EURO J Transp Logist (2015) 4:285–320
DOI 10.1007/s13676-014-0046-4

TUTORIAL

A tutorial on non-periodic train timetabling and platforming problems

Valentina Cacchiani · Laura Galli · Paolo Toth

Received: 24 June 2013 / Accepted: 25 February 2014 / Published online: 18 March 2014

Abstract In this tutorial, we give an overview of two fundamental problems arising in the optimization of a railway system: the train timetabling problem (TTP), in its *non-periodic* version, and the train platforming problem (TPP). We consider for both problems the planning stage, i.e. we face them from a tactical point of view. These problems correspond to two main phases that are usually optimized in close sequence by the railway infrastructure manager. First, in the TTP phase, a *schedule* of the trains in a railway network is determined. A schedule consists of the arrival and departure times of each train at each (visited) station. Second, in the TPP phase, one needs to determine a stopping *platform* and a *routing* for each train inside each (visited) station, according to the schedule found in the TTP phase. Due to the complexity of the two problems, an integrated approach is generally hopeless for real-world instances. Hence, the two phases are considered separately and optimized in sequence. Although there exist several versions for both problems, depending on the infrastructure manager and train operators requirements, we do not aim at presenting all of them, but rather at introducing the reader to the topic using small examples. We present models and solution approaches for the two problems in a didactic way and always refer the reader to the corresponding papers for technical details.

To the memory of our friend and colleague Alberto Caprara.

V. Cacchiani · P. Toth (✉)
DEI, University of Bologna, Viale Risorgimento 2, 40136 Bologna, Italy
e-mail: paolo.toth@unibo.it

V. Cacchiani
e-mail: valentina.cacchiani@unibo.it

L. Galli
Dipartimento di Informatica, University of Pisa, Largo B. Pontecorvo 3, 56127 Pisa, Italy
e-mail: galli@di.unipi.it

Keywords Train timetabling · Train platforming · Integer programming · Heuristic · Branch-and-cut-and-price

Mathematics Subject Classification (2000) 90-01 · 90C10 · 90C90 · 90C06

1 Introduction

Railway systems are full of challenging combinatorial optimization problems. Yet, due to their complexity, the optimization process is usually carried out in sequence, subdividing the full problem into several subproblems, which are solved one after the other (i.e. the output of a problem becomes the input of the following one). These subproblems are NP-hard and include (in order): line planning, train timetabling, train platforming, rolling stock circulation, train unit shunting and crew planning, see e.g. Caprara et al. (2007) for an overview on passenger railway optimization problems. In Fig. 1, we show how the subproblems depend on one another.

In this tutorial, we concentrate on two early phases, namely *train timetabling problem* (TTP) and *train platforming problem* (TPP). We consider for both problems the planning stage, i.e. we face them from a tactical point of view. In this case, a solution is generally used for the following 6 months or 1 year. Both problems are solved by the railway infrastructure manager, who is responsible for train and infrastructure planning, and real time traffic control. In particular, we consider the *non-periodic* version of TTP, because models and solution approaches for the periodic version are rather different and deserve to be treated separately. These two phases come right after the line planning problem; hence, they assume that the routes for the trains as well as the types and frequencies of the trains on each route have been defined. TTP and TPP are solved in sequence and are strictly connected, i.e. the two phases are generally iterated until both solutions are accepted by the infrastructure manager. The TTP is solved for a set of trains on a given railway network to determine a feasible *schedule*. Then, the TPP is solved for each station of the railway network in order to assign each train a stopping *platform* and define the corresponding *routing* inside the station. Note that TTP is solved for the whole network, while TPP is solved separately for each individual station in the

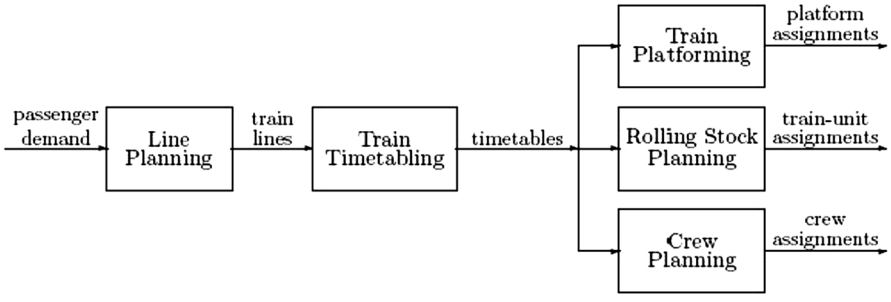

Fig. 1 Main problems solved in the planning of a passenger railway system

network, since each station is independent of the others, from a platforming point of view.

This tutorial is not meant to cover all versions nor all existing approaches of TTP and TPP. In fact, our goal is to present the main concepts of the two problems and some well-assessed models and algorithms. The tutorial aims at giving an overview of the two problems in a didactic way, presenting a description of the problems through examples, mixed integer programming (MIP) models and exact as well as heuristic solution methods based on branch-and-price. Apart from the direct application of such models and algorithms to the above mentioned problems, we believe that they also represent "paradigms", in the sense that some concepts can be adapted and applied to different contexts.

The tutorial is organized into two parts, the first one devoted to TTP and the second one to TPP. Inside each part, we present a brief review of the literature (Sects. 2.1, 3.1) and the problem description, specifying the input, the constraints and the objective that are taken into account (Sects. 2.2, 3.2). Then, in Sect. 2.4, we present two integer linear programming (ILP) models for the TTP (based on a graph representation of the problem presented in Sect. 2.3). In Sect. 3.3, we present an integer quadratic programming (IQP) model for TPP. All the presented models are characterized by either an exponential or a very large number of variables. Finally, we present solution methods to find optimal and heuristic solutions to the problems (Sects. 2.5, 3.4). For each problem, a set of examples is presented. In Appendix 5, we report the list of variables and parameters used in the described models for TTP and TPP.

2 The train timetabling problem

2.1 Literature review

Train timetabling problem (TTP) has received considerable attention in the literature. We refer the reader to Cacchiani (2009), Cacchiani and Toth (2012), Caprara (2010), Caprara et al. (2007, 2011b), Harrod (2012), Huisman et al. (2005) and Lusby et al. (2011) for recent surveys. Two are the main variants of train timetabling: one is to consider a periodic (or cyclic) schedule of the trains that is repeated every given time period (e.g. every hour), and the other one is to consider a more congested network and/or a competitive market for which a non-periodic schedule becomes more appropriate. Indeed, in the latter case, several train operators run trains on the same infrastructure and it becomes harder to obtain effective periodic schedules. The periodic timetabling was introduced by the seminal paper (Serafini and Ukovich 1989), in which a mathematical model for the periodic event scheduling problem is proposed. Many extensions of the model have been considered in the literature, most of them based on cycle bases (see e.g. Kroon and Peeters 2003; Liebchen and Möhring 2007; Liebchen et al. 2007; Lindner 2000; Lindner and Zimmermann 2005; Nachtigall 1994; Odijk 1996; Peeters 2003; Peeters and Kroon 2001; Schrijver and Steenbeek 1994). We refer the reader to Liebchen (2006) for a survey and tutorial on periodic train timetabling. One of the

first papers dealing with the non-periodic TTP is presented in Szpigel (1973), in which a job-shop scheduling formulation for TTP on a single-track railroad is proposed. Other works on non-periodic TTP propose exact methods (Cacchiani et al. 2008; Jovanovic and Harker 1991) or heuristic methods (Brännlund et al. 1998; Burdett and Kozan 2010; Cai and Goh 1994; Caprara et al. 2006; Carey and Lockwood 1995; Fischer et al. 2008; Higgings et al. 1997; Oliveira and Smith 2000). There are works dealing with a single one-way line (Cacchiani et al. 2008, 2010a, 2013; Cai and Goh 1994; Higgings et al. 1997; Szpigel 1973) or with a general railway network (Borndörfer et al. 2006; Borndörfer and Schlechte 2008; Cacchiani et al. 2010b; Fischer et al. 2008; Fischer and Helmberg 2013).

Another important classification of TTP consists in the distinction between nominal problem and robust problem. In the nominal case, the goal is to determine optimal timetables for a set of trains providing the maximum efficiency of the railway system, e.g. scheduling as many trains as possible on the network or obtaining the shortest travel time for the passengers between origin and destination. In the robust case, the aim is to determine robust timetables for the trains, i.e. to find a schedule that avoids, in case of disruptions in the railway network, delay propagation as much as possible. Therefore, the objectives of the latter variants are in contrast and usually a trade-off between them must be achieved. The robust version of TTP has been studied by stochastic programming (Kroon et al. 2008), light robustness (Fischetti and Monaci 2009; Fischetti et al. 2009), recoverable robustness (Cicerone et al. 2009; Liebchen et al. 2009), delay management (Liebchen et al. 2010) and bi-objective methods(Cacchiani et al. 2012; Schlechte and Borndörfer 2010; Schöbel and Kratz 2009).

Recently, many works have been developed in the field of train timetable rescheduling, concerning recovery models and algorithms for real time railway delay and disruption management. This problem corresponds to the operational stage of TTP (see e.g. Cacchiani et al. 2014). When delays or disruptions occur, the timetables determined for the trains during the planning phase can become infeasible because conflicts can arise. It is therefore necessary to compute a new schedule for the trains, which is needed in a very short time, and which should be as close as possible to the original plan. Much research is connected with the Alternative Graph model introduced in Mascis and Pacciarelli (2002). Exact models and branch-and-bound algorithms (see e.g. D'Ariano et al. 2007; Mannino and Mascis 2009; Schöbel 2009), as well as heuristic algorithms (see e.g. D'Ariano et al. 2008; Dollevoet et al. 2012; Lusby et al. 2013) have been developed.

