Approaches to a real-world train timetabling problem in a railway node

Valentina Cacchiani

DEI, University of Bologna, Viale Risorgimento 2, 40136 Bologna, Italy
valentina.cacchiani@unibo.it

Fabio Furini

LAMSADE, Université Paris-Dauphine, Place du Maréchal de Lattre de Tassigny, 75775 Paris, France
fabio.furini@dauphine.fr

Martin Philip Kidd

DEI, University of Bologna, Viale Risorgimento 2, 40136 Bologna, Italy
martin.kidd@unibo.it

Abstract

We consider the Train Timetabling Problem (TTP) in a railway node (i.e. a set of stations in an urban area interconnected by tracks), which calls for determining the best schedule for a given set of trains during a given time horizon, while satisfying several *track operational constraints*. In particular, we consider the context of a highly congested railway node in which different *Train Operators* wish to run trains according to timetables that they propose, called *ideal timetables*. The ideal timetables altogether may be (and usually are) conflicting, i.e. they do not respect one or more of the track operational constraints. The goal is to determine conflict-free timetables that differ as little as possible from the ideal ones. The problem was studied for a research project funded by Rete Ferroviaria Italiana (RFI), the main Italian railway Infrastructure Manager, who also provided us with real-world instances. We present an Integer Linear Programming (ILP) model for the problem, which adapts previous ILP models from the literature to deal with the case of a railway node. The Linear Programming (LP) relaxation of the model is used to derive a dual bound. In addition, we propose an iterative heuristic algorithm that is able to obtain good solutions to real-world instances with up to 1500 trains in short computing times. The proposed algorithm is also used to evaluate the capacity saturation of the railway nodes.

Keywords: Train timetabling, Integer Linear Programming, Relaxation, Heuristic Algorithm.

1. Introduction

Railway systems are very complex and their efficiency is essential for railway *Infrastructure Managers*, especially in a competitive market in which several *Train Operators* use the same infrastructure. The increasing utilization of railways for passengers and freights requires an

effective usage of infrastructure. In this paper, we focus on the *Train Timetabling Problem* (TTP) at a planning level, which corresponds to a fundamental and very difficult phase in the optimization of a railway system. We studied the problem within a research project funded by Rete Ferroviaria Italiana (RFI), the main Italian railway Infrastructure Manager, and present in this paper the results obtained.

1.1. Problem definition

The TTP calls for determining the best schedule for a given set of trains during a given time horizon. In this paper, we face the TTP in a *railway node* as defined by the RFI. We consider a set of stations and junctions in an urban area interconnected by tracks and a set of trains to be scheduled in a time horizon of one day. This schedule must respect several *track operational constraints*:

- minimum *headway* times between consecutive arrivals or departures of trains from a station on a track must be respected;
- the maximum number of trains simultaneously at a station (including the trains that traverse a station) must respect the number of available platforms at the station;
- overtaking of trains along a track is forbidden;
- rules for traversing *junctions* must be satisfied.

In particular, we consider the context of a highly congested railway node in which different Train Operators wish to run trains according to timetables that they propose, called *ideal timetables*. An ideal timetable for a train specifies the stations to be visited by the train in the railway node together with ideal arrival and departure times of the train at each of these stations. The ideal timetables usually do not satisfy all the track operational constraints. Therefore, departure/arrival times must be changed in order to obtain a set of non-conflicting timetables. The goal is to change the ideal timetables as little as possible. To this aim, Train Operators assign a *profit* to each train, according to passenger demand of service. This profit is obtained if the train is scheduled according to its ideal timetable. Every change in the ideal timetables corresponds to a decrease in profit. If the profit becomes null or negative due to the changes, the train is cancelled, i.e. it is not scheduled. This is the point of view of the railway Infrastructure Manager, who collects the ideal timetables provided by the Train Operators and tries to fulfill their requests in order to maximize the total profit, while determining timetables that respect all the track operational constraints.

The motivation of this study in cooperation with RFI was to obtain a more efficient usage of the railway infrastructure inside big railway nodes, in order to obtain a better schedule of passenger and freight trains on the network, and consequently improve the service. Often railway nodes constitute the bottlenecks of the network, as many trains need to traverse or stop at one or more stations inside a node. An optimized schedule of the trains inside the node can therefore contribute to an improved efficiency of the entire system. Furthermore, we analyzed the so-called *capacity saturation* of the railway node, i.e. the maximum number of trains that can be scheduled according to some given ideal timetables. Overall, the goal was to develop an automated tool that could provide good solutions in limited computing time for scheduling trains in a railway node.

1.2. Literature review

We can distinguish between two main areas of study of the TTP: the TTP at a planning level, and the TTP at an operational level (also known as Train Timetabling Rescheduling or Train Dispatching or Conflict Resolution). In the first case, the goal is to determine a schedule for a set of trains to be used usually either for six months or for one year, while in the second case the goal is to reschedule trains after a disruption or a delay occurred which led to infeasibilities in the planned schedule.

A considerable amount of literature has recently been published on the TTP at an operational level. The problem is often represented as a job-shop scheduling problem with additional constraints, and modelled by using the Alternative Graph model introduced in [41] (see e.g. [22], [23], [40]). Alternative approaches are presented e.g. in [12], [14], [26], and [48]. We refer the reader to [10] for a recent survey on the topic.

There is a huge amount of work on the TTP at a planning level. Cyclic or periodic timetables (i.e. timetables that repeat identically every hour) are often designed, which are preferred by the passengers as they are easy to remember, but which in turn are more expensive in the off-peak hours. The seminal work [46] proposed the Periodic Event Scheduling Problem (PESP) for the cyclic TTP, in which, for a given train, an event represents the arrival at or the departure from a given station. These events are scheduled for one cycle (e.g. one hour) and then the cycle is repeated. Cyclic constraints are imposed to satisfy safety requirements. The PESP model has been applied and extended in several works: see e.g. [35], [36], [37], [42], [43], [44], [45].

In the context of a competitive market, non-cyclic or aperiodic timetables are usually preferred. We refer the reader to [11], [15], [17], [18], [21], [30] and [38] for extensive surveys on the topic.

Several mathematical formulations have been proposed for the non-cyclic TTP. One of the first models is due to Szpigel [47], who proposed a job-shop scheduling formulation for the TTP on a single track railroad. Carey and Lockwood, in [20], present a model that contains binary variables used to describe the precedences between trains, and continuous variables representing the time instants at which a train departs from or arrives at a station. Similar models are proposed in [32] and [29]. In [50], a multi-activity network data envelopment analysis model is proposed and applied to simultaneously estimate passenger and freight technical efficiency, service effectiveness, and technical effectiveness for 20 selected railways for the year 2002. In [3], a model for the expansion of the Spanish railway network is proposed and a tool based on a scatter search heuristic, which incorporate specific requirements such as population coverage, origin-destination passenger flows, and budget constraints, is developed for supporting the decision makers. In [5], it is assumed that each train track is divided into blocks, and for safety reasons only one train can be present at any given time instant on each block. The time horizon is discretized and a binary variable is introduced for each train, block and time instant, assuming a value of 1 if the train occupies the considered block at the specified time instant, and 0 otherwise. In [28], two mathematical formulations are compared on a common set of sample problems, representing both multiple track high density services in Europe and single track bidirectional operations in North America. One formulation consists of imposing time intervals between trains for avoiding conflicts, while the other formulation controls the physical occupation of track segments. The results demonstrate that both models return comparable solutions in the aggregate, with some significant

differences in selected instances. In [49], the optimization of train movement is studied in a railway network. Two objectives are taken into account: the expected total energy consumption and the total traversal time. Due to difficulties in using analytical calculations, simulation based approaches are designed to compute the expected total energy consumption and total traversal time. A genetic algorithm is integrated with simulation in order to find the best control strategies on the railway network. In [4] a micro-macro approach is proposed, which consists of starting from a detailed microscopic representation of the railway corridor and transforming it into a macroscopic representation that can be handled by state-of-the-art integer programming optimization methods. The optimized timetable is then re-transformed to the microscopic level. The method is applied on a real-world case study, namely the Simplon corridor between Switzerland and Italy. The method is also used for capacity analysis.

A customary way for modeling the TTP is to discretize the time horizon (e.g. with an interval discretization of one minute) and to represent the problem on a suitable *time-space graph*, where each node corresponds to a time instant in which a train can arrive at or depart from a station, and where arcs represent either the travel of a train from one station to the next or the stop of a train at a station. An ILP model based on this graph representation of the TTP is proposed by [16] for the case of a single one-way line, and extended in [19] for dealing with additional real-world constraints while being extended in [8] for dealing with a railway network. It uses binary variables specifying whether or not an arc of the graph is selected in the solution for a train. Track operational constraints are imposed by clique constraints that forbid the simultaneous use of incompatible arcs. This ILP formulation is a multicommodity flow formulation for the TTP. In [16] a reformulation of the multicommodity flow model is presented, in which the track operational constraints are modeled using variables associated with the nodes of the graph, and then relaxed in a Lagrangian way. [8] studied the problem of scheduling additional freight trains on congested railway networks where a given set of passenger trains has a prescribed timetable that cannot be changed. An ILP formulation with arc-variables is presented, and the problem is solved by using a heuristic algorithm based on a Lagrangian relaxation. The time-space graph representation of the TTP is also used in [7]: an ILP formulation is proposed in which each variable corresponds to a full timetable for a train and a branch-and-cut-and-price approach is developed to obtain optimal solutions to real-world instances on a corridor. In [9], ILP formulations for the TTP are examined, which involve (exponentially many) binary variables associated with paths in the time-space graph corresponding to the timetables. These ILPs call for a maximum-weight clique in the same (exponentially large) compatibility graph, contain only stable-set constraints and differ only in the type of stable set actually considered. Existing ILP models from the literature are analyzed and new stronger ones are presented.

1.3. Contribution

The problem under consideration in this paper is NP-hard, since it is an extension of the problem considered in [16] in which the authors proved that the TTP is NP-hard. The problem is often still solved manually by a team of human planners, who determine the solutions from experience and by adapting previous solutions of the problem on the same network. We propose an automated approach, which can be used by the planners not only as a black box, but mainly to quickly evaluate many solutions and choose the best one. A

preliminary version of this work was presented in [27], where a fast heuristic algorithm was proposed for scheduling trains in a railway node. The algorithm was based on a time-space graph representation of the problem and was tested on an instance of the railway node of Milan.

