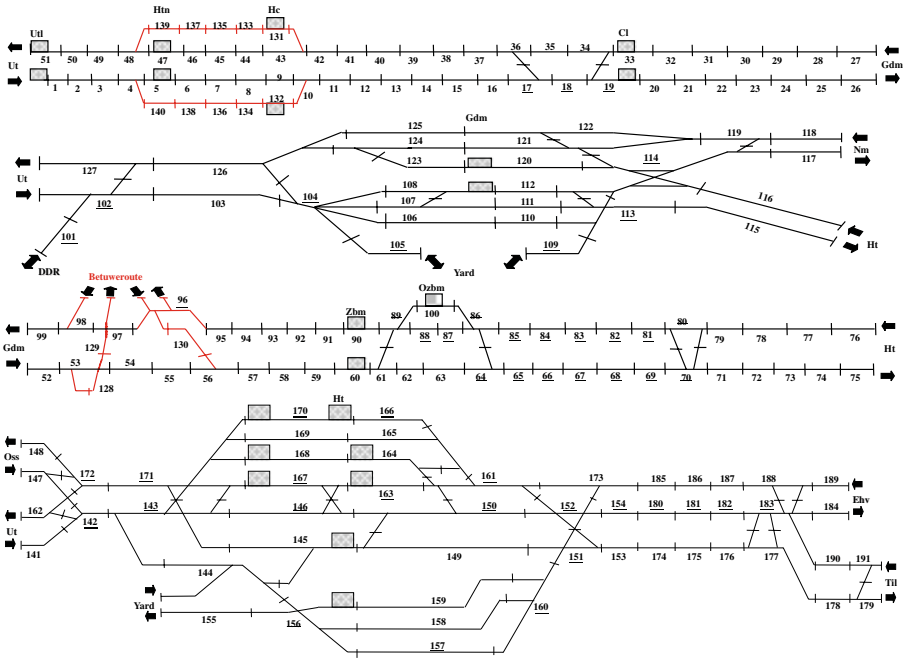

Fig. 5 Utrecht Den Bosch railway dispatching area

in Zaltbommel and Den Bosch stations. Connected train services (i.e., passenger connections) are modeled at Den Bosch station for the traffic directions from Ossen to Utrecht and vice versa. The minimum time for passenger connections varies from 2 to 5 min, depending on the distance between the arrival platforms. We also consider constraints due to the re-use of rolling stock when trains leave and enter the dispatching area. Intercity trains generally have long turning buffer times, and small delays are very likely to be completely absorbed. For local services, the situation is slightly different. Outside the studied dispatching area, some trains run along the corridors with long dwell times at major railway stations (up to 10 min), so we assume that delays will fade out. For others trains having larger delays or shorter dwell times, delays may survive the remaining part of their round-trip. In this case, we model rolling stock re-use constraints with the following assumptions: Trains with up to 2 h round-trip have a maximum turning buffer time of 12 min, while trains with larger round-trip times have one of 30 min.

We test ROMA under severe disturbances. Effects of disruptions and entrance perturbations are studied while considering a time horizon of several timetable hours. Randomly generated disruptions are included in each hour of traffic prediction while a set of trains running in the first timetable hour is delayed on the basis of entrance perturbations chosen in a time window of typical train delays. Output delays of each hour cause extra entrance delays since the rolling stock is re-used on a cyclic basis and delays cannot be

Table 1 Timetable disturbances description

Disturbance category	Entrance perturbations		Timetable disruptions		
	Max delay	Delayed trains	Rerouted trains	Blocked tracks	Perturbed dwell time
Small	300	5	0	0	0
Medium	900	10	5	1	30
Large	1500	20	10	4	60

completely absorbed. Moreover, disruptions obstruct main traffic directions and cause serious propagation of train delays. In fact, when a double track corridor is blocked in one of the two directions, trains traveling in opposite directions must share the only track available.

Table 1 describes the disturbances used to test ROMA. The first column indicates the disturbance category. The second column presents the values of maximum entrance delay (in seconds), while the third column shows the number of trains with an entrance delay larger than zero. For those trains the entrance delay is chosen randomly according to a uniform distribution and up to a given maximum value. Three instances are generated for each value of columns 2 and 3, yielding a total of 27 entrance delay configurations. Columns 4, 5 and 6 are used to represent disruptions in the network. Column 4 gives the number of trains rerouted by the disruption recovery module. Column 5 shows the number of blocked tracks. Specifically, the case of one blocked track corresponds to the unavailability of block section 83, while the other disruption (the large disturbance) corresponds to the unavailability of block sections: 168, 164, 67 and 175. The last column shows the dwell time perturbations (in seconds) which are applied to all the rerouted trains at their perturbed station stops. For each couple (blocked tracks, dwell time perturbation), we generate a disrupted timetable and for each of these timetables we test the 27 perturbations. In total, there are 81 disturbances, which are divided in 45 small, 27 medium and 9 large.