Another important distinction in TTP is between passenger trains and freight trains. Often the schedule for the passenger trains is computed first, while the freight train timetables are determined afterwards. When computing the schedule for the passenger trains, usually empty capacity slots are left for the freight trains, in order to ensure that they can be scheduled, based on freight demand and statistical data. Different objectives and constraints can arise. For example, in the case of freight trains, it is not important to build a periodic timetable, while having a periodic schedule for passenger trains is very convenient for passengers. One goal for passengers is to minimize their travel times from origin to destination, while this is less important for freight trains and detours from the high-speed lines can be used.

The peak hours are usually used to serve passenger traffic, and stops in minor stations are also desired, as opposed to the case of freight trains. We refer the reader to Caprara et al. (2007) for passenger railway optimization and to Cordeau et al. (1998) for further details on freight train scheduling.

In the recent years, research has been devoted to studying the integration of two or more subproblems of the optimization of a railway system. In particular, TTP has been integrated with other phases both during the planning phase and in real time rescheduling. We refer the reader to Barber et al. (2008) and to Lindner (2000) for the integration with line planning, to Cadarso and Marín (2012) for the integration with rolling stock assignment in the planning phase and to Adenso-Díaz et al. (1999) in the rescheduling phase, to Veelenturf et al. (2012) and Walker et al. (2005) for the integration with crew rescheduling in real time.

2.2 Problem description

In this tutorial, we focus on non-periodic nominal train timetabling both for passenger and freight trains (from now on shortly TTP) and take the point of view of the infrastructure manager. TTP calls for providing a timetable for a set of trains on a railway network, which satisfy the so-called *track capacity constraints*. An infrastructure manager is in charge of handling the railway network and receives requests from several train operators for scheduling trains to be operated for a given time horizon. Each of these requests specifies a path for a train along with the arrival and departure times for all stations along the path. Generally, these requests are mutually incompatible. Therefore, the infrastructure manager has to modify the arrival and/or departure times of some trains (and possibly to cancel some other trains), in order to come up with a proposed feasible solution for the train operators. The latter may either accept it or come up with new proposals. The process is iterated until the solution proposed by the infrastructure manager is accepted by all train operators.

For the sake of clarity, we present the case study of TTP on a single one-way line, called main corridor. In fact, in many cases, once the timetable for the trains on the main corridor has been determined, it is relatively easy to find a convenient timetable for the trains on the other lines of the network. The extension to a general railway network can easily be done (see e.g. Cacchiani et al. 2010b).

In the following, we give a formal definition of TTP, by specifying its input, constraints and objective.

2.2.1 Problem input: railway topology and trains timetables

In TTP, one needs to specify the railway *topology* (in our case a single, one-way track corridor linking two major stations, with a number of intermediate stations in between) together with a set of trains that are candidate to be run everyday of a given time horizon along the corridor. Let $S = \{1, \ldots, s\}$ represent the set of stations, numbered according to the order in which they appear along the corridor for the running direction considered, and $T = \{1, \ldots, t\}$ denote the set of candidate trains. For each train $j \in T$, a first (departure) station f_j and a last (destination)

station l_j ($l_j > f_j$) are given. Let $S^j := \{f_j, \ldots, l_j\} \subseteq S$ be the ordered set of stations visited by train j ($j \in T$). A *timetable* defines, for each train $j \in T$, the departure time from f_j, the arrival time at l_j, and the arrival and departure times for the intermediate stations $f_j + 1, \ldots, l_j - 1$. Each train is assigned by the train operator an *ideal timetable*, representing the most desirable timetable for the train. The ideal timetable defines a path for a train along with the arrival and departure times for all stations along the path. It represents the preferred schedule for the train, according to the train operator who associates an ideal timetable and an ideal profit to each train, based on the passenger and freight demands, on the lines of the railway network and on the available budget.

We present an example of a TTP instance that will be used through the TTP sections. The railway topology that we consider is shown in Fig. 2 and consists of a corridor with five stations ($S = \{1, 2, 3, 4, 5\}$).

Let $T = \{A, B, C\}$ be the set of trains. For each train, we have the following stations: $S^A = \{1, 2, 3\}$, $S^B = \{1, 2, 3, 4, 5\}$ and $S^C = \{3, 4, 5\}$. Consequently, we have: $f_A = 1$, $l_A = 3$, $f_B = 1$, $l_B = 5$, $f_C = 3$, $l_C = 5$. The ideal timetables provided on input are presented in Table 1. For example, train A departs from station 1 at 9:00, arrives at station 2 at 9:05 where it stops until 9:07 and arrives at station 3 at 9:18. As you can see, trains can visit a subset of stations or all of them.

2.2.2 Problem constraints

The *track capacity constraints* impose that:

- overtaking between trains occurs only within a station (as we are dealing with a corridor),
- for each station $i \in S$, a minimum time interval a_i between two consecutive arrivals of trains must be respected,
- for each station $i \in S$, a minimum time interval d_i between two consecutive departures of trains must be respected.

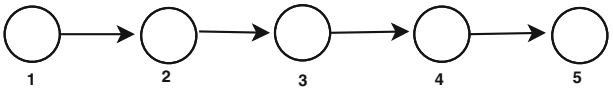

Fig. 2 Considered corridor with five stations

Table 1 Example of three ideal timetables

	Ideal timetable A		Ideal timetable B		Ideal timetable C	
Stations	Arr. time	Dep. time	Arr. time	Dep. time	Arr. time	Dep. time
1		9:00		9:00		
2	9:05	9:07	9:10	9:12		
3	9:18		9:30	9:35		9:33
4			10:00	10:03	10:02	10:07
5			10:20		10:24	

The minimum time interval between arrivals or departures is called *headway time*. Note that the headway times are checked at stations, i.e. two trains must respect a minimum time distance when arriving at (departing from resp.) a station, but they are related to the capacity of the corridor. In particular, constraints on the headway times implicitly impose a minimum time interval between two consecutive trains in the track connecting two consecutive stations. Let us consider $a_i = 4$ min for all stations $i \in S \setminus \{1\}$ and $d_i = 2$ min for all stations $i \in S \setminus \{s\}$ in our instance. An example of violated overtaking, arrival and departure constraints is shown in Fig. 3. The two parallel lines correspond to two consecutive stations s_1 and s_2 of the corridor. The nodes on the upper line (lower line resp.) correspond to time instants in which a train departs from (arrives at resp.) station s_1 (s_2 resp.). The corresponding time instants are shown in the figure (time increases from left to right). The arrows between nodes indicate the travel of trains from station s_1 to station s_2. On the left picture of Fig. 3, we can see that the two trains are overtaking each other since the train that leaves at 9:00 arrives later (at 9:09) than the train that leaves at 9:02 (which arrives at 9:05). Note that arrival and departure constraints are satisfied for the left picture. In the middle part of Fig. 3, the two trains are arriving at station s_2 too close in time to each other (only 3 min instead of 4 are in between the two arrivals). On the right picture, we can see that the two trains are departing from station s_1 too close in time: only 1 min is in between the two departures (instead of 2 min).

Besides the track capacity constraints, additional constraints can arise, such as manual block signaling (for managing a train on a track segment between two consecutive stations), station capacities (i.e. maximum number of trains that can be present in a station at the same time), prescribed timetable for a subset of the trains (which is imposed when some of the trains are already scheduled on the railway line and additional trains are to be inserted), or maintenance operations (that keep a track segment occupied for a given period). We refer the reader to Caprara et al. (2006) for a detailed description on how to deal with these constraints.

The ideal timetables given on input may be modified in order to satisfy the track capacity (and eventually additional) constraints. In particular, one is allowed to modify (anticipate or delay) the departure time of each train from its first station, and to increase (but not decrease) the stopping time interval at the (intermediate) stations. The first change is called *shift* and the latter change is called *stretch*. Moreover, one can cancel the train if it is not profitable (see Sect. 2.2.3). We consider the version of the problem in which the travel time of a given train between

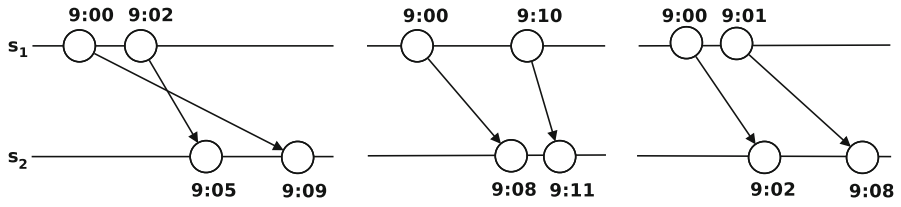

Fig. 3 Examples of violated overtaking, arrival and departure constraints

consecutive stations cannot be modified. In particular, the travel time is fixed to the minimum one. In this case, it is possible to write the overtaking constraints in the ILP in a form that is much stronger for the linear programming (LP) relaxation than in the version with variable travel times. Moreover, consider any two consecutive stations s_1 and s_2 and the track connecting them: increasing the stopping time of a train at station s_1 can be used to approximate the increase in speed along the track that connects the two stations. We refer to Caprara et al. (2002) and Caprara et al. (2006) for a comparison with the version in which this time can be increased as well. The timetable obtained in the solution will be referred to as the *actual timetable*.