We model the problem by a time-space graph representation similar to the one used in [16], [19], [7], and in [8]. In [16], [19] and [7], the TTP is solved for a railway corridor (i.e. a single one-way line connecting two major stations), while [8] studies the case of a railway network. Railway nodes, in which a set of stations and junctions are connected by tracks, present a more complex structure compared to railway corridors. This structure is similar to the one of a railway network, but it shows some differences. In particular, in a railway network a higher level abstraction of the railway topology is considered, i.e. only major stations are usually considered, while junctions are not taken into account. This is especially true for large scale railway networks, such as those studied in [8], in which constraints on the maximum number of trains that can be simultaneously at a station are also neglected. In this paper, we take into account both complex rules for correct junction traversal (see Section 2) and constraints on the maximum number of trains allowed to simultaneously be present at a station.

Given the large number of trains (up to 1500) present in the real-world instances provided by RFI, we decided to develop a heuristic algorithm that can be executed in short computing times rather than adopting more sophisticated Lagrangian-based or LP-based heuristic methods. Indeed, the negotiation process between the Infrastructure Manager and the Train Operators, in the timetable planning phase, can require several rounds before reaching a compromise that is accepted by everyone. Keeping the computing time short is therefore fundamental for having an algorithm of practical use.

The main contributions of this article are as follows:

1. an Integer Linear Programming (ILP) model is adapted from previous models from the literature for our real-world case study of the railway node by taking additional real-life constraints into account;
2. the Linear Programming (LP) relaxation of the model is solved in order to obtain a dual bound on the optimal solution value, which is useful for evaluating the quality of the computed solutions;
3. an effective heuristic algorithm is developed to obtain good solutions in short computing times, which uses different train orderings and a dynamic construction of the time-space graph;
4. *capacity saturation* of the railway node is analyzed;
5. real-world instances from RFI are tested and computational results show the usefulness of the proposed method in practice.

The paper is organized as follows: the problem at study is described in Section 2. In Section 3 an ILP formulation for the problem is given and, in Section 4, we describe how we solve an LP-relaxation of the presented formulation. Section 5 is devoted to the description of the proposed heuristic algorithm and computational results on real-world instances from RFI are given in Section 6. Finally, we draw some conclusions and hints on future research in Section 7.

2. Problem description

In this section we provide a formal description of the case study of the TTP in a railway node at RFI. We are given as input the description of the railway node topology N. In particular, it contains a set S of *stations*, a set J of *junctions* and a set R of (mono-directional) *tracks* $r = (h, i) \in R$, each of which connects two locations $h, i \in L$ (either a station or a junction), with $L = S \cup J$. Each station $i \in S$ has a *capacity* c_i, which represents the number of trains that can simultaneously stop at the station i. Two locations can be connected by single or *parallel* tracks, i.e. two tracks connecting the same two locations in the same direction. Each junction $j \in J$ has an *occupation time* o_j. In addition, for each junction, a set of rules for the occupation of the junction is given: more precisely, the rules specify pairs of incompatible itineraries of trains simultaneously traversing the junction. We are also given as input a set T of *trains*, which belong to different Train Operators and are characterized by a *train type* such as Eurostar, Euronight, etc, and by a flag indicating if it is a high priority train or a low priority train. For each train, we are also given its *ideal timetable*. The ideal timetable of a train $t \in T$ specifies the set $L_t \subseteq S \cup J$ of locations to be visited, the order in which they must be visited, the preferred tracks to be used (in case of parallel tracks between stations, the set of parallel tracks that can be used by the train is specified), an ideal arrival time $\alpha(t, \ell)$ and an ideal departure time $\epsilon(t, \ell)$ at each location $\ell \in L_t$. Note that a minimum stopping time at each location is implicitly given by the ideal timetable, i.e. the stopping time must not be decreased compared to the ideal timetable. The travel times inside junctions and on tracks connecting locations are assumed to be fixed as specified by the ideal arrival and departure times for each train. Furthermore, for each train $t \in T$, an ideal profit π_t is given, which represents the profit associated with the train if it is scheduled according to its ideal timetable.

The following *track operational constraints* must be satisfied:

1. *departure/arrival constraints*: a minimum headway time $h^+(s, r)$ ($h^-(s, r)$ resp.) between two consecutive arrivals (departures resp.) of trains at the same station $s \in S$ on the same track $r \in R$ must be respected;
2. *station capacity constraints*: the maximum capacity of each station must be respected;
3. *no-overtaking constraints*: overtaking between trains is only allowed at stations or by using parallel tracks;
4. *junction constraints*: for each junction $j \in J$, itineraries which are incompatible with one used by a train cannot be used by other trains for the duration of the occupation time o_j of the junction.

The ideal timetables are generally conflicting, i.e. they do not respect one or more of the track operational constraints. In order to resolve these conflicts, one is allowed to modify (anticipate or delay) the departure time of each train from its first station (*shift*), to increase the stopping time interval at the (intermediate) stations (*stretch*) and to choose a parallel track between a pair of locations instead of the preferred track (*parallel rerouting*). For each train $t \in T$ a maximum shift sh_t^+ ahead of schedule and maximum shift sh_t^- behind schedule is specified, as well as a maximum total stretch ϕ_t over all the visited stations, given in minutes. Furthermore, each train is associated with three weights (given in input) ω_t^+, ω_t^- and ψ_t, which correspond to the penalty for shifting ahead of schedule, shifting behind

schedule and stretching a train for one minute, respectively. The shift and stretch changes decrease the profit of the trains. On the contrary, parallel rerouting does not modify the profit of the trains, since it does not affect the time schedule of the train nor the service to the passengers. Note that only the parallel tracks specified in its ideal timetable can be used by each train.

The two operations of shift and stretch have a different impact on the passengers: while the shift does not directly affect passengers (recall that it is applied in the planning phase), the stretch increases the overall travel time of the passengers to reach their destinations. Therefore, shift and stretch are chosen to be penalized independently, to avoid that stretch is used to "compensate" the need of shift to obtain a feasible timetable at "low" cost. Other types of measure of distance from the ideal timetable could be used, such as considering point-to-point distance or taking into account the number of passengers that travel along the line and the connections between trains.

The timetable of a train obtained after the changes have been applied is called *actual* timetable and the profit obtained by decreasing the ideal profit according to the shift and stretch penalties of the train is called *actual* profit. If the actual profit becomes null or negative, the train is cancelled. Indeed, it is not worth scheduling a train which causes a loss instead of a profit. Note that these changes and train cancellations do not directly affect the passengers since they are applied during the planning phase, i.e. before the schedule of the trains is published for the passengers.

3. Mathematical formulation

This section presents an ILP model for the TTP, which is adapted from ILP models presented in [16] and [19] for a railway corridor, and in [8] for a railway network (see Section 1.2 for further details). The model is based on a time-space graph representation with the time horizon of one day discretized to one minute intervals. In Section 3.1 we briefly recall this graph representation and show the ILP formulation that we use in Section 3.2.

3.1. Graph representation

Let H be the set of discrete time steps, in the given time horizon. According to the time discretization, we have $H = \{1, \dots, q\}$ (e.g. $q = 1440$ if we consider a time horizon of one day and a time interval of one minute). We define a time-space directed graph $G = (V, A)$ with node set V and arc set A. Each node represents either a departure or an arrival of some train at a specific location from a specific track at a specific time instant. In particular, for each track $r = (h, i)$ in R, there are a set $U(i, r)$ of arrival nodes at i on r and a set $W(h, r)$ of departure nodes from h on r. In addition, for notational convenience, we have in G an artificial *source* node σ and an artificial *sink* node τ. The set of nodes V is given by:

$$V = \{\sigma, \tau\} \cup \bigcup_{r=(h,i)\in R} (U(i, r) \cup W(h, r)).$$

Let $\theta(v)$ be the time instant associated with a given node $v \in V$ and $\Delta(u, v) := \theta(v) - \theta(u)$ be the time difference between two nodes. We say that node u *precedes* node v (i.e. $u \preceq v$) if $\Delta(v, u) \geq \Delta(u, v)$. I.e, if the time interval between $\theta(v)$ and $\theta(u)$ is not smaller than the time

interval between $\theta(u)$ and $\theta(v)$ (note that we will perform these subtractions by using the modulo q operation implicitly). Analogously, we will use the notation $u \prec v$, $u \succeq v$, $u \succ v$.

The set $V_t \subseteq V$ of nodes associated with each train $t \in T$ is defined by the source, the sink, and the nodes in V corresponding to possible departure and arrivals of train t along its path in the railway node, taking into account the structure of the railway node and the implicit time windows defined by the allowed changes of shift and stretch along the path.

Each arc of G represents either the travel of a train $t \in T$ from a station to a next one, the stop of t at a station or the travel of t through a junction. More precisely, arc set A is partitioned into sets A_t, one for each train $t \in T$. The arcs are defined as follows:

1. *travel arcs between stations*: for each track $r = (h, i) \in R$, corresponding to two locations $h, i \in L_t$ such that train t may travel from h to i on track (h, i) with travel time $d_t(h, i)$, A_t contains travel arcs of the form (v, u), one for each $v \in W(h, r) \cap V_t$ and $u \in U(i, r) \cap V_t$ such that $v \preceq u$ and $\Delta(v, u) = d_t(h, i)$;
2. *stop arcs at stations*: let $S_t \subseteq S$ be the set of stations visited by train $t \in T$, f_t its first station and l_t its last station. For each intermediate station $i \in S_t \setminus \{f_t, l_t\}$, consider the (possibly null) minimum stopping time $ds_t(i)$, for train t in i. For each track r_1 and each track r_2 such that train t may arrive at i on r_1 and depart from i on r_2, A_t contains station arcs of the form (u, v) for each $u \in U(i, r_1) \cap V_t$ and $v \in W(i, r_2) \cap V_t$ such that $u \preceq v$ and $\Delta(u, v) \geq ds_t(i)$;
3. *travel arcs through junctions*: let $J_t \subseteq J$ be the set of junctions visited by train $t \in T$ and o_j be the corresponding occupation time of junction $j \in J$. For each track r_1 and each track r_2 such that train t may arrive at $j \in J_t$ on r_1 and depart from $j \in J_t$ on r_2, A_t contains junction arcs of the form (u, v) for each $u \in U(i, r_1) \cap V_t$ and $v \in W(i, r_2) \cap V_t$ such that $\Delta(u, v) = o_j$;

Finally, A_t contains *starting* arcs (σ, v), for every node v corresponding to a possible departure node from the first station f_t of train t along any track that can be used by the train, and *ending* arcs (u, τ), for every node u corresponding to a possible arrival node at the last station l_t of train t along any track that can be used by the train.

Note that, for each train $t \in T$, $G_t = (V_t, A_t)$ is an acyclic directed graph and a timetable for train t corresponds to a path from σ to τ in G_t.