In the following subsections, we present the computational results. We first evaluate the best ROMA configuration over a subset of large disturbances. Next, we provide a detailed description of the solutions obtained using the best configuration. Each run of the BB algorithm is truncated after 30 s of computation. Computation times and delays are always expressed in seconds.

5.2 Testing ROMA configurations

The real-time purpose of train dispatching imposes strict time limits to produce a new feasible timetable, limiting the execution of the real-time optimization module. A reduction of the computation time of the scheduling algorithm can be attained by decomposing the time horizon into reasonable time intervals, at the cost of less accurate schedules. In this subsection, we study these aspects for three large disturbances of Section 5.1.

Table 2 presents a comparison between four ROMA configurations, varying the scheduling algorithm (BB or FCFS) and the scheduling strategy. The results are shown in terms of consecutive delays, since primary delays here are considered unavoidable. Each row of the table presents average results computed for increasing length of the time horizon. Specifically, the first column indicates the time horizon in terms of timetable hours. The second column reports on the solutions computed using the single scheduling strategy (case "Single") or the iterative scheduling strategy (case "Iterative"). Columns 3–6 and 7–10 show the results with BB and FCFS, respectively. Columns 3 and 7 give the maximum consecutive delays. Columns 4 and 8 show the average consecutive delays. Columns 5 and 9 indicate the average computation time of the four ROMA configurations over the 3 disturbances. Finally, columns 6 and 10 give the average number of speed profile adjustments due to trains facing a yellow or red signal aspect (i.e., the runs of the speed updating procedure). Being a simple counter, this indicator may be very sensitive to the number of running trains.

From this set of experiments, the combination of BB and iterative scheduling is the best configuration in terms of maximum consecutive delay minimization and provides good results in terms of average consecutive delays. However, the main difference between the four configurations is that the computation time increases considerably when the time horizon of traffic prediction is enlarged. This effect does not depend on the number of speed profile adjustments but is directly related to the scheduling process and instance size. When dealing with real-time application purposes, the global approach is only suitable for time horizons of up to 2 h.

Table 3 presents a comparison of the two resolution approaches of Section 4.2 (i.e., the global and temporal decomposition) when varying the time horizon of traffic prediction. For both approaches, we use the best configuration of Table 2, i.e., BB and iterative scheduling. The first column of Table 3 refers to the time horizon length, expressed in timetable hours. The second column reports on the solutions computed using the global approach (i.e., the first three rows) and temporal decomposition approach (i.e., the successive rows). Columns 3 and 4 show the maximum and average consecutive delays respectively. Columns 5 indicates the average computation time and column 6 gives the average number of speed profile adjustments. Column

Table 2 Results with the global formulation approach and four system configurations

Time horiz	Scheduling strategy	BB algorithm				FCFS algorithm			
		Max delay	Avg delay	Comp time	Num iter	Max delay	Avg delay	Comp time	Num iter
1	Single	336	25.76	6	49	440	24.25	6	51
2	Single	384	24.81	151	114	480	37.21	122	113
3	Single	384	21.09	2057	149	480	35.63	2514	154
1	Iterative	305	25.97	14	50	336	25.59	14	53
2	Iterative	339	23.27	690	107	423	23.50	693	114
3	Iterative	350	21.05	16740	142	423	30.01	19870	156

Table 3 Global versus decomposition

Time horiz	Resolution approach	Max delay	Avg delay	Comp time	Num iter	Running trains	$\|A\|$	$\|X\|$
1	Global	305	25.97	14	50	40	4133	–
2	Global	339	23.27	690	107	80	17734	–
3	Global	350	21.05	16740	142	120	40803	–
1	Decomp	305	25.97	14	50	40	4133	–
2	Decomp	443	26.29	503	175	92	12889	2858
3	Decomp	443	21.29	548	233	134	18119	3906
4	Decomp	443	19.84	597	305	177	23368	4822
5	Decomp	443	16.42	637	359	218	28506	5812
6	Decomp	443	17.59	677	422	260	33656	6677
7	Decomp	443	16.78	714	474	302	38604	7481
8	Decomp	443	16.16	750	527	343	43659	8350
9	Decomp	443	14.59	787	579	385	48606	9154

7 presents the average number of trains running in the corresponding time horizon. We note that each hour of timetable has 40 running trains and the instances of the global approach increase of this number of trains for each hour of traffic prediction. However, the instances of the temporal decomposition approach present a larger number of trains. This is due to the fact that some delayed train routes are split into two consecutive hours of timetable, e.g. the time horizon of 9 h contains 25 split train routes. The average number of alternative pairs is indicated in column 8, while the last column shows the number of alternative pairs "fixed" when using the temporal decomposition approach.