2.2.3 Problem objective

The objective is to change as little as possible the ideal timetables given on input. This is obtained by defining a profit for each train and maximizing the sum of the profits of the scheduled trains, defined as follows. The *profit* achieved for each train $j \in T$ is given by $\pi_j - \alpha_j v_j - \gamma_j \mu_j$, i.e. the ideal profit is decreased by considering the penalties due to shift or stretch changes, where:

- π_j represents the *ideal profit* that is the profit achieved if the train travels according to its ideal timetable: it can be thought as the amount of money that the train operator is willing to pay if the train is scheduled according to his request, and therefore, it is proportional to the importance given to the train;
- v_j represents the *shift* that is the absolute difference between the departure times from station f_j in the ideal and actual timetables;
- μ_j represents the *stretch* that is the (non-negative) difference between the travel time from f_j to l_j in the actual and ideal timetables (equal to the sum of the stopping time increases over all intermediate stations);
- α_j, γ_j are given non-negative parameters, which are expressed as profit loss per time unit.

If the profit achieved by a train becomes null or negative due to the shift and/or stretch changes, the train is cancelled.

In Table 2, we show an example of ideal profits, and shift and stretch penalties for the three trains of our instance. These values are assigned to the trains by the train operators. If a train operator assigns a high profit and small shift and stretch penalties to a train, it means that he wants the train to be scheduled, even if with a timetable substantially different from the ideal one. On the contrary, if a train operator assigns a high profit and high shift and stretch penalties, it means that he wants the train to be scheduled according to its ideal timetable. Therefore, these values reflect the importance that the train operator gives to each train and to its ideal schedule. In Table 3, we show three examples of actual timetables for train A, derived by shifting and/or stretching its ideal timetable. The profits of the actual timetables are shown in the last row of the table. Actual timetable$_1$ is subject to shift of 2 min, and therefore, its profit corresponds to the ideal profit of 200, decreased by $6 * 2$. Actual timetable$_2$ receives a stretch of 1 min at station 2 and its profit

Table 2 Example of ideal profits and shift/stretch penalties

	Train A	Train B	Train C
Ideal profit π	200	100	200
Shift penalty α	6	2	10
Stretch penalty γ	10	4	20

Table 3 Example of actual profits corresponding to different actual timetables for train A

Stations	Actual timetable$_1$ A		Actual timetable$_2$ A		Actual timetable$_3$ A	
	Arr. time	Dep. time	Arr. time	Dep. time	Arr. time	Dep. time
1		9:02		9:00		9:02
2	9:07	9:09	9:05	9:08	9:07	9:11
3	9:20		9:19		9:22	
Actual profits	200 − 6*2		200 − 10*1		200 − 6*2 − 10 *2	

corresponds to $200 - 10$. Finally, Actual timetable$_3$ is shifted of 2 min and stretched of 2 min at station 2, getting an overall profit of $200 - 6 * 2 - 10 * 2$.

2.3 Graph representation

In this section, we outline the representation of the problem on a graph. This is a very common way of representing the problem, which is also very convenient for deriving an ILP model for it. Times are here discretized in minutes and expressed as integers from 1 to $q := 1{,}440$ (the number of minutes in a day), although a finer discretization would also be possible without changing the model, the computing times and the core memory requirements of the associated algorithms could increase considerably.

Let $G = (V, A)$ be the (directed, acyclic) space-time multigraph in which nodes represent arrivals/departures at/from a station in given time instants, and, for each train $j \in T$, a path from a node associated with station f_j to a node associated with station l_j represents a timetable for train j. The node set V has the form $\{\sigma, \tau\} \cup (U^2 \cup \ldots \cup U^s) \cup (W^1 \cup \ldots \cup W^{s-1})$, where

- σ and τ are an *artificial source node* and an *artificial sink node*, respectively;
- set U^i, $i \in S \setminus \{1\}$ represents the set of time instants in which some train can arrive at station i; the nodes in $U^2 \cup \ldots \cup U^s$ are called *arrival* nodes;
- set W^i, $i \in S \setminus \{s\}$, represents the set of time instants in which some train can depart from station i; the nodes in $W^1 \cup \ldots \cup W^{s-1}$ are called *departure* nodes.

Let $\theta(v)$ be the time instant associated with a given node $v \in V$. Moreover, let $\Delta(u, v) := \theta(v) - \theta(u)$ if $\theta(v) \geq \theta(u)$, and $\Delta(u, v) := \theta(v) - \theta(u) + q$ otherwise. That is, the time distance $\Delta(u, v)$ between two nodes u and v is expressed as the

difference between their corresponding time instants, taking into account the periodicity of the time horizon. It is useful to define a precedence relation between nodes for expressing the track capacity constraints in a mathematical way. 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 cyclic time interval between $\theta(v)$ and $\theta(u)$ is not smaller than the cyclic time interval between $\theta(u)$ and $\theta(v)$].

Note that not all time instants correspond to possible arrivals/departures of a given train j at a station $i \in S^j$. Accordingly, let $V^j \subseteq \{\sigma,\tau\} \cup (U^{f_j+1} \cup \ldots \cup U^{l_j}) \cup (W^{f_j} \cup \ldots \cup W^{l_j-1})$ denote the set of nodes associated with time instants corresponding to possible arrivals/departures of train j in a positive-profit timetable.

The arc set A is partitioned into sets $A^1,\ldots,A^t$, one for each train $j \in T$. In particular, for every train $j \in T$, A^j contains

- a set of *starting arcs* (σ,v), for each $v \in W^{f_j} \cap V^j$, whose profit is $p_{(\sigma,v)} := \pi_j - \alpha_j v(v)$, with $v(v) := \min\{\Delta(v^*,v), \Delta(v,v^*)\}$, where v^* is the node associated with the departure of train j from station i in the ideal timetable. I.e. we associate the ideal profit minus the possible shift penalty to the starting arcs;
- a set of *segment arcs* (v,u), for each $i \in S^j \setminus \{l_j\}$, $v \in W^i \cap V^j$ and $u \in U^{i+1} \cap V^j$ such that $\Delta(v,u)$ is equal to the travel time of train j from station i to station $i+1$, whose profit is $p_{(v,u)} := 0$;
- a set of *station arcs* (u,v), for each $i \in S^j \setminus \{f_j,l_j\}$, $u \in U^i \cap V^j$ and $v \in W^i \cap V^j$ such that $\Delta(u,v)$ is at least equal to the minimum stop time of train j in station i, whose profit is $p_{(u,v)} := -\gamma_j \mu(u,v)$, with $\mu(u,v) := \Delta(u,v) - \Delta(u^*,v^*)$, where u^* and v^* are the nodes associated, respectively, with the arrival and departure of train j at station i in the ideal timetable. I.e. we associate the stretch penalty to the station arcs;
- a set of *ending arcs* (u,τ), for each $u \in U^{l_j} \cap V^j$, whose profit is $p_{(u,\tau)} := 0$.

To satisfy the track capacity constraints, one should impose that certain pairs of arcs, associated with different trains, cannot be selected in the overall solution.

In Fig. 4, we show an example of the described multigraph for a corridor of 3 stations. For station 1, the set of departure nodes W^1 is shown. For station 2, the sets of arrival nodes U^2 and departure nodes W^2 are shown. For station 3, the set of arrival nodes U^3 is shown. The figure also shows the artificial source node σ and the artificial sink node τ, the starting, segment, station and ending arcs corresponding to two alternative paths (timetables) for a single train.

In Fig. 5, we show a graph representation of our instance presenting the ideal timetables for trains A, B and C. We highlight in bold the conflicts arising between the ideal timetables. In particular, train A and B depart at the same time (9:00), therefore violating the minimum headway of 2 min between two departures. Trains C and B overtake each other between stations 3 and 4. Finally, the arrivals of trains B and C at station 4 are too close in time (only 2 min apart from each other instead of at least 4 min).

As previously mentioned, in order to obtain a feasible solution, we can modify the ideal timetables of the trains by shifting and/or stretching them. In Table 4, we

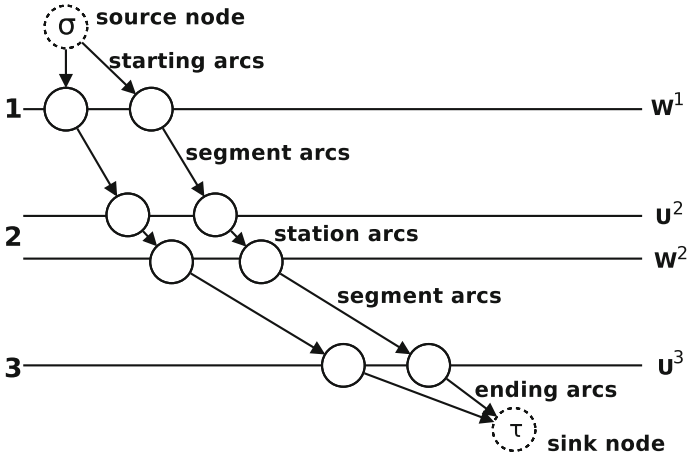

Fig. 4 Multigraph representation

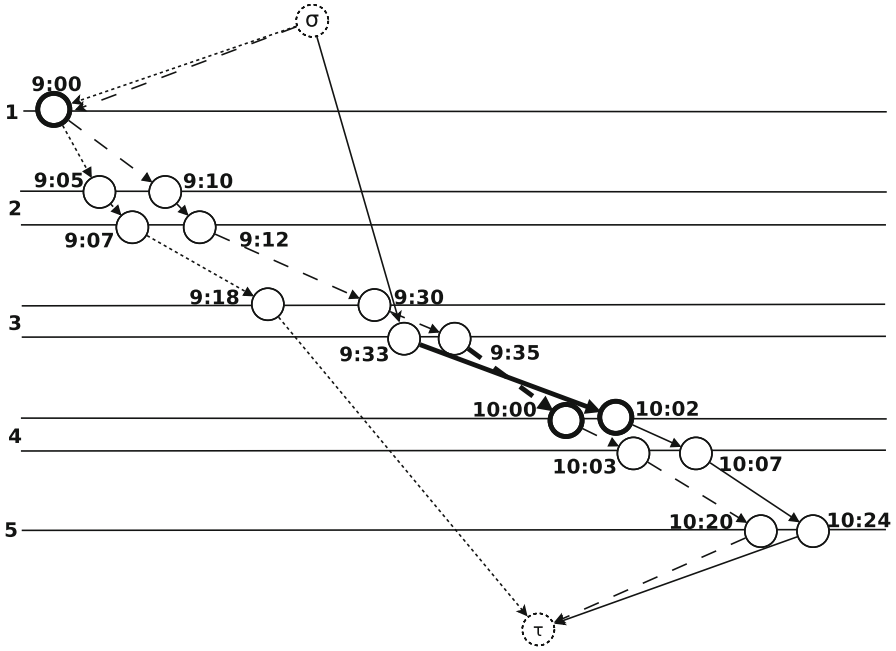

Fig. 5 Ideal paths of the presented example

show a feasible solution for the described instance. Train *A* has been shifted earlier by 2 min. Train *C* receives a stretch of 6 min in station 3 and a stretch of 2 min in station 4. Train *B* runs according to its ideal timetable.