Let (σ, w) be the starting arc in a path of train $t \in T$. The shift is given by the absolute difference between the actual departure time $\theta(w)$ of train t and the ideal departure time of train t from its first station, i.e.

$$\nu_t(w) := |\theta(w) - \epsilon(t, f_t)|.$$

For each stop arc $(u, w) \in A^t$, associated with station $i \in S_t$, the stretch is given by the difference between the actual stopping time and the minimum stopping time of train t at station i, i.e.

$$\mu_t(u, w) := \Delta(u, w) - ds_t(i).$$

The total stretch of a train is given by the sum of the stretches of the arcs along its path.

Let ν_t be the shift of train t in its actual timetable and μ_t be its total stretch. The actual profit of $t \in T$ is computed as: $\pi_t - \omega_t^{+/-}\ \nu_t - \psi_t\ \mu_t$, where ω_t^+ (ω_t^- resp.) is used if shift ahead (behind resp.) is applied.

In order to define the profit of the actual timetable of a train $t \in T$, we assign profits to the arcs in the corresponding path. In particular, we associate to the starting arcs the ideal profit minus the penalty due to the shift that the train may get, and to the stop arcs the penalty due to the stretch that the train may get. More precisely, for each starting arc (σ, w) the profit is $p_{t(\sigma,w)} := \pi_t - \omega_t^{+/-}\nu_t(w)$, for each stop arc $(u, w) \in A_t$ the profit is $p_{t(u,w)} := -\psi_t\mu_t(u, w)$, while for each travel arc, junction arc and for each ending arc the profit is 0.

3.2. ILP formulation

For each train $t \in T$ and node $v \in V_t$, let $\delta_t^+(v)$ and $\delta_t^-(v)$ denote the sets of arcs in A_t leaving and entering node v, respectively.

For sake of clarity, we start presenting the model without the track operational constraints. Then, we present the additional constraints arising in our application.

3.2.1. Model without track operational constraints

We introduce a binary variable x_{ta}, for each train $t \in T$ and each arc $a \in A_t$ of the time-space graph defined above, which takes a value of one if the corresponding arc a is selected in the solution for the corresponding train t (and zero otherwise). The choice of an arc for a train in the solution represents either the travel of the train from a station to the next one, the stop of the train at a station or the travel of the train through a junction.

For convenience, in order to express the track operational constraints, we introduce a binary variable y_v for each node $v \in V$, which takes a value of one if the corresponding node is visited by some train in the solution (and zero otherwise), and a binary variable z_{tv} for train $t \in T$ and each node $v \in V$, which takes a value of one if the corresponding train t visits the corresponding node v in the solution (and zero otherwise).

The model reads as follows, where $\tilde{V}_t = V_t \setminus \{\sigma, \tau\}$ and $\tilde{V} = V \setminus \{\sigma, \tau\}$:

$$\max \sum_{t \in T} \sum_{a \in A_t} p_{ta} x_{ta} \tag{1}$$

$$\sum_{a \in \delta_t^+(\sigma)} x_{ta} \leq 1 \qquad t \in T \tag{2}$$

$$\sum_{a \in \delta_t^-(v)} x_{ta} = \sum_{a \in \delta_t^+(v)} x_{ta} \qquad t \in T, v \in \tilde{V}_t \tag{3}$$

$$z_{tv} = \sum_{a \in \delta_t^-(v)} x_{ta} \qquad t \in T, v \in \tilde{V}_t \tag{4}$$

$$y_v = \sum_{t \in T : v \in V_t} z_{tv} \qquad v \in \tilde{V} \tag{5}$$

$$x_{ta} \in \{0, 1\} \qquad t \in T, a \in A_t \tag{6}$$

$$z_{tv} \in \{0, 1\} \qquad t \in T, v \in V_t \tag{7}$$

$$y_v \in \{0, 1\} \qquad v \in V. \tag{8}$$

The goal is to maximize the total profit of the selected arcs. Constraints (2) ensure that at most one starting arc for each train is selected. Constraints (3) are flow conservation constraints: together with constraints (2) they require that for each train at most one path (timetable) in the time-space graph from σ to τ is selected. Constraints (4) and (5) are used to link the three types of variables. Finally, constraints (6), (7) and (8) require that the variables are binary. Note that constraints (7) are redundant, since they are implied by constraints (2) and (3) together with constraints (4) and (6) (they are only used to improve the readability of the model).

In the following we introduce the track operational constraints.

3.2.2. Departure/arrival constraints

These constraints impose that the headway times between consecutive departures (arrivals resp.) of trains from (at resp.) a station on a track are to be respected. They are expressed in the form of *clique* constraints. Indeed, they impose that at most one train can depart from (arrive at, resp.) a station i on a track r within a time interval of time length $\leq h^-(i,r)$ ($\leq h^+(i,r)$, resp.). For each station, each track, and each node representing a departure (arrival resp.) from that station using that track, we consider a time interval of length shorter than the minimum headway between two consecutive departures (arrivals resp.), starting at the time associated with the specific node. We then impose that at most one departure node (arrival node resp.) associated with this time interval can be visited in the solution. In particular, for the departure constraints, let i and r be the considered station and track, respectively, and $w \in W(i,r)$ be the considered departure node from station i on track r. We then consider all the nodes $v \in W(i,r)$, such that the time instant associated with node v is greater or equal than the time instant of node w, and for which the time distance $\Delta(w,v)$ from w is shorter than the minimum headway time $h^-(i,r)$ between two consecutive departures from station i on track r. I.e. w is the first node of the time interval of length shorter than the minimum headway time and v are all the nodes belonging to such interval. The departure constraints impose that at most one of these nodes can be visited by a train (i.e. at most one train can depart from one of these nodes) in the solution. These constraints are expressed for all stations, tracks and for all possible departure nodes $w \in W(i,r)$. The departure constraints read as follows:

$$\sum_{\substack{v \in W(i,r): w \preceq v, \\ \Delta(w,v) < h^-(i,r)}} y_v \leq 1, \qquad w \in W(i,r), i \in S, r \in R, \tag{9}$$

Arrival constraints are expressed in a similar way. Let i and r be the considered station and track, respectively, and $u \in U(i,r)$ be the considered arrival node at station i on track r. We then consider all the nodes $v \in U(i,r)$, such that the time instant associated with node v is greater or equal than the time instant of node u, and for which the time distance $\Delta(u,v)$ from u is shorter than the minimum headway time $h^+(i,r)$ between two consecutive arrivals at station i on track r. I.e. u is the first node of the time interval of length shorter than the minimum headway time and v are all the nodes belonging to such interval. The arrival constraints impose that at most one of these nodes can be visited by a train (i.e. at most one train can arrive at one of these nodes) in the solution. These constraints are expressed

for all stations, tracks and for all possible arrival nodes $u \in U(i,r)$.

The arrival constraints read as follows:

$$\sum_{\substack{v \in U(i,r): u \preceq v, \\ \Delta(u,v) < h^+(i,r)}} y_v \leq 1, \qquad u \in U(i,r), i \in S, r \in R. \tag{10}$$

Note that constraints (8) are redundant when departure/arrival constraints are included, since they are implied by constraints (5) together with constraints (9) and (10).

3.2.3. Station capacity constraints

These constraints require that, for each time instant of the considered time horizon, the maximum number of trains that can be simultaneously present at a station is respected. In particular, for each station $i \in S$ and for each time instant $\bar{q} \in H$, we impose that at most c_i trains can be simultaneously stopped at station i. This is expressed by imposing that at most c_i station arcs, representing a stop of a train at station i at time instant $\bar{q}$, can be selected in the solution for some trains.

The station capacity constraints read as follows:

$$\sum_{t \in T} \sum_{r \in R} \sum_{\substack{a=(u,v): \\ u \in U(i,r) \cap V_t, \\ v \in W(i,r) \cap V_t, \\ \theta(u) \leq \bar{q}, \theta(v) \geq \bar{q}}} x_{ta} \leq c_i, \qquad i \in S, \bar{q} \in H. \tag{11}$$

In constraints (11), arc $a = (u,v)$ is a station arc of station i for train t, representing the stop of train t at station i, since u belongs to the arrival nodes $U(i,r) \cap V_t$ and v belongs to the departure nodes $W(i,r) \cap V_t$. It represents the presence of train t at station i at time instant $\bar{q}$ since the arrival time $\theta(u)$ of t at i is smaller or equal to $\bar{q}$ while the departure time $\theta(v)$ of t from i is greater or equal to $\bar{q}$. Therefore, arc $a = (u,v) : u \in U(i,r) \cap V_t, v \in W(i,r) \cap V_t, \theta(u) \leq \bar{q}, \theta(v) \geq \bar{q}$ represents that train t is present at station i at time $\bar{q}$. In constraints (11), for each station $i \in S$ and for each time instant $\bar{q} \in H$, we sum over all trains, all tracks and all station arcs of these trains, the arc variables x_{ta} and impose that at most c_i of them can take value one, i.e. at most c_i trains can be simultaneously present at station i at time instant $\bar{q}$.

We would like to mention that stations are dealt with in a macroscopic way: we check the number of trains simultaneously present at a station, but we neglect the details on the platforming problem, which is itself an NP-hard problem (see [34]). Constraints similar to those that will be described in Section 3.2.5 could be used to avoid conflicting itineraries of trains inside stations.

3.2.4. No-overtaking constraints

These constraints ensure that no overtaking between trains is allowed along a track between two consecutive stations. They are expressed as clique constraints. Let R_t be the set of all tracks used by train $t \in T$. We consider each pair of trains t and k traveling along the same track $r \in R_t \cap R_k$ from station h to station i for which $d_t(h,i) \geq d_k(h,i)$ (i.e. k travels faster than t along r). Then we consider two departure nodes w_1 and w_2 (representing departures from station h) which are not compatible in the sense that k overtakes t if t uses

w_1 and k uses w_2. We also consider two departure nodes w_3 and w_4, where node w_3 is the first departure node from station h such that the minimum headway between two consecutive departures of trains t and k from h is respected, while w_4 is the first departure node from station h such that the minimum headway between two consecutive arrivals of trains t and k at i is respected. Hence, in order to obtain stronger constraints, we insert in the clique all the travel arcs of trains t and k between stations h and i such that either they represent k overtaking t or they are incompatible due to departure or arrival constraints.