The global approach gives better results in terms of delay minimization, reducing on average the maximum consecutive delay of 21% compared to the temporal decomposition approach. The temporal decomposition approach is sub-optimal because a partial selection of alternative arcs is fixed (i.e., $|X|$) for the coordination of successive time intervals. On the other hand, the temporal decomposition is the only viable approach for computing traffic predictions over a time horizon of more than 3 h. The complexity of large time horizons is directly due to the number of alternative pairs (i.e., $|A|$), while the number of speed profile adjustments is not a critical time consuming factor.

This first set of experiments allowed us to set the best ROMA configuration. Moreover, the decomposition approach offers similar results to the global approach while making the code more compatible with short-term predictions of rail operations.

5.3 Effects of timetable disturbances

The ROMA performance is now evaluated using the best configuration of Section 5.2 and the temporal decomposition approach. We study the overall set of disturbances of Section 5.1 when varying the time horizon of traffic prediction. The average results are reported for enlarging the time horizon length hour by hour, up to 9 timetable hours.

Figure 6 shows the average times to compute short-term traffic predictions. Results are divided into the three disturbances categories of Table 1. These computational results show that ROMA can handle 1 h instances within a few seconds (independently from the disturbance category), which is a reasonable time horizon length for a real-time decision support system. When dealing with larger time horizons and up to medium disturbances, ROMA returns, on average, a solution within 5 min of computation. Starting from a time horizon of 2 h, large disturbances are already too time consuming compared to real-time application purposes, i.e., considerable modifications of speed and location of trains may happen while the dispatching system is computing a solution. In this case, the most critical factor for a real-time use of such a system is not the time horizon length but the level of disturbance in the network. Besides, the presence of several blocked tracks may cause problems when computing a new feasible timetable in real-time.

Figures 7 and 8 show the average results in terms of maximum and average total delays. Since traffic perturbations are inserted in the first hour of timetable, in general, the maximum total delays increase between the first 2 h of traffic prediction and decrease strongly after the second hour of traffic prediction because of available recovery times. The average total delays also decrease when enlarging the time horizon length. However, the successive hours are still perturbed because of propagation of delays and of the presence of blocked tracks that do not allow a complete return to the original traffic situation.

Figure 7 reports the total delays on the basis of different types of the entrance disturbances. The results, obtained for the overall set of instances, are divided into six plots. The three plots on the left show the maximum total delays for each disturbance category of Table 1 by varying the timetable disruption, the maximum entrance delay and the number of delayed trains. The average delays are similarly shown on the right of the figure. These plots describe the propagation of train delays, while dispatchers usually cannot precisely evaluate the short-term consequences of timetable disturbances in complicated railway networks. In fact, it is impossible for them to keep a complete check on all trains running in the railway network and predict and control their future movements.