In Fig 6, we show the paths corresponding to the actual timetables of Table 4.

Table 4 Example of actual feasible timetables

Stations	Actual timetable A		Actual timetable B		Actual timetable C	
	Arr. time	Dep. time	Arr. time	Dep. time	Arr. time	Dep. time
1		8:58		9:00		
2	9:03	9:05	9:10	9:12		
3	9:16		9:30	9:41		9:33
4			10:06	10:11	10:02	10:07
5			10:28		10:24	

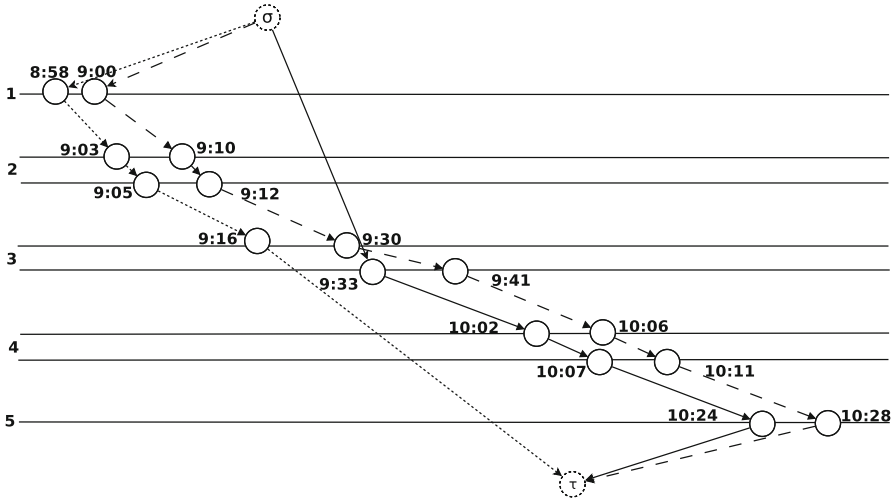

Fig. 6 Actual paths of the presented example

2.4 Integer linear programming models

In this section, we present two ILP models for TTP, both based on the graph representation of the problem presented in Sect. 2.3. The first one uses binary variables for each path of the graph, while the second one uses binary variables for each arc of the graph.

2.4.1 Path model

We next present an ILP formulation for TTP using path variables. It is a simplified version of the one proposed in Cacchiani et al. (2008) and is a *set packing* model. For each $j \in T$, let $\mathcal{P}^j$ be the collection of possible paths for train j, each associated with a path from σ to τ in G containing only arcs in A^j (paths are seen as arc subsets) and having positive profit. Furthermore, let $\mathcal{P} := \mathcal{P}^1 \cup \ldots \cup \mathcal{P}^t$ be the overall (multi-)collection of paths, and $p_P := \sum_{a \in P} p_a$ the actual profit for path $P \in \mathcal{P}$. Let

us define a graph with one node for each feasible path $P \in \mathcal{P}$ in the graph G from σ to τ, with associated profit p_P, and one edge joining each pair of nodes corresponding to paths that can be both selected in the solution. Then, the ILP model calls for determining a maximum-weight clique on this graph (see Cacchiani et al. (2010a)).

We introduce a binary variable x_P, $P \in \mathcal{P}$, for each possible path for a train, equal to 1 if, and only if, the path is chosen in the solution. Considering that for each train $j \in T$ the corresponding path can be shifted and/or stretched with respect to the *ideal path*, there is an exponentially large number of possible paths for a train. Let $\mathcal{P}^j_w \subseteq \mathcal{P}^j$ be the (possibly empty) subcollection of paths for train j that visit node $w \in V^j$, and $\mathcal{P}_w := \mathcal{P}^1_w \cup \ldots \cup \mathcal{P}^t_w$ be the subcollection of paths that visit node $w \in V$. Let r^i_j and r^i_k be the travel times of trains j and k ($j \in T$, $k \in T$, $j \neq k$) from station i to station $i+1$ ($i, i+1 \in S^j \cap S^k$), respectively. The ILP model is the following

$$\max \sum_{P \in \mathcal{P}} p_P x_P \tag{1}$$

subject to

$$\sum_{P \in \mathcal{P}^j} x_P \leq 1, \qquad \forall j \in T, \tag{2}$$

$$\sum_{w \in U^i : w \succeq u, \Delta(u,w) < a_i} \; \sum_{P \in \mathcal{P}_w} x_P \leq 1, \qquad \forall i \in S \setminus \{1\}, \forall u \in U^i, \tag{3}$$

$$\sum_{w \in W^i : w \succeq v, \Delta(v,w) < d_i} \; \sum_{P \in \mathcal{P}_w} x_P \leq 1, \qquad \forall i \in S \setminus \{s\}, \forall v \in W^i, \tag{4}$$

$$\sum_{P \in \mathcal{P}^i_{v_1}} x_P + \sum_{P \in \mathcal{P}^k_{v_2}} x_P \leq 1,$$

$$\forall i \in S \setminus \{s\}, \;\; \forall j,k \in T \;\; (\text{with } j \neq k; i, i+1 \in S^j \cap S^k), \;\; r^i_j \geq r^i_k, \\ \forall v_1, v_2 \in W^i \;\; v_1 \preceq v_2, \theta(v_2) \leq \theta(v_1) + r^i_j - r^i_k \tag{5}$$

$$x_P \geq 0, \text{binary}, \qquad \forall \mathrm{P} \in \mathcal{P}. \tag{6}$$

The objective function (1) asks for the maximization of the profits of the paths selected in the solution. According to the definition of the profits, the goal is to change as little as possible the ideal timetables. Constraints (2) require to select at most one path for each train. Note that we do not need equality constraints in (2) since we allow train cancellation. The *arrival time constraints* (3) and the *departure time constraints* (4) prevent two consecutive arrivals and departures at the same station i to be too close in time. In particular, constraints (3) are defined for each arrival node $u \in U^i$ at station $i \in S \setminus \{1\}$ (station 1 is not considered as it is the first station of the corridor and no arrival is considered at this station). Given an arrival node u, consider a time window starting at u and of length $< a_i$: only one train can arrive at station i within this time window. In Fig. 7, we show station i and a set of

arrival nodes, together with a set of paths (of any train) arriving at station i at some of these nodes. In Fig. 7, the dotted arcs correspond to arcs belonging to a set of paths (of any train) arriving at station i at some of these nodes. In order to satisfy the arrival constraints, one needs to consider a node $u \in U^i$, a time window shorter than a_i (e.g. 4) and the nodes belonging to it (bold nodes from u to w in the figure). Then, the constraints impose to select at most one path among all the ones visiting the nodes in the time window. There must be a constraint for each node $u \in U^i$ (and each station $i \in S \setminus \{1\}$). In the figure, the next window will be starting in node u' and ending in node w'.

The same holds for constraints (4): they are defined for each departure node $v \in W^i$ from station $i \in S \setminus \{s\}$ (station s is not considered as it is the last station of the corridor and no departure is considered from this station). Similarly to the arrival time constraints, the departure time constraints consider a time window starting in u and of length $< d_i$: only one train can depart from station i within this time window.

Constraints (5) are the *overtaking constraints* and impose to have no overtaking between pairs of trains along the corridor. These constraints are expressed in a clique form and require to select at most one path among conflicting ones, i.e. paths that correspond to two trains overtaking each other along a track between two consecutive stations. One needs to consider two trains, j and k, such that $r_j^i \geq r_k^i$, i.e. j is the "slow" train and k is the "fast" train, and that both visit stations i and $i+1$ (see Fig. 8). Then, one needs to consider a departure node v_1 of the slow train and a departure node v_2 of the fast train, and all the corresponding paths that visit these nodes. The two trains are overtaking each other if v_2 is after v_1 and is before the first departure $(v_1 + r_j^i - r_k^i)$ of the fast train whose corresponding arrival coincides with the arrival of the slow train (dotted node in Fig. 8). Indeed, recall that the travel times cannot be modified. The overtaking constraints (5) impose to choose at most one path among all the ones of the slow train visiting node v_1 and all the ones of the fast train visiting node v_2.