Towards this end we further define the set of quadruples of nodes $\mathcal{O}(t,k,r)$ which contains the quadruple (w_1,w_2,w_3,w_4) for $w_1,w_3 \in W(h,r)\cap V_t$ and $w_2,w_4 \in W(h,r)\cap V_k$ if the following properties hold (according to the definitions of w_1, w_2, w_3 and w_4 above):

$$0 \leq \Delta(w_2,w_1) \leq d_t(h,i) - d_k(h,i),$$

$$\Delta(w_2,w_3) = h^-(h,r)$$

and

$$\Delta(w_1,w_4) = h^+(i,r) + d_t(h,i) - d_k(h,i).$$

The overtaking constraints read as follows:

$$\sum_{\substack{w\in W(h,r)\cap V_t:\\ w_1\preceq w\prec w_3}} z_{tw} + \sum_{\substack{w\in W(h,r)\cap V_k:\\ w_2\preceq w\prec w_4}} z_{kw} \leq 1,$$
$$t,k\in T, r=(h,i)\in R_t\cap R_k, d_t(h,i)\geq d_k(h,i),$$
$$(w_1,w_2,w_3,w_4)\in\mathcal{O}(t,k,r) \tag{12}$$

Note that we are able to express these constraints as clique constraints only on the departure nodes because of the fact that the travel times between consecutive stations is fixed for each train. We refer the reader to [8] for further details.

3.2.5. Junction constraints

These constraints are used to impose the correct transit of trains through junctions by using the notion of incompatible itineraries for trains traveling through the junction. In Figure 1, we show a schematic example of a junction, which connects four stations (A,B,C and D). Four trains (T_1, T_2, T_3 and T_4), each one at one of stations are shown. The arrows indicate the possible itineraries of the trains. We can see that the itinerary of train T_1 from station A, through the junction, to station D is incompatible with that of train T_3 from station B, through the junction, to station C. In particular, train T_1 blocks the junction (for all the occupation time) for train T_3, and, for safety reasons, train T_3 is stopped at station B, until the junction is free again. Similarly, the itinerary of train T_2 from station C, through the junction, to station B is incompatible with that of train T_4 from station D, through the junction, to station A. Note that not all itineraries shown in Figure 1 are incompatible with each other. For example, itineraries of trains T_2 from C to B and T_3 from B to C are compatible with each other. Therefore, the junction constraints need to be expressed for pairs of trains, as explained below.

Let $\mathcal{I}_j$ be the set of all possible itineraries traversing a junction $j\in J$ in the railway node. Each $I\in\mathcal{I}_j$ is defined as $I=\{s_{in}, r_{in}, j, r_{out}, s_{out}\}$, where s_{in} and s_{out} are stations, r_{in} a track

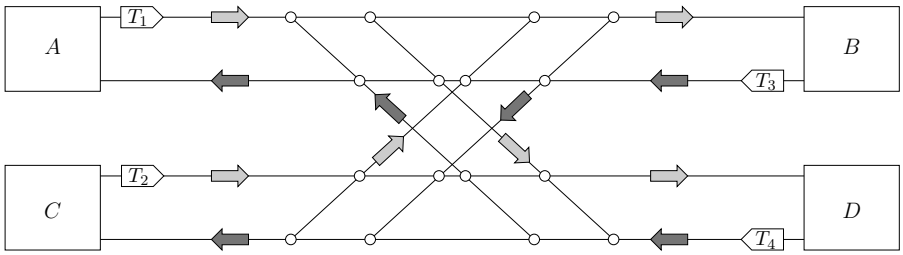

Figure 1: An example of a junction.

connecting s_{in} and j and r_{out} a track connecting j and s_{out}. For each junction $j \in J$, a set of pairs of incompatible itineraries $P_j = \{(I^{1a}, I^{1b}), \ldots, (I^{ma}, I^{mb})\}$ is then also given, where $(I^{ga}, I^{gb}) \in \mathcal{I}_j \times \mathcal{I}_j$ for $1 \leq g \leq m$.

Let $T_I \subseteq T$ be the subset of trains which uses itinerary $I \in \mathcal{I}_j$ ($j \in J$) in their ideal path. Given a junction $j \in J$, consider a pair of trains t and k such that $t \in T_{I^{ga}}$ and $k \in T_{I^{gb}}$ for some $(I^{ga}, I^{gb}) \in P_j$, i.e. consider two trains that can use a pair of incompatible itineraries traversing junction j. The junction constraint imposes that if, at time θ, train t travels from s_{in}^{ga} to j on r_{in}^{ga} and then continues to s_{out}^{ga} on r_{out}^{ga}, then from time θ to time $\theta + o_j$ train k cannot travel from s_{in}^{gb} to j on r_{in}^{gb} and then to s_{out}^{gb} on r_{out}^{gb}.

The junction constraints read as follows:

$$z_{tu} + \sum_{\substack{v \in W(s_{in}^{gb}, r_{in}^{gb}): \\ u \preceq v, \Delta(u,v) \leq o_j}} z_{kv} \leq 1,$$

$$j \in J, (I^{ga}, I^{gb}) \in P_j,$$

$$I^{ga} = \{s_{in}^{ga}, r_{in}^{ga}, j, r_{out}^{ga}, s_{out}^{ga}\}, I^{gb} = \{s_{in}^{gb}, r_{in}^{gb}, j, r_{out}^{gb}, s_{out}^{gb}\},$$

$$t \in T_{I^{ga}}, k \in T_{I^{gb}}, u \in U(j, r_{in}^{ga}). \tag{13}$$

With respect to the ILP model proposed in [8], this model includes new constraints, namely (11), taking into account the maximum number of trains simultaneously at each station, and (13), expressing the rules for incompatible itineraries inside the junctions.

4. Upper bound computation

In contrast to what is done in [16] and in [8], in which the ILP model for the TTP is relaxed in a Lagrangian way, we solve a LP-relaxation of the described model. We remove the redundant constraints (7) and (8), and we replace constraints (6) with the following ones:

$$0 \leq x_{ta} \leq 1 \qquad\qquad t \in T, a \in A_t. \tag{14}$$

We also further relax the obtained model by elimination of constraints (12) and (13).

The relaxed LP-model is given by (1)–(5), (9)–(11), (14). It is solved by the general purpose solver CPLEX [31] in order to derive an upper bound on the optimal solution of the problem. It is useful especially for evaluating the quality of the solutions found by the proposed heuristic algorithm, described in the next section.

5. Heuristic algorithm

Since the mathematical formulation is characterized by large numbers of constraints and variables, which prevents the use thereof in real-world applications, we decided to develop a heuristic algorithm for obtaining good quality solutions to the TTP in short computing times.

The algorithm is executed for a given number of iterations. At each iteration, trains are considered one at a time in a predetermined order (described below), and a schedule is computed as follows. Let $\{O_1, \ldots, O_{|T|}\}$ be the set of trains in the given order for the current iteration, and let O_t be the current train in the given order. A reduced time-space graph for train O_t is built dynamically, taking into account the trains $O_1, \ldots, O_{t-1}$ already scheduled and their corresponding timetables. In other words only the subset of arrival and departure nodes and subset of arcs that respect the track operational constraints are inserted in the reduced graph of train O_t. Given the reduced time-space graph for train O_t, and the arc profits as explained in Section 3.1, the maximum profit path from the source σ to the sink τ is computed by dynamic programming. Since the time-space graph for each train is acyclic, if we neglect the constraints on the maximum total stretch, the maximum profit path can be computed in polynomial time (see e.g. [2]). In the following, we describe the standard dynamic programming procedure that computes the maximum profit path without taking into account the maximum shift and maximum total stretch constraints. Then, we explain how we deal with these constraints.

In the dynamic programming procedure, we need to store for each node $v \in V_{O_t}$ the maximum profit of a path from the source σ to v, along with the predecessor node of v in the path, in order to be able to reconstruct it. More precisely, let O_t be the current train for which we are computing the maximum profit path, and $G^{red}_{O_t} = (V_{O_t}, A_{O_t})$ its reduced time-space graph. We associate a label $l(v) = (pf(v), pr(v))$ with each node $v \in V_{O_t}$, where $pf(v)$ represents the maximum profit of a path from the source σ to v and $pr(v)$ the predecessor node of v in the path.

Labels are initialized as follows. For all nodes connected to σ by a starting arc in A_{O_t}, we set $l(v) = (\pi_{O_t} - \omega^{+/-}_{O_t}\nu_{O_t}(v), \sigma)$, i.e. we set the profit of the node to be the ideal profit π_{O_t} of train O_t minus the shift penalty $\omega^{+/-}_{O_t}\nu_{O_t}(v)$ (see Section 3.1 for the starting arc profit definition), and its predecessor to be the source node σ. For each other node, we set $l(v) = (0, v)$.

We iteratively consider the next node v in V_{O_t}, according to the order given by the stations visited by train O_t and, for each station, in chronological order. From the current node v, if its label has a positive profit, we consider each node s_v of its successor nodes, i.e. of the nodes that are connected to v by an arc in A_{O_t}. We label the current successor node s_v as $l(s_v) = (pf(v) + p_{O_t(v,s_v)}, v)$, if $pf(v) + p_{O_t(v,s_v)} > pf(s_v)$, where $p_{O_t(v,s_v)}$ corresponds to the profit of arc (v, s_v) (see Section 3.1 for the arc profit definitions). The procedure is iterated until all the nodes in V_{O_t} have been considered.

If the sink node τ has a label with positive profit, we reconstruct the maximum profit path found, by iteratively going backward to the predecessor node stored in the label of the current node, until we reach node σ. If the sink node τ has a label with profit 0, it means that no feasible path exists for the current train (due to the conflicts with the already scheduled trains $O_1, \ldots, O_{t-1}$. If no such path exists, if the total profit of the path is negative or if it does not

respect the maximum total stretch for the train, the train is cancelled. Otherwise, the path is selected in the solution, and the corresponding train timetable is updated accordingly.

In the Appendix, we report the pseudocode for the dynamic programming procedure to compute the maximum profit path.

Note that we also need to take into account the constraints on the maximum shift and on the maximum total stretch for each train. The maximum shift can be dealt with in a very easy way: it consists of limiting the set of departure nodes of a train from its first station. Imposing a maximum total stretch is more difficult and increases the complexity of the dynamic programming procedure: finding a path of maximum profit respecting this constraint amounts to finding an optimal path in an acyclic graph with a resource constraint (see e.g. [25]). More precisely, we need to store, for each possible value s of the stretch until node v, the maximum profit of path (if any) from σ to v with total stretch s.

In order to keep the computational time limited we treat the constraints on the maximum total stretch in a heuristic way, by computing the maximum profit path while neglecting the maximum total stretch which is checked only at the end. If the obtained path respects the maximum total stretch (and has a positive profit), the path is selected in the solution, otherwise the train is cancelled. Based on extensive computational results, this choice turned out to be the best trade-off between the computational time of the proposed heuristic algorithm and the number of cancelled trains due to maximum total stretch (see Section 6.2 for further details on this issue).