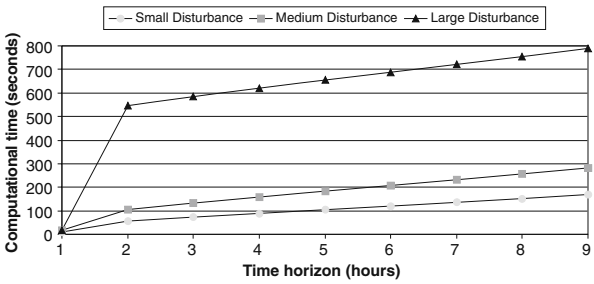

Fig. 6 Computation times for short-term prediction of railway traffic

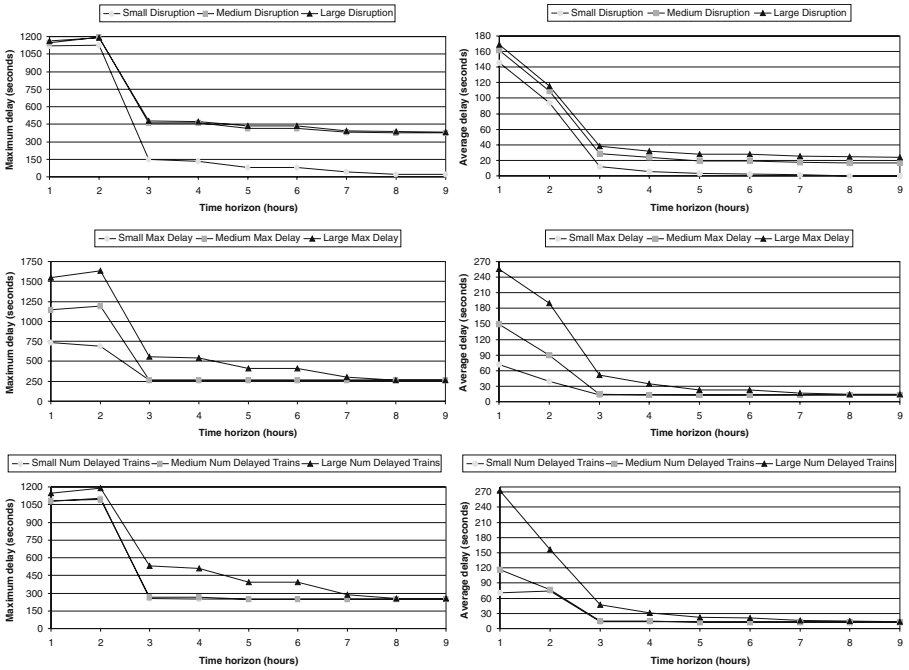

Fig. 7 Short-term propagation of train delays varying the timetable disturbances (disruptions, maximum entrance delays and number of delayed trains)

When dealing only with perturbations (small disruption), the maximum total delay drops to 20 s after a time period of 9 h. In case of blocked tracks (medium and large disruptions), the total delays are more difficult to recover as the trains change their scheduled routes. Different maximum values of entrance perturbations have a significant direct impact on the total delays. Similar results are obtained when changing the number of delayed trains.

As far as the total delay at each relevant point is concerned, from Fig. 8 we observe that Den Bosch (Ht) is the most critical stop in the first 2 h of traffic prediction. From the third hour on, the critical station becomes Zaltbommel

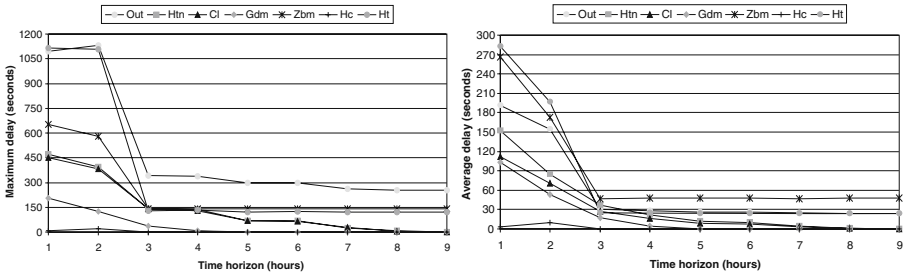

Fig. 8 Short-term propagation of train delays at the stations (Htn, Cl, Gdm, Zbm, Hc and Ht) and dispatching area borders ("Out")

(Zbm) and the largest delays are registered in the border areas of the railway network ("Out"). In general, when the short-term prediction enlarges over 3 h the average total delay stabilizes at Zaltbommel station, Den Bosch station and the border areas while decreasing at the other stations. After 9 h of traffic prediction, the average total delay is less than 1 min at all relevant points whereas the maximum total delay at the border areas is reduced of 77%. Furthermore, Den Bosch (Ht) registers the largest delay in the second hour of traffic prediction, i.e., around 20 min. This value is greater than the maximum total delay at the other stations.

6 Conclusions

This paper showed the effectiveness of our dispatching system in order to reduce the propagation of train delays. The proposed tool can be adopted for the real-time railway traffic control process and for the evaluation of disturbances of up to several timetable hours.

While the temporal decomposition approach allows the resolution of large scheduling problems, the global approach is rather time consuming but offers better solutions in terms of delay minimization. Further research should therefore be addressed to the analysis of more sophisticated techniques of problem decomposition in order to fill the gap between solution quality and computation times, and to cope with the railway traffic management in consecutive dispatching areas.

More general research should address the problem of optimizing the use of other dispatching measures such as cancellation of train routes, modification of connected train services and ad-hoc rerouting actions. These measures may also influence train speed profiles and orders at conflict points, resulting in complex compound problems. In a paper to follow, we study a real-time railway traffic management system that combines retiming, reordering and local rerouting actions to recover train delays.

Acknowledgements We thank ProRail (The Netherlands) for providing the instances, I.A. Hansen and D. Pacciarelli for their valuable comments. This work is partially supported by the programs "Towards Reliable Mobility" of the Transport Research Centre Delft, the Dutch foundation "Next Generation Infrastructures" and ProRail.