Constraints (5) can be strengthened by including clique constraints that take into account additional paths of trains j and k which are incompatible, in terms of headway times violation, with those that correspond to an overtaking of the two trains. We refer the reader to Cacchiani et al. (2008) for further details. Constraints (6) impose the variables to assume positive integer values. Other ILP models for TTP, calling for a maximum-profit collection of compatible paths in a suitable graph, are presented in Cacchiani et al. (2010a). These models look for a maximum-

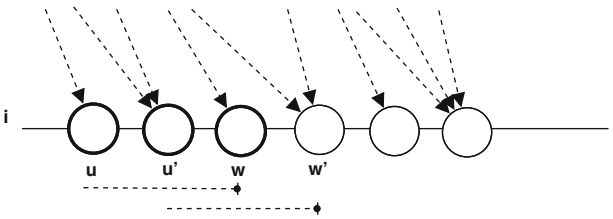

Fig. 7 Departure/arrival constraint

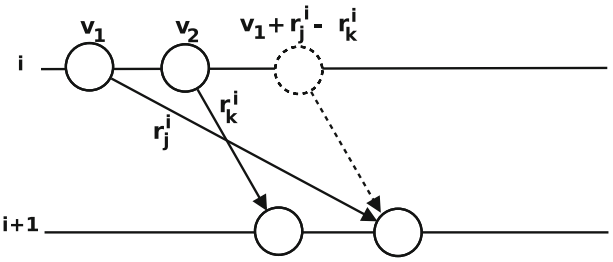

Fig. 8 Overtaking constraint

weight clique in an exponentially large compatibility graph. We refer the reader to Cacchiani et al. (2010a) for an analysis of these models.

2.4.2 Arc model

We present an ILP formulation for TTP using arc variables. The model was presented in Caprara et al. (2002) and was extended to deal with a general railway network in Cacchiani et al. (2010b). Let us introduce for each train $j \in T$ and each arc $a \in A^j$ a binary variable x_a equal to 1 if, and only if, arc a is selected in an optimal solution. Let p_a be the profit associated with each arc $a \in A$, as defined in Sect. 2.3. The arc variables are related with the path variables of model (1)–(6) as follows: $x_a = \sum_{P \in \mathcal{P}: a \in P} x_P$. For convenience, we introduce binary variables y_v equal to 1 if, and only if, node $v \in V$ is visited by any train, and binary variables z_{jv} equal to 1 if, and only if, train $j \in T$ visits node $v \in V$. Let $\delta_j^+(v)$ be the set of arcs of train j leaving node v and $\delta_j^-(v)$ be the set of arcs of train j entering node v. Let r_j^i and r_k^i be the travel times of trains j and k ($j \in T$, $k \in T$, $j \neq k$) from station i to station $i+1$ ($i, i+1 \in S^j \cap S^k$), respectively. The ILP model reads as follows:

$$\max \sum_{a \in A} p_a x_a \tag{7}$$

subject to

$$\sum_{a \in \delta_j^+(\sigma)} x_a \leq 1, \qquad \forall j \in T, \tag{8}$$

$$\sum_{a \in \delta_j^-(v)} x_a = \sum_{a \in \delta_j^+(v)} x_a, \qquad \forall j \in T, \forall v \in V^j \setminus \{\sigma, \tau\} \tag{9}$$

$$z_{jv} = \sum_{a \in \delta_j^-(v)} x_a, \qquad \forall j \in T, \forall v \in V^j \setminus \{\sigma, \tau\} \tag{10}$$

$$y_v = \sum_{j \in T: v \in V^j} z_{jv}, \qquad \forall v \in V \setminus \{\sigma, \tau\} \tag{11}$$

$$\sum_{w \in U^i: w \succeq u, \Delta(u,w) < a_i} y_w \leq 1, \qquad \forall i \in S \setminus \{1\}, \forall u \in U^i, \tag{12}$$

$$\sum_{w \in W^i : w \succeq v, \Delta(v,w) < d_i} y_w \leq 1, \qquad \forall i \in S \setminus \{s\}, \forall v \in W^i, \tag{13}$$

$$z_{jv_1} + z_{kv_2} \leq 1,$$
$$\forall i \in S \setminus \{s\}, \ \ \forall j,k \in T \ \ (\text{with} j \neq k; \forall i, i+1 \in S^j \cap S^k), \ \ r^i_j \geq r^i_k,$$
$$\forall v_1, v_2 \in W^i \ \ v_1 \preceq v_2, \theta(v_2) \leq \theta(v_1) + r^i_j - r^i_k \tag{14}$$

$$x_a \geq 0, \text{binary}, \qquad \forall \text{a} \in \text{A}, \tag{15}$$

$$y_v \geq 0, \text{binary}, \qquad \forall \text{v} \in \text{V}, \tag{16}$$

$$z_{jv} \geq 0, \text{binary}, \qquad \forall \text{j} \in \text{T}, \forall \text{v} \in \text{V}. \tag{17}$$

The meaning of objective function and constraints is analogous to the one presented for the path model. The objective function (7) asks for the maximization of the profits of the arcs selected in the solution. Constraints (8) require to select at most one timetable for each train, i.e. at most one arc among the starting arcs from the source σ for train $j \in T$. Constraints (9) impose equality on the number of selected arcs, associated with a given train, entering and leaving each arrival or departure node. As a consequence, the set of selected arcs associated with a train can either be empty, or define a path from the source σ to the sink τ. Constraints (10) and (11) are the linking constraints between different types of variables. In particular, z_{jv} assumes value 1 if there is an arc of train j entering node v. Similarly, y_v assumes value 1 if there is any train j visiting node v. The *arrival time constraints* (12) and the *departure time constraints* (13) prevent two consecutive arrivals and departures, respectively, at the same station i to be too close in time (i.e. at least the minimum headway time must be respected). The y variables are used to express easily these constraints. In order to satisfy the arrival constraints one needs to consider a node $u \in U^i$, a time window shorter than a_i and the nodes belonging to it (bold nodes from u to w in Fig. 7). Then, the constraints impose that at most one node can be visited among all the ones in the time window. There must be a constraint for each node $u \in U^i$ (and each station $i \in S \setminus \{1\}$).

Constraints (13) can be explained in a similar way. Constraints (14) are the *overtaking constraints*. The z variables are used to express these constraints. More precisely, one needs to consider two trains, j and k, such that $r^i_j \geq r^i_k$, i.e. j is the "slow" train and k is the "fast" train, and that both visit stations i and $i+1$ (see Fig. 8). The overtaking constraints (14) impose to choose at most one arc between the arc of the slow train leaving node v_1 and the arc of the fast train leaving node v_2. Note that in the path model, we have in general more than one path visiting the same node for the same train, while in the arc model, we have exactly one arc for a train visiting a node.

Constraints (14) can be strengthened with similar considerations as for the path model. Constraints (15)–(17) impose the variables to assume positive integer values.

2.5 Solution methods

In this section, we describe exact and heuristic solution methods for TTP, based on the presented ILP models. The exact method consists of a branch-and-cut-and-price algorithm, based on model (1)–(6) (see Sect. 2.5.1). This method is effective for small/medium size real-world instances (see Cacchiani et al. (2008), in which instances with up to 90 trains or 50 stations are solved to optimality). For larger size instances, heuristic approaches are more appropriate. LP-based heuristic algorithms are presented in Sect. 2.5.2. A Lagrangian-based heuristic algorithm, proposed in Caprara et al. (2002) and based on model (7)–(17), is presented in Sect. 2.5.3. The algorithm presented in Caprara et al. (2002) was extended in Cacchiani et al. (2010b) to deal with a railway network and instances with up to 500 trains and 120 stations where heuristically solved.

2.5.1 Branch-and-cut-and-price algorithm

In this section, we describe the branch-and-cut-and-price algorithm presented in Cacchiani et al. (2008) and refer the reader to that paper for a more detailed description.

We first of all describe how the LP relaxation of model (1)–(6) can be solved effectively.

2.5.1.1 LP relaxation solution The ILP model (1)–(6) has exponentially many variables. A commonly used technique for dealing with it consists of using column generation. The pricing problem calls for determining an optimal path in the (acyclic) graph considered, i.e. a timetable for each train with a positive *reduced profit*. Therefore, the pricing problem can be solved in polynomial time and makes the approach effective (see Cacchiani et al. 2008; Caprara et al. 2002). The reduced profit of a path takes into account the original profit of the path (including shift and stretch penalties) and the dual variables associated to the visited nodes (which are dealt with as penalties/prizes).

In addition, the ILP model has a very large number of constraints. These constraints can be handled by constraint separation algorithms [also known as constraint generation: see e.g. Nemhauser and Wolsey (1988)]. In particular, the separation of constraints (3), (4) and (5) can be done by enumeration, taking into account that the variables of the model can be fractional. The effect of adding new constraints with non-negative dual variables simply corresponds to changing the "penalties" of some nodes in this path computation, i.e. the addition of new constraints does not destroy the structure of the column generation problem (see Cacchiani et al. 2010a).

The general structure of the method to solve the LP relaxation is the following:

1. Initialize a reduced LP with only the t variables associated with the ideal paths for each train, constraints (2) and constraints:

$$x_P \geq 0, \qquad \forall P \in \mathcal{P}, \tag{18}$$

2. Solve the reduced LP, obtaining the primal solution x^* and the dual solution y^*;
3. Apply column generation: if variables with positive reduced profit with respect to y^* are found, add them to the reduced LP and go to Step 2.;
4. Apply separation for constraints (3) and (4): if constraints violated by x^* are found, add them to the reduced LP and go to Step 2.;
5. Apply separation for constraints (5): if constraints violated by x^* are found, add them to the reduced LP and go to Step 2.;
6. Terminate since x^*, y^* is an optimal primal-dual pair for the whole LP (1)–(5),(18).