The algorithm behaviour is highly dependent on the order in which the trains are examined, and the solutions can therefore be improved by repeating the algorithm with different initial orderings. We consider three different ordering schemes for ordering the trains:

1. *random order*: trains are ordered in a random way;
2. *random order with priority*: according to the importance given to trains by the Train Operator, they are partitioned into two categories, namely trains with high priority and trains with low priority. Within each category trains are then ordered in a random way;
3. *adaptive order*: trains are again partitioned into two categories, namely trains with high priority and trains with low priority. Trains which were cancelled in a previous iteration because of conflicts with other scheduled trains are then inserted at the top of the ordering of the category to which they belong. In this way it is more likely that they will be scheduled in the iterations to follow, leading to different (and hopefully better) solutions. This is an adaptive approach, which collects information on critical trains from the previous iterations and tries to adjust the decisions (i.e. the ordering of the trains) accordingly.

A computational comparison between these different orderings is carried out in the next section of the paper, in particular we will discuss the quality of the solutions that these different orderings allow to obtain.

6. Computational results

In this section we present computational results obtained by using real-world data provided by RFI. In particular, we consider four instances: the railway node of Milan with two

different levels of detail of the railway network and two different sets of trains to be scheduled (indicated as `Milan (a)` and `Milan (b)` in what follows), and the railway nodes of `Bologna` and `Florence`. We show in Figure 2 the network of the railway node of Milan.

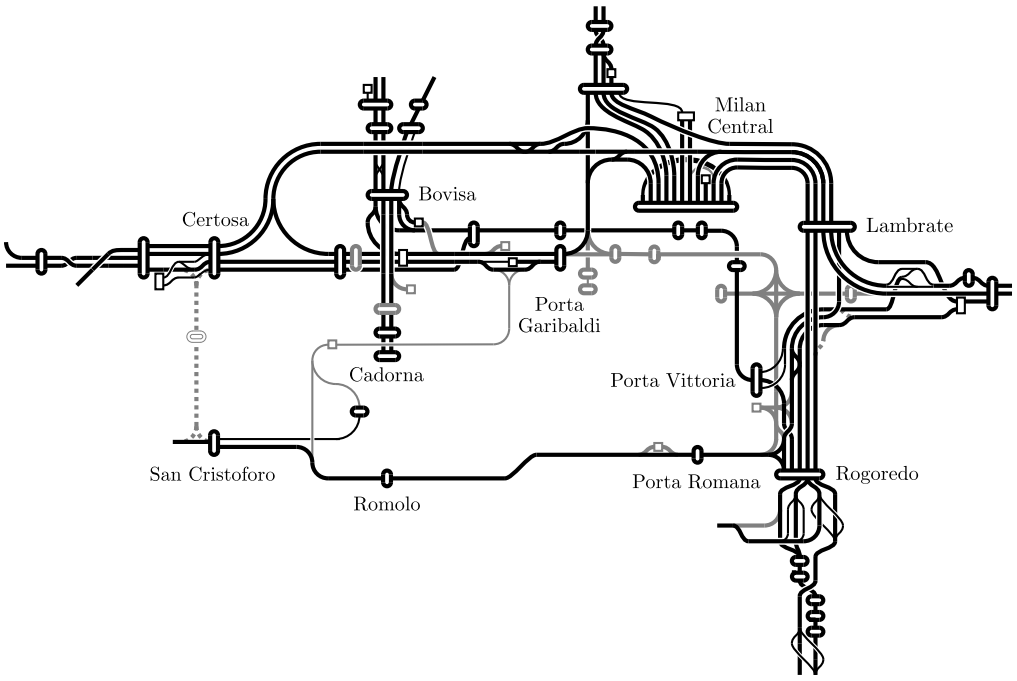

Figure 2: Railway node of Milan.

The considered time horizon is one day and the time discretization is one minute. The description of the instances is presented in Table 1. In particular, we show, for each instance, the number of stations and junctions, the number of trains, the number of different train types (such as high-speed trains, freight trains, etc), the number of tracks, the number of parallel tracks, the sum of all the profits of the trains, station capacity values, headway time values and junction occupation time values. The sum of all the profits of the trains would be obtained if all the trains would be scheduled according to their ideal timetables, and is therefore an upper bound on the optimal solution value. As it can be seen in Table 1, the railway node of Bologna contains a larger number of stations than the other ones. The reason for this is that the boundary of each railway node was determined by the practitioners, according to what is done in real-world practice. Different levels of detail for a node can also be considered, as is the case of the node of Milan.

The ideal profits for each train and the penalty weights for shifting and stretching are set according to [7], as shown in Table 2. Since additional train types are present in the instances of our case study, we agreed with RFI to classify them in one of the existing classes. The maximum shift ahead of schedule is set to 1 minute, the maximum shift behind schedule is set to 10 minutes and the maximum total stretch is set to 2 minutes, according to what was proposed by RFI.

The algorithm was implemented in C++ and run in Linux Ubuntu (compiled using g++ with option -O3) on an Intel(R) Core(TM) i7-3770 CPU computer with 8 processors each clocked at 3.40GHz, and with a total of 16 GB of RAM.

	Stations	Junctions	Trains	Train types	Tracks	Parallel tracks	Profits
`Milan (a)`	44	6	1500	17	134	16	164300
`Milan (b)`	53	10	1544	19	146	14	170510
`Bologna`	222	26	1080	17	490	12	125300
`Florence`	8	1	36	4	22	4	5120

	Station capacity values	Headway time values (minutes)	Junction occupation time values (minutes)
`Milan (a)`	$[2, 17], 24$	$3, 6$	5
`Milan (b)`	$[2, 16], 24$	$[3, 6]$	4
`Bologna`	$[2, 10], 27$	2	2
`Florence`	$2, 3, 9, 17, 30$	6	5

Table 1: Instance details

Train type	Eurostar	Euronight	Intercity	Express	Combined	Direct	Local	Freight
pi_t	200	150	120	110	100	100	100	100
ω_t^+/ω_t^-	7	7	6	5	6	5	5	2
ψ_t	10	10	9	8	9	8	6	3

Table 2: Profits and penalty weights for each train type

6.1. Relaxation of the ILP model

In this section we present the characteristics of our model (number of variables and constraints), and the upper bound values obtained by relaxing different subsets of constraints, with the corresponding computational times. Our goal is to show the trade-off between upper bound quality and computational times.

Table 3 shows, in the first column, the subset of considered constraints of model (1)-(5), (9), (10), (11), (14). For each instance, in Table 3, we report the number of variables, the number of constraints according to the selected subset (shown in the first column), the value of the upper bound obtained and the corresponding computing time (expressed in seconds). As expected, an increase in the number of constraints leads to a larger computing time but also to a better upper bound. Note that, if we consider only the subset of constraints (1)-(3), (14), the obtained upper bound coincides with the sum of the profits of all the trains, since no constraint is imposed except from computing a set of maximum profit paths for all the trains.

In the following section, we show the percentage gaps between the heuristic solution values and the best upper bounds of Table 3.

Constraint sets	Number of variables	Number of constraints	UB Value	Time	Number of variables	Number of constraints	UB Value	Time
(1)–(3), (14)		848566	164300	33		665738	170510	16
(1)–(3), (5), (9), (10), (14)	7234403	1421930	154457	8190	3509648	1249210	162071	1607
(1)–(3), (5), (9)–(11), (14)		1460723	154030	132426		1288998	160058	6550
		Milan (a)				Milan (b)		
(1)–(3), (14)		1259725	125300	54		6794	5120	1
(1)–(3), (5), (9), (10), (14)	6527895	2729915	124831	69	34909	13852	5021	2
(1)–(3), (5), (9)–(11), (14)		2903492	124831	178		14320	5021	2
		Bologna				Florence		

Table 3: Upper bounds obtained by relaxing different subset of constraints.

6.2. Heuristic solutions

In this section, we present the heuristic solutions obtained by applying the heuristic algorithm presented in Section 5 with three different orderings of trains. In particular, we consider two different configurations: the first one runs the algorithm for 100 iterations, while the second one for 10000 iterations. These two configurations are used to show the applicability of the proposed heuristic in different situations. In particular, 100 iterations can be used if the Infrastructure Manager wants to analyze several different scenarios, corresponding to different train timetable requests from the Train Operators. On the contrary, 10000 iterations can be used to refine the solution. We would like to mention that the configuration with 100 iterations shows the applicability of our method to real-time train rescheduling. The extension of the proposed algorithm to this context will be investigated as future research. In the latter case, however, additional real-life constraints need to be taken into account, which go beyond the scope of this paper.

In Table 4, we show, for each instance, the obtained heuristic solution values. The first column indicates the ordering rule, while the second column shows the considered number of iterations. For each instance we then show the heuristic solution value (Value), the corresponding computing time (expressed in seconds) and the percentage gap with respect to the best upper bound value (UB Value) computed as in Section 6.1 (Gap=((UB Value - Heur sol Value)/UB Value)*100). In addition, we report the percentage (Canc.) of trains cancelled and the number (h.p.) of high priority trains cancelled.

A higher number of iterations leads to better results for all instances. The computing time also increases as the number of iterations increases, but even for 10000 iterations the time remains less than half an hour, which is reasonable when dealing with the planning problem. If the results are required in a shorter computing time, we can set the limit to 100 iterations and still obtain good results, which shows that the developed algorithm is of practical use. The negotiation between the Infrastructure Manager and the Train Operators in the planning phase can be a long process, requiring many rounds before reaching an adequate solution. Keeping the computing time limited to half an hour at most is therefore fundamental for practical use.

The adaptive order gives rise to the best results for the instances `Milan (a)`, `Milan (b)` and `Bologna`, both in the case of 100 and 10000 iterations. The random order reaches the best solution for the instance `Florence`. However, using the adaptive order, the percentage gap is very small (1.57%).

In Table 4, we also report the percentage of cancelled trains and the number of high priority cancelled trains. As can be seen, a larger number of iterations generally reduces the number of cancelled trains. In all cases, a larger number of iterations leads to a better value of the objective function. Recall that the aim is to maximize the sum of the train profits, i.e. changing the ideal timetables as little as possible. In addition, we are able to obtain solutions where no high priority trains are cancelled for all instances. This is a significant result since priorities are set by the Train Operators according to the passenger demands, and it is therefore important to be able to meet their requests, especially on high priority train scheduling. In Section 5, we explained that we treat the constraints on the maximum total stretch in a heuristic way. We would like to mention that the number of cancelled trains due to maximum total stretch violation is very low for all the instances (at most 4 for the `Milan (b)` instance).