References

Adams J, Balas E, Zawack D (1988) The shifting bottleneck procedure for job shop scheduling. Manage Sci 34:391–401

Ahuja RK, Magnanti TL, Orlin JB (1993) Network flows: theory, algorithms, and applications. Prentice Hall, Englewood Cliffs

Assad AA (1980) Models for rail transportation. Transp Res Part A 14(A):205–220
Chambers R, Carraway R, Towe T, Morin T (1991) Dominance and decomposition heuristics for single machine scheduling. Oper Res 39(4):639–647
Cordeau JF, Toth P, Vigo D (1998) A survey of optimization models for train routing and scheduling. Transp Sci 32(4):380–420
D'Ariano A, Albrecht T (2006) Running time re-optimization during real-time timetable perturbations. In: Allan J, Brebbia CA, Rumsey AF, Sciutto G, Sone S, Goodman CJ (eds) Computers in railways X. WIT, Southampton, pp 531–540
D'Ariano A, Corman F, Pacciarelli D, Pranzo M (2006) Real-time train conflict detection and resolution: global sequencing and local rerouting. In: van Zuylen HJ (ed) Proceedings 9th TRAIL congress, selected papers. Delft University Press, Delft, pp 77–92
D'Ariano A, Pacciarelli D, Pranzo M (2007a) A branch and bound algorithm for scheduling trains in a railway network. Eur J Oper Res 183(2):643–657
D'Ariano A, Pranzo M, Hansen IA (2007b) Conflict resolution and train speed co-ordination for solving real-time timetable perturbations. IEEE Trans Intell Transp Syst 8(2):208–222
Hansen IA, Dekking FM, Goverde RMP, Heidergott B, Meester LE (eds) (2005) Proceedings of the 1st international seminar on railway operations modelling and analysis, Delft, 8–10 June 2005
Jacobs J (2004) Reducing delays by means of computer-aided 'on-the-spot' rescheduling. In: Allan J, Brebbia CA, Hill RJ, Sciutto G, Sone S (eds) Computers in railways IX. WIT, Southampton, pp 603–612
Kauppi A, Wikström J, Sandblad B, Andersson AW (2006) Future train traffic control: control by re-planning. Cogn Technol Work 8(1):50–56
Mascis A, Pacciarelli D (2002) Job shop scheduling with blocking and no-wait constraints. Eur J Oper Res 143(3):498–517
Mascis A, Pacciarelli P, Pranzo M (2004) Scheduling models for short-term railway traffic optimization. In: Proceedings of the 9th international conference on computer-aided scheduling of public transport. Lecture notes in economics and mathematical systems, San Diego, 9–11 August 2004
Mazzarello M, Ottaviani E (2007) A traffic management system for real-time traffic optimization in railways. Transp Res Part B 41(2):246–274
Oh SM, Hong SH, Choi IC (2004) Railway conflict detection and resolution in the Korean railway system. In: Allan J, Brebbia CA, Hill RJ, Sciutto G, Sone S (eds) Computers in railways IX. WIT, Southampton, pp 675–684
Ovacik IM, Uzsoy R (1997) Decomposition methods for complex factory scheduling problems. Kluwer, Boston
Pacciarelli D, Pranzo M (2006) Speed regulation in rail networks. In: van Zuylen HJ, Middelham F (eds) Proceedings of the 11th IFAC symposium on control in transportation systems, Delft, 29–31 August 2006
Pachl J (2002) Railway operation and control. VTD Rail, Mountlake Terrace
Pardalos P, Resende G (2002) Handbook of applied optimization. Oxford University Press, New York
Rodriguez J (2000) Empirical study of a railway traffic management constraint programming model. In: Allan J, Hill RJ, Brebbia CA, Sciutto G, Sone S (eds) Computers in railways VII. WIT, Southampton, pp 1017–1025
Rodriguez J (2007) A constraint programming model for real-time trains scheduling at junctions. Transp Res Part B 41(2):231–245
Törnquist J (2005) Computer-based decision support for railway traffic scheduling and dispatching: a review of models and algorithms. In: 5th workshop on algorithmic methods and models for optimization of railways. http://drops.dagstuhl.de/opus/volltexte/2006/659
Törnquist J, Persson JA (2007) N-tracked railway traffic re-scheduling during disturbances. Transp Res Part B 41(3):342–362
van den Top J (2006) Effectiveness of performance indicators and process control in Dutch rail transport. In: van Zuylen HJ (ed) Proceedings 9th TRAIL congress, CD-ROM. Delft University Press, Delft
Zeng X, Happiette M, Rabenasolo B (1998) Analysis of the temporal decomposition procedure for scheduling with release and due dates. Eur J Oper Res 109:599–619