2.5.1.2 Branching rules In the exact approach, upper bounds are computed by solving the LP relaxation by column generation and constraint separation, as described above. In order to derive integer solutions, different branching rules can be implemented. One possibility is to branch on the choice of the arcs of the multigraph. Indeed, branching on the choice of the columns is complicated because the column generation process would generate again the same columns, unless involved forbidding constraints are imposed (which could destroy the structure of the pricing problem). In order to choose the arcs so as to construct the path for one train after another, the following branching rule can be adopted:

– choose a train, say j, to branch on, based on the optimal LP solution x^*.
– select a departure node v from its first station f_j (including the possibility of not scheduling the train) or, if the departure from f_j has already been fixed, select a departure node associated with the first not yet fixed station. Also in this case, it is useful to base the choice on the optimal LP solution, trying to identify a "good" path for the train. Note that, as the travel times between consecutive stations are fixed, the specification of all departure nodes for a train uniquely identifies its path in the solution.
– branch by imposing that the path for train j visits the selected departure node $v \in W^i \cap V^j$, by setting $x_P = 0$ for all paths P that do not visit that node.

At each node of the decision tree, the LP relaxation, amended by the branching constraints, is solved by applying column generation and constraint separation as described above. The branching conditions can easily be taken into account in the column generation procedure, since it suffices to skip, in the computation of the maximum reduced profit path of each train, the set of nodes that are incompatible with those already fixed by the branching decisions.

It is generally useful to explore the decision tree according to a depth-first strategy, in order to limit the storage requirement. In addition, heuristic algorithms can be executed at the root node in order to determine a good lower bound.

2.5.2 LP-based heuristic algorithms

In this section, we present two heuristic algorithms based on the LP relaxation of model (1)–(6). We refer the reader to Cacchiani et al. (2008) for further details. Similar approaches can be obtained by considering the arc model. The construction

of heuristic solutions from the LP-relaxation can be done as follows. Some variables x_P, associated with train paths, are fixed to 1 or to 0, based on the LP-solution. The LP-relaxation is then solved again until an integer solution is found (or until the problem turns out to be infeasible). An alternative approach is to construct the path of each train step-by-step, as it is done in the enumerative algorithm of the previous section (i.e. by choosing arcs in the graph for each train). Another way to obtain a heuristic solution in short computing times is to consider the master problem containing only the variables generated in the LP relaxation solution, and to solve it with integrality constraints on the variables. Once a feasible solution has been derived, it is very useful to apply local search procedures in order to improve it. A simple local search is to consider k trains that were shifted and/or stretched and/or cancelled in the solution found and to solve heuristically or optimally the ILP model, by keeping fixed the paths for the remaining trains.

2.5.3 Lagrangian-based heuristic algorithms

This section is devoted to describe Lagrangian-based heuristic algorithms. We give an overview of these algorithms and refer the reader to Cacchiani et al. (2010b), Caprara et al. (2002, 2006) for a deeper explanation. The arc model (7)–(17) can be given on input to a general purpose ILP solver and solved to optimality. However, both the number of variables and the number of constraints of formulation (7)–(17) are very large for real-world TTP instances. As mentioned in Caprara et al. (2002), with that time state-of-the-art ILP solvers, it was "impossible to design an exact algorithm based on this model which is capable of finding an optimal solution within reasonable computing time". Therefore, in Caprara et al. (2002), a Lagrangian-based heuristic algorithm, combined with a subgradient optimization procedure, was developed. Constraints (12), (13) and (14) are relaxed in a Lagrangian way: there is a Lagrangian multiplier associated with the nodes of G (due to the y variables) as well as a Lagrangian multiplier associated with train-node pairs (due to the z variables). The resulting Lagrangian-relaxed problem calls for a set of paths for the trains, each having maximum Lagrangian profit, given by the sum of the original profits for the arcs in the path, minus the sum of the Lagrangian multipliers associated with the nodes visited by the path. Near-optimal Lagrangian multipliers can be determined through a subgradient optimization procedure. As the number of relaxed constraints is very large, a dynamic constraint-generation scheme is used (see Caprara et al. (2002) for further details). At each iteration of the subgradient optimization procedure, a heuristic solution is computed as follows. The trains are ranked according to decreasing values of the Lagrangian profit of the associated path in the relaxed solution. Then, the trains are scheduled one by one, choosing the path with maximum Lagrangian profit that is compatible with the paths already computed (those corresponding to the trains with higher ranking). Choosing the path with maximum Lagrangian profit, instead of the path with maximum actual profit, is an attempt to take care of the trains with lower ranking which still have to be scheduled. Local search procedures are applied in order to improve the solution found.

2.6 Computational experience

In this section, we present a comparison of the described solution methods with a set of real-world instances provided by Rete Ferroviaria Italiana (the main Italian railway infrastructure management company). The aim is to provide an insight into strengths and weaknesses of the different methods. The results reported in Table 5 are taken from Cacchiani et al. (2008). All times reported are expressed in CPU seconds on a PC Pentium IV 2.4 GHz with 512 Mb RAM. LP relaxation is solved using CPLEX 9.0. A time limit of 100,000 seconds was imposed to the exact branch-and-cut-and-price algorithm.

In the first three columns of Table 5, we report, respectively, the name of the corridor instance, the number of stations (which is between 16 and 102) and the number of trains (which is between 16 and 221) of the corresponding instance. The planning horizon consists of one day and times are discretized in minutes. Table 5 shows a comparison of the results obtained with the Lagrangian heuristic (Lagr. heur) described in Sect. 2.5.3 (see Caprara et al. 2006, 2002), the LP-based heuristics (LP H1 and LP H2) described in Sect. 2.5.2 (see Cacchiani et al. 2008) and the branch-and-cut-and-price method (B&C&P) described in Sect. 2.5.1 (see Cacchiani et al. 2008). We refer the reader to Cacchiani et al. (2008), Caprara et al. (2002, 2006) for further details. For each method, we report the value of the upper bound, the corresponding computing time, the value of the solution, the corresponding computing time, and the percentage gap between the best upper bound (or the value of the optimal solution) and the value of the heuristic solution found by the method. The Lagrangian upper bound is computed as described in Sect 2.5.3, and the LP relaxation upper bound, computed at the root node of the branch-and-cut-and-price algorithm, is obtained as described in Sect. 2.5.1. A "-" for the branch-and-cut-and-price indicates that the optimal solution was not found within the time limit (indicated as TL). For each instance, we show in bold the best solution found by the four methods.

The results show that 3 of the 11 instances were solved to optimality. When comparing the upper bounds obtained, we can see that the LP upper bound dominates the Lagrangian upper bound and can be computed within smaller computing times. The solutions found by LP H2 are of quality comparable to those found by Lagr. heur and require, on average, a similar computing time. In almost all cases, the solutions found by LP H1 are better than those found by Lagr. heur, but the associated computing time is much higher.

As it is evident from the results, the branch-and-cut-and-price approach is suitable for instances of small/medium size. When the dimension becomes larger, heuristic methods are more appropriate. The advantage of the Lagrangian-based approaches is that they are able to deal with large and congested instances in reasonable computing times (see e.g. Cacchiani et al. 2010b). However, the quality of the solutions and their computing times are related to finding good Lagrangian multipliers, and several iterations of subgradient optimization can be needed. The strength of the LP-based approaches is that they obtain better upper bounds and solutions. However, when dealing with instances on railway networks, if train rerouting is allowed (i.e. if trains have ideal timetables that specify only origin and

Table 5 Comparison of the results obtained with the Lagrangian heuristic of Caprara et al. (2006), the LP-based heuristics of Cacchiani et al. (2008) and the branch-and-cut-and-price of Cacchiani et al. (2008)

Name	$\|S\|$	$\|T\|$	Lagr. UB		Lagr. heur			LP UB		LP H1			LP H2			B&C&P	
			Value	Time	Value	Time	Gap %	Value	Time	Value	Time	Gap %	Value	Time	Gap %	Value	Time
PC-BO-1	17	40	4,314	65	3,629	54	11.29	4,091	14	**3,776**	5,836	7.70	3,725	137	8.95	–	TL
MU-VR	48	54	5,032	128	4,211	97	13.96	4,894	19	**4,253**	5,210	13.10	3,968	42	18.92	–	TL
BZ-VR	27	128	16,152	259	15,994	260	0.67	16,102	11	15,977	2,297	0.78	**15,997**	12	0.65	–	TL
PC-BO-2	17	93	10,953	75	10,861	78	0.19	10,914	6	**10,882**	504	0.00	10,727	3	1.42	**10,882**	683
PC-BO-3	17	60	7,235	73	7,135	73	0.36	7,200	4	7,153	644	0.11	7,138	6	0.32	**7,161**	77,562
PC-BO-4	17	221	24,243	616	21,425	523	10.33	23,894	761	**22,041**	43,200	7.76	21,753	6,029	8.96	–	TL
CH-RM	102	41	5,850	392	5,567	383	4.40	5,823	26	**5,574**	5,530	4.28	5,407	73	7.14	–	TL
BN-BO	48	68	6,909	126	**6,774**	123	1.14	6,852	6	6,716	1,811	1.98	6,649	5	2.96	–	TL
CH-MI	16	194	21,259	232	20,816	232	1.49	21,131	15	**21,022**	13,653	0.52	20,919	31	1.00	–	TL
MO-MI-1	54	16	1,727	17	**1,684**	15	0.00	1,708	2	**1,684**	36	0.00	1,634	3	2.97	**1,684**	44
MO-MI-2	54	100	11,336	415	9,318	354	16.69	11,185	288	9,453	48,051	15.49	**9,498**	1,239	15.08	–	TL

destination stations with the corresponding ideal departure/arrival times, without fixing any intermediate station to be visited), there can be many alternative paths for each train. In this case, the convergence of the column generation process can become slow. In addition, as seen in Table 5, the LP-based approaches can require much longer computing times.