Ordering Rule	Iter.	Value	Time	Gap	Canc.	h.p	Value	Time	Gap	Canc.	h.p
`Random Order`	100	136434	16	11.42	11.40	10	147043	9	8.13	8.68	7
	10000	137433	1647	10.78	11.13	6	147760	844	7.68	8.81	9
`With Priority`	100	137151	17	10.96	11.47	0	147990	9	7.54	8.74	0
	10000	137811	1642	10.53	11.33	0	148621	842	7.15	9.13	0
`Adaptive Order`	100	138413	16	10.14	10.13	0	149642	8	6.51	8.29	0
	10000	138845	1657	9.86	9.93	0	150000	835	6.28	7.58	0
		`Milan (a)`					`Milan (b)`				
`Random Order`	100	117302	15	6.03	1.85	12	4921	0	1.99	2.78	0
	10000	117909	1473	5.55	1.67	8	5018	6	0.06	0.00	0
`With Priority`	100	118561	15	5.02	1.30	0	4942	0	1.57	2.78	0
	10000	118955	1468	4.71	0.83	0	4949	5	1.43	2.78	0
`Adaptive Order`	100	118836	15	4.80	0.65	0	4942	0	1.57	2.78	0
	10000	119031	1459	4.65	1.02	0	4942	5	1.57	2.78	0
		`Bologna`					`Florence`				

Table 4: Heuristic solutions obtained with different train orderings and different number of iterations.

The results presented in Table 4 are plotted in Figure 3. In particular, we report for each train ordering the percentage gap between the best upper bound and the current best solution found versus the number of iterations. The plots are shown for each instance. We can see that the adaptive order outperforms the other orders for all instances except `Florence`, which is, however, also the smallest instance. In the first iterations (up to 1000) the decrease of the percentage gap is significant, while during the next iterations the improvement is less evident.

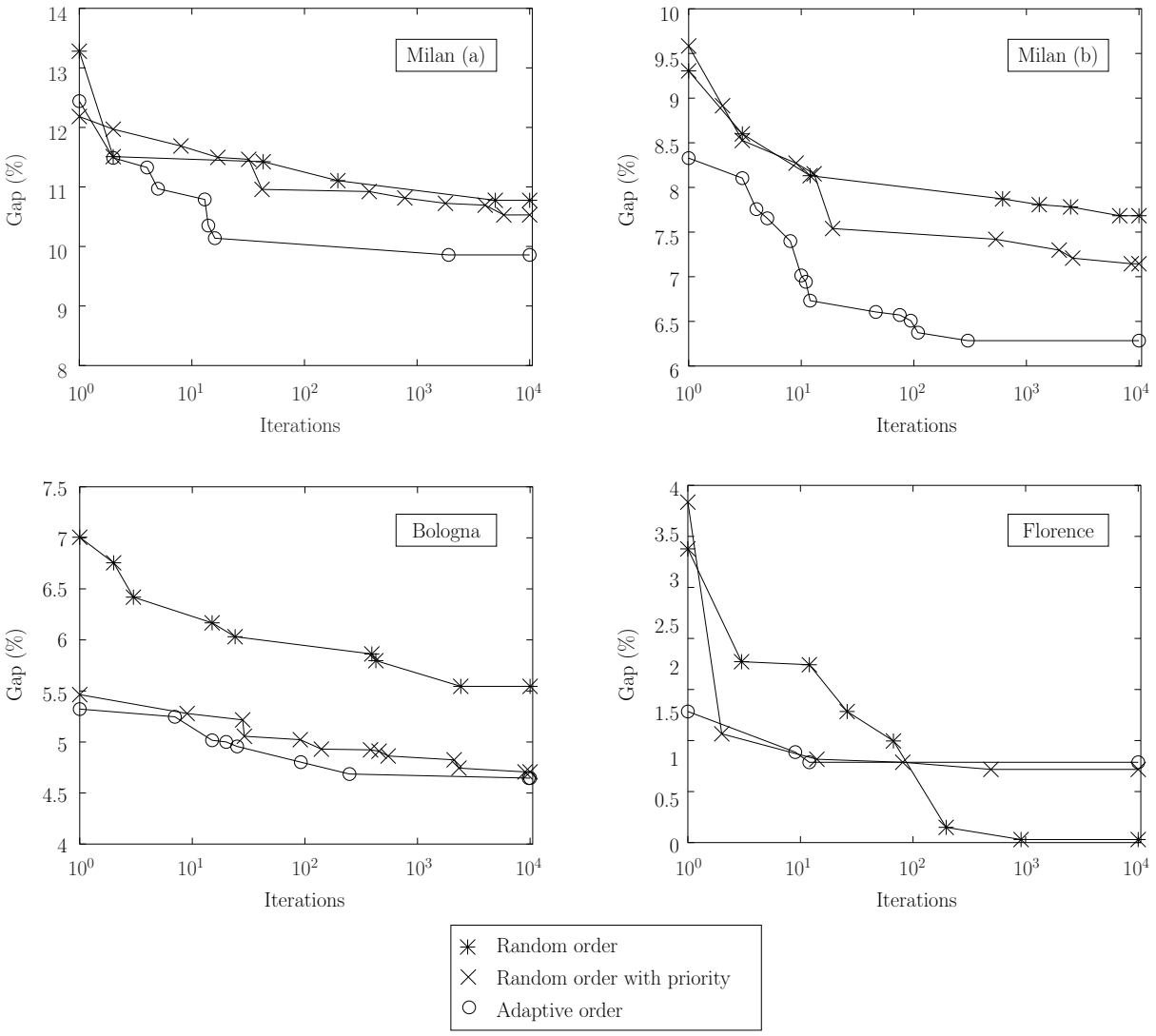

Figure 3: Plots of the percentage gap with respect to the upper bound at each iteration in which the current best solution was improved.

6.3. Practicality of the solutions

In this section, we investigate some issues concerning the applicability of the solutions found by the proposed heuristic algorithm in practice. We consider the timetables obtained by using the adaptive ordering method and 10000 iterations.

Recall that, as mentioned in Section 3.2.3, the capacity of the stations is only considered as the maximum number of trains simultaneously present at the station, neglecting the specific itineraries that each train should follow to reach a specific platform. When the capacity of a station is saturated (i.e. the number of trains simultaneously present at the station is equal to the number of its platforms), it is more difficult to find a platform assignment that avoids conflicting itineraries. In order to show the practicality of the solutions found for our real-world instances, we evaluate, for each station, how often the number of trains simultaneously present at the station equals its capacity. For each of the four real-world instances, we count the total number of time instants of station capacity saturation in the final timetables (if two

stations are saturated at the same time instant, we count two instants of station capacity saturation). It turns out that the station capacity is saturated in at most 1.1% of the time instants, which occurs for `Milan (b)` instance. For the other instances, the station capacity is saturated in less than 0.8% of the time instants. This is an indication of the practicality of the obtained timetables, even if the train platforming problem must be solved to validate the timetables at a microscopic level and to assign specific platforms to trains.

A second issue that we analyze is the robustness of the obtained timetables to delay propagation. We estimate the effects of delay scenarios on the timetables obtained for the railway node of `Bologna`, for which a record of historical delays occurred in practice was available from RFI. The first of the recorded delays occurs at 04:45 and the last one at 22:10. Each delay is caused by a specific reason, such as delay of the crew, longer stop at a station, broken rolling stock unit, or delay because of a connection with another delayed train. Delay durations range between 60 and 2460 seconds, but most delays are around 120-400 seconds.

Since we have a different timetable from the one on which the historical delays occurred, we used the historical data in order to generate delays to be applied to the timetable obtained by the proposed algorithm. In particular, we divided the day in intervals of 30 minutes each, and evaluated, for each time interval and station, the minimum and maximum delay durations, occurred in the historical data. Then, we generate the delay scenarios as follows. We randomly select, with uniform probability, a station and a time interval in which a delay occurred. Given the station and the time interval, we generate a delay for a train (if any) that departs from that station in the given time interval, with duration uniformly distributed between the minimum and maximum delay durations computed. Each delay scenario contains a single delayed train. Overall we generated 100 delay scenarios.

For each delay scenario, we evaluate the effect on the timetable as follows. When a delay occurs, some conflicts between trains can arise. In this case, it is necessary to reschedule trains in order to obtain a feasible timetable. To this aim, we again use our heuristic algorithm with adaptive order. In this case, we keep as fixed all the train timetables before the delay time and allow to shift trains only later than the scheduled departure time in the planned timetable. In addition, we set the minimum stopping time of each train at each of its visited stations as in the planned timetable (and not as in the ideal one). To emulate a real-time context, the algorithm is executed until either a time limit of two minutes is reached or no trains are cancelled. In more than 90% of the scenarios, the computing time is less than 5 seconds, while, in the remaining case, the longest computing time is 77 seconds (i.e. the algorithm stops because there are no cancelled trains). The maximum shift and the maximum stretch are both set to 5 minutes. We underline that the proposed algorithm is not an effective rescheduling algorithm, but is used here for train rescheduling only to evaluate the robustness of the obtained timetable to delay propagation. The development of a rescheduling algorithm goes beyond the scope of this paper and it will be investigated as future research.

In order to analyze delay propagation, we compute the total additional delay of the rescheduled trains with respect to the planned timetable, by summing all the delays of trains (except from the originally delayed one) at their final stations. Clearly, a smaller delay corresponds to a more robust timetable. It turns out that for the large majority (77 out of 100) of the scenarios there are no additional trains delayed at their final stations. In the remaining cases (23 out of 100), most scenarios have one additional delayed train, while the worst scenario has 4 additional delayed trains. We also compute the maximum additional

delay cumulated by a train at its final station in each of the 23 scenarios in which at least one train has an additional delay. In most cases, the maximum delay is at most two minutes, while the higher maximum additional delay among all scenarios corresponds to 5 minutes. This computationally shows that the obtained timetable is robust to delays of this kind.

Finally, we evaluate, for each of the 23 scenarios and for each delayed train, the percentage of stations, with respect to the total number of stations visited by the train, in which the train is delayed. In particular, we distinguish trains that are delayed in:

- $< 25\%$ of the stations they visit,
- $\geq 25\%$ and $< 50\%$ of the stations they visit,
- $\geq 50\%$ and $< 75\%$ of the stations they visit,
- $\geq 75\%$ and $\leq 100\%$ of the stations they visit.

It turns out that, overall the 23 scenarios, 5.77% of the delayed trains fall in the first category, 19.23% in the second one, 21.15% in the third one and 53.85% in the last one. This shows that about half of the delayed trains are delayed in many of the visited stations. One reason for this is certainly that the rescheduling algorithm does not allow to increase the speed along the tracks nor to decrease the stopping time with respect to the planned timetable. Therefore, if a train is delayed at its first station, it continues to be delayed (even if no additional delay occurs) until it reaches its final station. This further motivates the need for an effective rescheduling algorithm.

6.4. Capacity saturation

This section is dedicated to the evaluation of capacity saturation of the railway nodes, i.e. we investigate the maximum number of trains that can be scheduled in the railway node. Note that this is different from the station capacity saturation described in the previous section, which concerns single stations only and not the entire railway node.