3 The train platforming problem

The goal of the train platforming phase, following the train timetabling phase, is to find an assignment of trains to platforms in a railway station, and the corresponding routing inside the station. The practical relevance of the problem inspired the definition of a few different versions, depending on the infrastructure manager and train operators requirements. The TPP is relatively easy to solve for small contexts, i.e. stations with very few platforms and alternative paths to route the trains. However, it becomes a difficult optimization problem when applied to complex railway station topologies, such as those associated with the main European stations, having hundreds of trains and tens of platforms. In this section, we present the train platforming problem as proposed by the main Italian infrastructure manager *Rete Ferroviaria Italiana* (RFI) and considered in Caprara et al. (2011a) and Galli (2011). This version has many features in common with the Netherland Railways version addressed in Kroon et al. (1997), Zwaneveld (1997), Zwaneveld et al. (2001), Zwaneveld et al. (1996). We start our presentation with a brief literature review and introduce the platforming optimization problem using a small example. Next, we show a possible 0–1 quadratic programming (01-QP) model having a large number of variables, and a solution approach based on a branch-and-cut-and-price algorithm.

3.1 Literature review

As it is often the case with practical problems, every reference generally considers a different version, making it difficult to compare the proposed methods. The easiest version is the one considered by De Luca Cardillo and Mione (1999) and Billionnet (2003), who address a simplified case in which, for each train, the scheduled arrival and departure times cannot be changed and the paths used to route the trains within the station are uniquely determined by the choice of the platform. A more general version of the problem, similar to the one we present (see Caprara et al. 2011a), in which arrival and departure times and arrival and departure paths are not fixed a priori is addressed in Zwaneveld (1997), Zwaneveld et al. (1996, 2001), Kroon et al. (1997). Finally, the version addressed in Carey and Carville (2003) is an intermediate one, in that arrival and departure times can be changed, but the assignment of a train to a platform uniquely determines the paths that the train will follow on its way to and from the platform. Recently, Caprara et al. (2014) developed a *delay robust* model for *event scheduling* problems and show that TPP belongs to this class.

3.2 Problem description

In the following, we describe TPP by specifying its input, constraints and objective function, using a small example. We refer the reader to Caprara et al. (2011a) for a more detailed description.

3.2.1 Problem input: station topology and train timetable

Similarly to TTP, the input of TPP consists of a topology and a timetable and all times are discretized (e.g. in *minutes*). The topology required by TPP is a *station topology*, because TPP is solved for a specific railway station. In Fig. 9, we present an example of TPP topology instance that will be used throughout this section. Station topologies are diagrams consisting of nodes and segments that represent physical elements of the train station. The topology in Fig. 9 consists of 3 platforms, 2 directions, 5 arrival paths and 4 departure paths, which will be explained in the following.

There are several points at which a train may stop within the station to load/unload passengers and/or goods; these points are called *platforms* and can be of different type and length. The set B of platforms includes *regular platforms*, corresponding to platforms that one foresees to use, and *dummy platforms*, corresponding to virtual additional platforms that one would like not to use but that may be necessary to find a feasible solution. In the example of Fig. 9, the regular platforms are indicated with a bold line and correspond to the three segments delimited by nodes with Roman numbers I, II and III (clearly, the dummy platforms are never depicted). The concept of dummy platform was introduced in order to measure the lack of capacity in a railway station, if not all trains can be assigned to regular platforms according to the given timetable. For long-term planning, the use of dummy platforms suggests to enlarge the station, whereas for medium-term planning, it is a clear indication that not all trains can be scheduled (unless the safety margins are relaxed).

We also have a set D of *directions* for train arrivals and departures and a collection $\mathcal{R}$ of *paths* connecting directions and platforms. For each direction $d \in D$, we have a *travel time* g_d for all paths connecting d to any platform (independent of the specific path, platform and train). Moreover, for each ordered pair $(d_1, d_2) \in D \times D$ corresponding to arrival direction d_1 and departure direction d_2, the input specifies a *preference list* $L_{d_1,d_2} \subseteq B$ of preferred platforms for all trains that arrive from direction d_1 and depart to direction d_2. In our example, directions are named $D1$ and $D2$. Both directions are associated with arrival and departure tracks denoted by an "inbound" and an "outbound" arrow, respectively. Inbound nodes and tracks are represented using a "dashed" line; outbound nodes and tracks are represented using a "dotted" line. Inbound nodes $D1$ and $D2$ represent "entry" points from direction $D1$ and $D2$, respectively. Outbound nodes $D1$ and $D2$ represent "exit" points towards direction $D1$ and $D2$, respectively. We assume the travel time $g_d = 5$ min for all directions, and no preference list is given, for simplicity.

For each direction $d \in D$ and platform $b \in B$, we have a (possibly empty) set $\mathcal{R}_{d,b} \subseteq \mathcal{R}$ of paths linking direction d to platform b. Specifically, we have

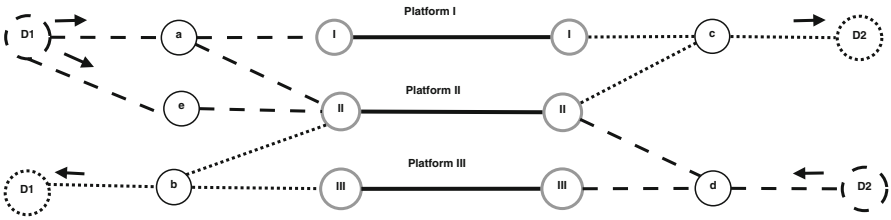

Fig. 9 Station topology example

$\mathcal{R}_{d,b} = \mathcal{R}^{\mathrm{a}}_{d,b} \cup \mathcal{R}^{\mathrm{d}}_{d,b}$, where the paths in $\mathcal{R}^{\mathrm{a}}_{d,b}$ are *arrival paths* to get from d to b and $\mathcal{R}^{\mathrm{d}}_{d,b}$ are *departure paths* to get from b to d. Note that we may have paths in $\mathcal{R}^{\mathrm{a}}_{d,b} \cap \mathcal{R}^{\mathrm{d}}_{d,b}$ called *two-way paths*. For each path $R \in \mathcal{R}$, we are given a list $\mathcal{I}_R \subseteq \mathcal{R}$ of *incompatible* paths, i.e. paths that share at least one physical element. (In particular, a path R is always incompatible with itself.) In our example, there is one arrival path from direction $D1$ to platform I (passing through node a), two arrival paths from direction $D1$ to platform II (one passing through node a, the other through node e), and two arrival paths from direction $D2$ to platforms II and III, passing through node d. The arrival paths are represented with dashed lines. Similarly, there are two departure paths towards direction $D1$ from platforms II and III passing through node b, and two departure paths towards direction $D2$ from platforms I and II passing through node c. The departure paths are represented with dotted lines. Note that the following pairs of paths are mutually incompatible:

- arrival paths from direction $D1$ to platforms I and II, passing through node a, because they share the same (inbound) track and node a
- arrival paths from direction $D2$ to platforms II and III, passing through node d, because they share the same (inbound) track and node d
- departure paths from platforms I and II towards direction $D2$, passing through node c, because they share the same (outbound) track and node c
- departure paths from platforms II and III towards direction $D1$, passing through node b, because they share the same (outbound) track and node b.

However, the arrival paths from direction $D1$ passing through node a are *compatible* with the arrival path from direction $D1$ passing through node e. This is because although trains coming from the same direction could have incompatibilities on the railway line, yet these incompatibilities are *outside* the railway station and are resolved in the timetabling phase.

The *timetable* required by TPP is the one produced after the TTP phase, restricted to the trains that go through the considered station. Each train $t \in T$ has an associated *ideal arrival time* u^{a}_t at a platform, along with a maximum *arrival shift* s^{a}_t, and an associated *ideal departure time* u^{d}_t from the platform, along with a maximum *departure shift* s^{d}_t, meaning that the train must arrive to a platform in the interval $[u^{\mathrm{a}}_t - s^{\mathrm{a}}_t, u^{\mathrm{a}}_t + s^{\mathrm{a}}_t]$ and depart in the interval $[u^{\mathrm{d}}_t - s^{\mathrm{d}}_t, u^{\mathrm{d}}_t + s^{\mathrm{d}}_t]$. Each train also specifies a *minimum stopping time*, in our example 1 min. Moreover, each $t \in T$ has

an associated arrival direction $d_t^{\mathrm{a}} \in D$, a departure direction $d_t^{\mathrm{d}} \in D$ and a set $C_t \subseteq B$ of candidate platforms where it may stop, corresponding to the platforms for which there exist at least two paths linking respectively the arrival and departure directions of t to the given platform.

In our example, let $T = \{1, 2, 3\}$ be the set of trains. The timetable provided on input is presented in Table 6. For all trains, we have $s_t^{\mathrm{a}} = s_t^{\mathrm{d}} = 1$, i.e. the maximum arrival and departure shifts are 1 min. The set of candidate platforms is indicated, for each train, in column "Platf." of Table 6.