6.4.1. Literature review

A common definition of capacity of a single railway line is given by the ratio between a given time period and the minimum headway time between two consecutive trains. This represents the total number of trains that can traverse a critical section in the given time period and is called bottleneck approach (see [24], [6]). When dealing with a railway node, this definition has to be extended as it cannot be simply determined as the sum of the capacities of the lines inside the node. Several types of trains must be taken into account and their specific schedules, which lead to interactions and conflicts. A complete analysis can show the efficiencies and weaknesses of the railway infrastructure.

Several works studied the evaluation of railway infrastructure capacity. In [24], the capacity of a railway junction is determined as the maximum number of trains, among a predefined set of trains, that can operate on the junction. The assumption that trains do not stop during the run in the junction is made and a microscopic detail of the junction is considered. Two methods are developed: the first one is a constraint programming approach solved by a greedy heuristic; the second one is a set-packing problem solved by an adapted GRASP metaheuristic. In [6], complex railway networks are analyzed and several aspects related to

the infrastructure capacity, such as different train types, their directions and lengths, the headway and dwell times, are taken into account. The authors define the capacity as the maximum number of trains that can traverse the entire railway or certain bottleneck sections in a given duration of time. The developed approach consists of an optimization approach, based on the bottleneck method. A case study is used to illustrate the proposed technique. In [1], a survey of methods to evaluate railway capacity is presented. They include analytical, optimization and simulation methods. The authors present a system that embeds analytical and optimization approaches. An analysis is also conducted that shows how different parameters affect railway capacity. In particular, train speed, train heterogeneity, headway times and buffer times are considered.

6.4.2. Capacity saturation evaluation

We propose to use the developed heuristic algorithm for capacity saturation evaluation of the four instances, in the scenario corresponding to the set of trains given by RFI. In order to estimate the capacity, we consider two different parameter settings. In the first one, the profit of the trains are increased to 100000: in this way, we can avoid train cancellations caused by null or negative profits due to shift and stretch changes. In the second one, in addition to profit increase, maximum shift and maximum stretch values are increased to 20 minutes each to allow more trains to be scheduled.

In Table 5, we report the results of capacity saturation analysis, which were obtained by executing the proposed heuristic algorithm for 10000 iterations using adaptive ordering of the trains. For each instance, we show the percentage of scheduled trains in the standard configuration and the corresponding computing time (in seconds). For the cases of increased train profits and increased train profits together with increased maximum shift and maximum stretch values, we report the percentage of additional scheduled trains. As it can be seen from Table 5, additional trains can be scheduled for the instances `Milan (a)`, `Milan (b)` and `Bologna`, both when increasing profits and when allowing for larger maximum shift and maximum stretch. On the contrary, the node of `Florence` shows to be saturated. The results are plotted in Figure 4, in which we plot the percentage of scheduled trains at each iteration in which the number of scheduled trains was improved for each instance. The percentage of scheduled trains ranges between 93% and 100% for the four instances. This indicates that the considered railway nodes are properly sized for the current traffic since at least 90% of the train requests can be scheduled on each node.

	Standard configuration		Increased profits		Increased profits, max shift and max stretch	
Instance	% sched.	Time	% add.	Time	% add.	Time
`Milan (a)`	90.07	1656	1.27	1664	3.00	2625
`Milan (b)`	92.42	842	0.58	843	3.82	2655
`Bologna`	98.98	1459	0.83	1464	1.02	5109
`Florence`	97.22	5	0.00	5	0.00	18

Table 5: Results for TTP saturation using the adaptive ordering method for 10000 iterations.

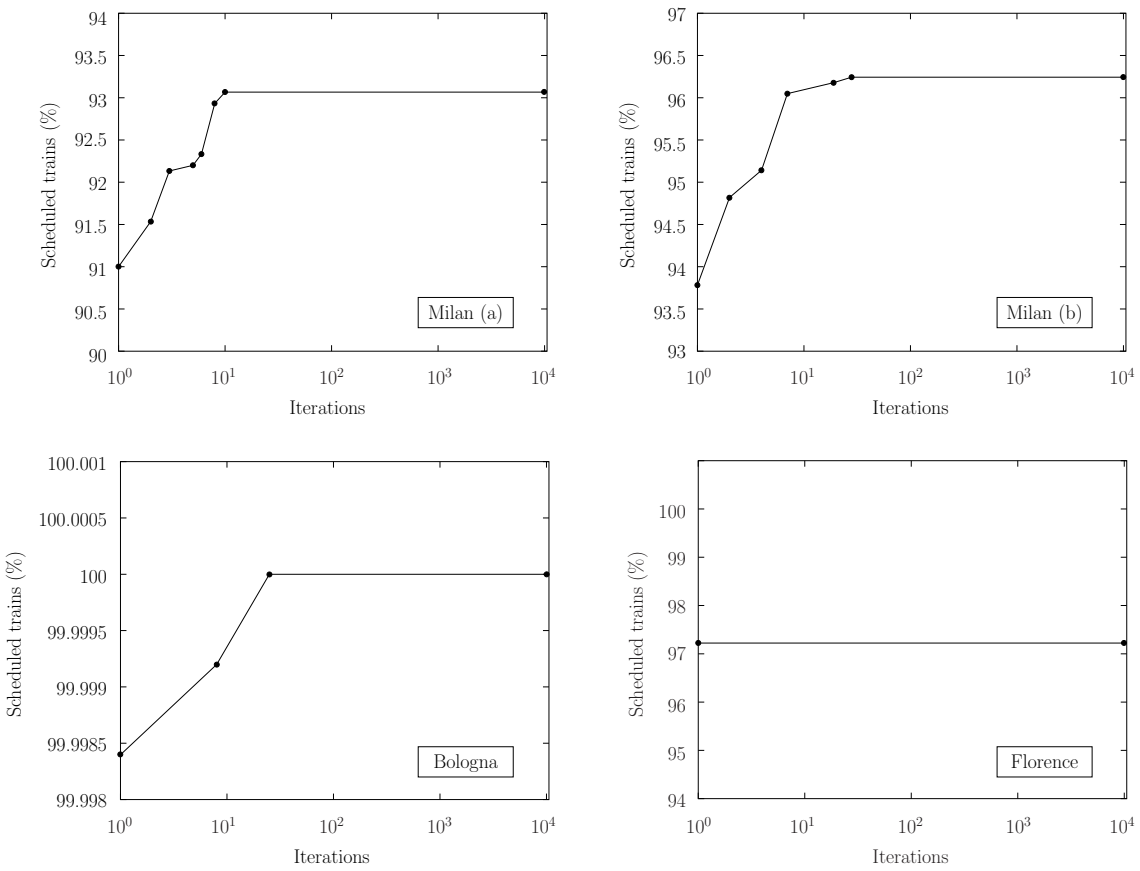

Figure 4: The percentage of scheduled trains with increased profits, and maximum shift and maximum stretch, at each iteration in which the number of scheduled trains was improved.

However, if additional trains need to be scheduled, the capacity might not be sufficient, and additional tracks may be required. In this case, the analysis can be used to determine where additional tracks should be placed in order to avoid bottlenecks. We evaluated the effect of adding (from 1 to 5) additional tracks in bottleneck areas of the railway nodes. These bottleneck areas are determined as follows. For every two stations $s_1, s_2 \in S$ and each time instant $h \in H$, let $\mathcal{T}(s_1, s_2, h)$ give the number of trains which are scheduled to be present on the track section between s_1 and s_2 at time h, which may be seen as a measure of the congestion of that track section at that time. However, $\sum_{h \in H} \mathcal{T}(s_1, s_2, h)$ is simply the sum of the travel times of all trains traveling between s_1 and s_2, and does not reflect the degree to which there are conflicts on this track section. Alternatively we therefore propose $\binom{\mathcal{T}(s_1,s_2,h)}{2}$ as a measure of congestion, i.e. the number of pairs of possible conflicts between trains for that time instant. A measure of the total congestion of the track section between stations s_1 and s_2 throughout the considered time horizon is therefore considered to be

$$\sum_{h \in H} \binom{\mathcal{T}(s_1, s_2, h)}{2}.$$

In Table 6, we show the percentage of trains scheduled when additional tracks are inserted in the railway node. It may be noted that the percentage of scheduled trains does not change

significantly when we insert additional tracks. Therefore, we can conclude that there is not a problematic bottleneck area in the railway node, and in order to be able to schedule more trains several changes in the infrastructure would be needed.

	Number of additional tracks					
Instance	0 track	1 track	2 tracks	3 tracks	4 tracks	5 tracks
`Milan (a)`	90.35	90.35	90.35	90.35	90.35	90.35
`Milan (b)`	92.20	92.20	92.20	92.80	93.73	93.73
`Bologna`	95.37	95.37	95.46	95.46	95.46	95.46
`Florence`	97.22	97.22	97.22	97.22	97.22	97.22

Table 6: Percentage of trains scheduled when additional tracks are added to the railway node.

7. Conclusions and future research

This paper deals with the Train Timetabling Problem in a railway node. We consider the point of view of the Infrastructure Manager, who collects requests from the Train Operators of scheduling trains in the railway node, according to so-called ideal timetables, which in turn are proposed by the Train Operators based on passengers and freight demand. The goal is to determine conflict-free timetables, which differ as little as possible from the ideal ones, while respecting several track operational constraints. The problem was studied for a research project funded by Rete Ferroviaria Italiana, the main Italian railway Infrastructure Manager.

We have proposed an Integer Linear Programming model for the problem, which is based on a time-space graph adapted from previous models in the literature for dealing with the railway nodes. The mathematical formulation is characterized by a large number of constraints and variables, which prevents the use thereof in real-world applications. Therefore, we have proposed a relaxation of the model for deriving dual bounds and a heuristic algorithm which obtains good solutions to real-world instances with up to 1500 trains in short computing times. In addition, we have showed the usefulness of the presented heuristic algorithm in capacity saturation analysis.

Future research will be devoted to the extension of the described approach to deal with real-time timetable rescheduling in cases where delays or disruptions cause infeasibilities in the planned timetables. This is an important issue, since the increasingly utilized railway systems experience frequent delays, and an efficient use of the infrastructure is crucial at an operational level. Another research area is to investigate how to embed the proposed approach in a complex scheduling system, in which timetables are first determined for railway corridors and then railway nodes are considered in a closed loop framework in order to derive improved timetables for the entire railway network.

Acknowledgments

We are grateful to Pier Luigi Guida and his group at RFI for the fruitful discussion on the topic and for having provided us with the instances for our tests. We would like to thank Vito Achille of RFI for having provided us with the delay scenarios of the railway node of Bologna. The research on the robustness issues described in this paper has been developed within the ON-TIME project (European Union's Seventh Framework Programme).

[1] Abril, M., Barber, F., Ingolotti, L., Salido, M. A., Tormos, P., and Lova, A., 2008, An assessment of railway capacity. Transportation Research Part E, 44(5), 774-806.

[2] Ahuja, R.K., Magnanti, T.L., and Orlin, J.B., 1993, Network flows: Theory, Algorithms, and Applications. Prentice Hall, New Jersey.

[3] Blanco, V., Puerto, J., and Ramos, A. B., 2011, Expanding the Spanish high-speed railway network. Omega, 39(2), 138-150.

[4] Borndörfer, R., Erol, B., Graffagnino, T., Schlechte, T., and Swarat, E., 2012, Optimizing the Simplon railway corridor, Annals of Operations Research, 1-14.

[5] Brännlund, U., Lindberg, P.O., Nöu, A. and Nilsson, J.E., 1998, Allocation of scarce track capacity using lagrangian relaxation, Transportation Science, 32, 358–369.

[6] Burdett, R. L., and Kozan, E., 2006, Techniques for absolute capacity determination in railways. Transportation Research Part B, 40(8), 616-632.

[7] Cacchiani, V., Caprara, A. and Toth, P., 2008, A column generation approach to train timetabling on a corridor, 4OR, 6, 125–142.

[8] Cacchiani, V., Caprara, A. and Toth, P., 2010, Scheduling extra freight trains on railway networks, Transportation Research Part B, 44, 215–231.

[9] Cacchiani, V., Caprara, A., and Toth, P. (2010). Non-cyclic train timetabling and comparability graphs. Operations Research Letters, 38(3), 179-184.

[10] Cacchiani, V., Huisman, D., Kidd, M., Kroon, L., Toth, P., Veelenturf, L., and Wagenaar J., 2014, An Overview of Recovery Models and Algorithms for Real-time Railway Rescheduling, Transportation Research Part B, 63, 15-37.

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

[12] Cadarso, L., Marín, Á., and Maróti, G., 2013, Recovery of disruptions in rapid transit networks. Transportation Research Part E, 53, 1533.

[13] Cai, X., and Goh, C.J., 1994, A fast heuristic for the train scheduling problem, Computers and Operations Research 21, 499-510.

[14] Caimi, G., Fuchsberger, M., Laumanns, M., and Lüthi, M., 2012, A model predictive control approach for discrete-time rescheduling in complex central railway station areas. Computers and Operations Research, 39, 2578-2593.

[15] Caprara, A., 2010, Almost 20 Years of Combinatorial Optimization for Railway Planning: from Lagrangian Relaxation to Column Generation, In Thomas Erlebach and Marco Lübbecke (eds.), Proceedings of the 10th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS), Schloss Dagstuhl, Germany, 1-12.

[16] Caprara, A., Fischetti, M. and Toth, P., 2002. Modeling and solving the train timetabling problem. Operations Research, 50, 851–861.

[17] Caprara, A., Kroon, L., Monaci, M., Peeters, M., and Toth, P., 2006. Passenger railway optimization. Handbooks in operations research and management science, 14, 129-187.

[18] Caprara, A., Kroon, L., and Toth, P., 2011. Optimization problems in passenger railway systems. Wiley Encyclopedia of Operations Research and Management Science 6, 3896-3905.

[19] Caprara, A., Monaci, M., Toth, P., and Guida, P. L., 2006, A Lagrangian heuristic algorithm for a real-world train timetabling problem. Discrete Applied Mathematics, 154(5), 738-753.

[20] Carey, M. and Lockwood D., 1995, A model, algorithms and strategy for train pathing, Journal of the Operational Research Society, 46, 988–1005.

[21] Cordeau, J. F., Toth, P., and Vigo, D., 1998, A survey of optimization models for train routing and scheduling, Transportation science, 32(4), 380-404.

[22] D'Ariano, A., Corman, F., Pacciarelli, D., and Pranzo, M., 2008, Reordering and local rerouting strategies to manage train traffic in real time, Transportation Science, 42(4), 405419.

[23] D'Ariano, A., Pacciarelli, D., and Pranzo, M., 2007, A branch and bound algorithm for scheduling trains on a railway network. European Journal of Operational Research, 183(2), 643657.

[24] Delorme, X., Rodriguez, J., and Gandibleux, X., 2001, Heuristics for railway infrastructure saturation. Electronic Notes in Theoretical Computer Science, 50(1), 39-53.

[25] Desrosiers, J., Dumas, Y., Solomon, M. M., Soumis, F., 1995, Time constrained routing and scheduling. In Ball, M.O., Magnanti, T.L., Monma, C.L., Nemhauser, G.L. (eds.), Handbooks in Operations Research and Management Science, Network Routing, vol. 8. Elsevier, Amsterdam, 35-139.

[26] Dollevoet, T., Huisman, D., Schmidt, M., and Schöbel, A., 2012, Delay management with rerouting of passengers, Transportation Science, 46, 7489.

[27] Furini, F., and Kidd, M.P., 2013, A fast heuristic approach for train timetabling in a railway node, Electronic Notes in Discrete Mathematics, 41, 205-212.

[28] Harrod, S., and Schlechte, T., 2013, A direct comparison of physical block occupancy versus timed block occupancy in train timetabling formulations, Transportation Research Part E, 50-66.

[29] Higgings, A., Kozan, E., and Ferreira, L., 1997, Heuristic techniques for single line train scheduling, Journal of Heuristics 3, 4362.

[30] Huisman, D., Kroon, L.G., Lentink, R.M., and Vromans, M.J.C.M., 2005, Operations Research in passenger railway transportation. Statistica Neerlandica, 59, 467497.

[31] IBM ILOG, (2014) CPLEX Optimizer 12: CPLEX `www.ibm.com/software/integration/optimization/cplex-optimizer/`.

[32] Jovanovic, D., and Harker, P.T., 1991, Tactical scheduling of rail operations: the SCAN I system, Transportation Science 25, 4664.

[33] Kroon, L., Huisman, D., Abbink, E., Fioole, P. J., Fischetti, M., Maróti, G., and Ybema, R., 2009, The new Dutch timetable: The OR revolution. Interfaces, 39(1), 6-17.

[34] Kroon, L., Romeijn, H., and Zwaneveld, P., 1997, Routing trains through railway stations: Complexity issues. European Journal of of Operational Research, 98, 485-498.

[35] Liebchen, C., and Möhring, R., 2007, The modeling power of the periodic event scheduling problem: Railway timetables-and beyond, In: F. Geraets et al. (Eds.), Algorithmic Methods for Railway Optimization, Lecture Notes in Computer Science, vol. 4359, Springer, 3-40.

[36] Liebchen, C., Proksch, M., and Wagner, F.H., 2007, Performance of algorithms for periodic timetable optimization, In: M. Hickman, P. M irchandani, S. Voss (Eds.), Computer-Aided Systems in Public Transport (CASPT 2004), LNEMS, vol. 600, Springer, 151-180.

[37] Lindner, T., and Zimmermann, U.T., 2005, Cost optimal periodic train scheduling, Mathematical Methods of Operations Research 62, 281295.

[38] Lusby, R. M., Larsen, J., Ehrgott, M., and Ryan, D., 2011, Railway track allocation: models and methods. OR spectrum, 33(4), 843-883.

[39] Lusby, R. M., Larsen, J., Ehrgott, M., and Ryan, D., 2013, A set-packing inspired method for real-time junction train routing. Computers and Operations Research, 40, 713724.

[40] Mannino, C., and Mascis, A., 2009, Optimal real-time traffic control in metro stations, Operations Research, 57(4), 10261039.

[41] Mascis, A., and Pacciarelli, D., 2002, Job-shop scheduling with blocking and no-wait constraints, European Journal of Operational Research, 143(3), 498517.

[42] Nachtigall, K., and Voget, S., 1996, A genetic algorithm approach to periodic railway synchronization, Computers and Operations Research 23, 453463.

[43] Odijk, M., 1996, A constraint generation algorithm for the construction of periodic railway timetables, Transportation Research Part B 30, 455464.

[44] Peeters, L.W.P., 2003, Cyclic Railway Timetable Optimization, Ph.D Thesis, Erasmus Research Institute of Management, Erasmus University, Rotterdam, 2003.

[45] Peeters, L.W.P., and Kroon, L.G., 2001, A cycle based optimization model for the cyclic railway timetabling problem, In: J. Daduna, S. Voss (Eds.), Computer-Aided ransit Scheduling, Lecture Notes in Economics and Mathematical Systems, vol. 505, Springer-Verlag, Berlin, 275296.

[46] Serafini, P., and Ukovich, W., 1989, A mathematical model for periodic event scheduling problems, SIAM Journal on Discrete Mathematics 2, 550581.

[47] Szpigel, B., 1973, Optimal train scheduling on a single track railway, in: M. Ross (Ed.), OR72, North-Holland, Amsterdam, 343351.

[48] Törnquist, J., 2012, Design of an effective algorithm for fast response to the rescheduling of railway traffic during disturbances, Transportation Research Part C, 20, 6278.

[49] Yang, L., Li, K., Gao, Z., and Li, X., 2012, Optimizing trains movement on a railway network. Omega, 40(5), 619-633.

[50] Yu, M. M., and Lin, E. T. (2008). Efficiency and effectiveness in railway performance using a multi-activity network DEA model. Omega, 36(6), 1005-1017.

Appendix

In Figure 5, we present the dynamic programming procedure for the maximum profit path computation on the acyclic time-space graph.

O_t: current train
$G^{red}_{O_t} = (V_{O_t}, A_{O_t})$: reduced time-space graph for O_t
for $v \in V_{O_t}$ **do**
  if *(σ,v)* is a starting arc of A_{O_t} **then**
    $l(v) := (\pi_{O_t} - \omega^{+/-}_{O_t} \nu_{O_t}(v), \sigma)$
  end
  else
    $l(v) := (0, v)$
  end
end
//nodes are evaluated according to the order given by the stations visited by train O_t and, for each station, in cronological order
for $v \in V_{O_t}$ **do**
  if $l(v) \neq (0, v)$ **then**
    for $s_v \in V_{O_t} : (v, s_v) \in A_{O_t}$ **do**
      if $pf(v) + p_{O_t(v,s_v)} > pf(s_v)$ **then**
        $l(s_v) := (pf(v) + p_{O_t(v,s_v)}, v)$
      end
    end
  end
end
if $l(\tau) \neq (0, \tau)$ **then**
  return found path for O_t
end
else
  no feasible path for O_t
end

Figure 5: Dynamic programming procedure for computing the maximum profit path of a train.