3.2.2 Problem constraints

Assignment constraints impose that every train in the timetable must be assigned a platform, an arrival path and a departure path. If there are not enough regular platforms for all the trains, then a dummy platform must be used. Clearly, as we will see in the objective function section, this has a "very high" cost because it corresponds to a "practically infeasible" solution. Note that if a dummy platform is chosen, one does not need to select the paths, because dummy platforms are only meant to measure the lack of platform capacity in the station, so no routing information is required. In our example, train 1 can either go to platform I or go to platform II. If we select platform II, there are two possible arrival paths, namely "entry" $D1$, node a, platform II or "entry" $D1$, node e, platform II; the only possible departure path is platform II, node c, "exit" $D2$. As said in the previous section, each train also has a maximum arrival and departure shift time. Therefore, one can modify the timetable to some extent and define different *actual* arrival and departure times for each train. In our example, the actual arrival time v_1^{a} of train 1 must belong to the interval [8:59, 9:01] and the actual departure time v_1^{d} must belong to the interval [9:02, 9:04]. To summarize, assignment constraints ask for assigning each train a 5-*tuple* consisting of platform, arrival path, departure path, shift on arrival, and shift on departure. (Of course, shifts can be equal to zero). *Operational* constraints apply both to platforms and paths as follows:

- A platform cannot be assigned to more than one train at the same time.
- Incompatible paths can be assigned to different trains at the same time, only if the overlapping time interval is under a threshold π. In our example, we assume $\pi = 2$ min.

Table 6 Example of a schedule

Directions			Times		Shifts		
Train	Arr. Dir.	Dep. Dir.	Arr. Time	Dep. Time	Arr. Shift	Dep. Shift	Platf.
1	$D1$	$D2$	9:00	9:03	1	1	I, II
2	$D2$	$D1$	9:05	9:10	1	1	II, III
3	$D1$	$D1$	9:09	9:12	1	1	II

Conventionally, a train occupies a platform for the interval $[v_t^a - h, v_t^d + h]$, where h is a so-called *headway* value introduced for safety reasons. In our example, we assume $h = 3$. Moreover, the train occupies arrival path R^a for the interval $[v_t^a - g_{d_t^a}, v_t^a - 1]$ and the departure path R^d for the interval $[v_t^d + 1, v_t^d + g_{d_t^d}]$, recalling the path travel times defined above. So, if we assign train 1 to platform II with no time shifts, the actual arrival and departure times are v_1^a =9:00, v_1^d =9:03 and the occupation time intervals for arrival path, platform and departure path are given in Table 7. Note again that there are two possible arrival paths from direction $D1$ towards platform II. In our solution example we choose the path passing through node a. With regards to train 2, even if we shift forward its arrival time, it will always occupy the platform for an interval that overlaps with the (platform) interval of train 1, see Table 7. Therefore, we must assign the two trains to different platforms. If we select platform III for train 2, the only available (arrival and departure) paths are those passing through nodes d and b connecting platform III to direction $D2$ and $D1$, respectively. We assume no shift is applied to train 2 and the occupation time intervals are shown in Table 7. Note that the paths used by train 1 and 2 are compatible, since they have nothing in common. Finally, train 3 can only be assigned to platform II, for example using the arrival path from direction $D1$ through node e and the (only) departure path towards direction $D1$ through node b. Again, the occupation time intervals are shown in Table 7. Note that the platform occupation time interval of train 3 overlaps with that of train 1 and both trains are assigned to platform II. So we need to shift forward (by 1 min) the arrival time of train 3 (note that the departure time of train 3 is not changed). Also the departure paths of trains 2 and 3 are incompatible and the corresponding time intervals overlap by 3 min (i.e. 9:13, 9:14, 9:15), whereas the threshold π is 2 min. Again, we need to use shifts, in this case it is enough to shift forward (by 1 min) the departure time of train 3. The complete platforming information for our example are shown in Table 8. The arrival and departure times of train 3 are, respectively, 9:10 and 9:13.

3.2.3 Problem objective

The objective function is computed using the following coefficients, for which we also report the numerical values to give an idea of their relative importance: $\alpha_1 = 1{,}000$, $\alpha_2 = 100{,}000$, $\alpha_3 = 1$, $\alpha_4 = 100$, $\alpha_5 = 10{,}000$, $\alpha_6 = 5$.

The objective function consists of three parts: fixed *platform* costs, *train platforming* costs, and *path incompatibility* costs. Each platform cost is given by

Table 7 Occupation time intervals

Train	Arrival path	Platform	Departure path
1	8:55–8:59	8:57–9:06	9:04–9:08
2	9:00–9:04	9:02–9:13	9:11–9:15
3	9:04–9:08	9:06–9:15	9:13–9:17

Table 8 Solution example

Train	Arrival path	Platform	Departure path	Arrival shift	Departure shift
1	*D*1-*a*-II	II	II-*c*-*D*2	0	0
2	*D*2-*d*-III	III	III-*b*-*D*1	0	0
3	*D*1-*e*-II	II	II-*b*-*D*1	+1	+1

$c_b = \alpha_1$ if b is a regular platform, and $c_b = \alpha_2$ if b is a dummy platform (in other words the cost for using a dummy platform is two orders of magnitude larger than the cost for using a regular platform). Each train platforming cost c_t is given by $\alpha_3 \cdot s_t$, where s_t is the total shift (expressed in minutes) of train t (counting both the arrival and departure shifts), plus α_4 if the train uses a regular platform not in the preference list $L_{d_t^{\mathrm{a}},d_t^{\mathrm{d}}}$, plus α_5 if, instead, the train uses a dummy platform. Finally, for each pair of trains t_1 and t_2, we have a penalty $\alpha_6 \cdot o_{t_1,t_2}$, where o_{t_1,t_2} is the sum of the durations of the time intervals (in minutes) in which trains t_1 and t_2 occupy incompatible paths at the same time. Let's see the cost in our example. We use 2 regular platforms, so the platform cost is $2 * \alpha_1 = 2 * 1{,}000 = 2{,}000$. Train 1 and 2 use regular platforms, no shift, no preference list, so their cost is 0. Train 3 uses a regular platform, 1 min of shift on arrival and 1 min of shift on departure (2 min of shift in total), no preference list, so the cost is $2 * \alpha_3 = 2 * 1 = 2$. Finally, with regards to penalties for incompatible paths, the only pair of trains that uses incompatible paths at the same time is pair $(2, 3)$. As said above, we have 2 min of overlapping between the departure intervals of trains 2 and 3 that use incompatible paths, so the penalty is $2 * \alpha_6 = 2 * 5 = 10$. The total cost of our solution is $2{,}000 + 2 + 10 = 2{,}012$.

3.3 A 0-1 quadratic model

As we mentioned in the previous section, each train t must be assigned a 5-tuple consisting of a *platform* $b \in C_t$, an *arrival path* $R^{\mathrm{a}} \in \mathcal{R}^{\mathrm{a}}_{d_t^{\mathrm{a}},b}$, a *departure path* $R^{\mathrm{d}} \in \mathcal{R}^{\mathrm{d}}_{d_t^{\mathrm{d}},b}$, and the corresponding *actual arrival time* $v_t^{\mathrm{a}} \in [u_t^{\mathrm{a}} - s_t^{\mathrm{a}}, u_t^{\mathrm{a}} + s_t^{\mathrm{a}}]$ and *actual departure time* $v_t^{\mathrm{d}} \in [u_t^{\mathrm{d}} - s_t^{\mathrm{d}}, u_t^{\mathrm{d}} + s_t^{\mathrm{d}}]$. In the following, we call this 5-tuple a *pattern*. We denote with P_t the set of patterns that can be assigned to train $t \in T$. We also discussed *operational constraints* forbidding the assignment of patterns to trains if this implies occupying the same platform at the same time, or using arrival/departure paths that intersect at the same time if the overlapping time interval is larger than a threshold π. This can be easily represented by defining a *pattern incompatibility graph* with one node for each train-pattern pair (t, p), with $p \in P_t$, and an edge joining each pair $(t_1, p_1), (t_2, p_2)$ of incompatible patterns. This graph models "hard" incompatibilities that must be forbidden in a feasible solution. Note that there are also "soft" incompatibilities, generally associated with the use of incompatible arrival/departure paths if the overlapping is under the *threshold* π, that

are admitted but penalized in the objective function, as illustrated in the previous section, so they do not appear in the pattern incompatibility graph.

TPP requires the assignment of a pattern $p \in P_t$ to each train $t \in T$ so that no two incompatible patterns are assigned and the objective function defined by the following coefficients is minimized. There are a cost c_b for each platform $b \in B$ that is used in the solution, a cost $c_{t,p}$ associated with the assignment of pattern $p \in P_t$ to train $t \in T$, and a cost c_{t_1,p_1,t_2,p_2} associated with the assignment of pattern $p_1 \in P_{t_1}$ to train t_1 and of pattern $p_2 \in P_{t_2}$ to train t_2 for $(t_1, t_2) \in T^2$, in case these two patterns have a "soft" incompatibility. Note that the cost $c_{t,p}$ is the *train platforming* cost if pattern p is assigned to train t, and the cost c_{t_1,p_1,t_2,p_2} is the *path incompatibility* cost if patterns p_1 and p_2 are assigned to trains t_1, t_2, respectively.

Here, $T^2 \subseteq \{(t_1, t_2) : t_1, t_2 \in T, t_1 \neq t_2\}$ denotes the set of pairs of distinct trains whose patterns may have a "hard" or "soft" incompatibility.

We now present the 0–1 quadratic programming (0–1 QP) model proposed in Caprara et al. (2011a). Let G be the pattern incompatibility graph, and let $\mathcal{K}$ be the whole collection of cliques in G and $\mathcal{K}_b$ be the collection of cliques in G associated with sets of patterns that use platform b at the same time. That is, for each $K \in \mathcal{K}_b$ with $(t_1, p_1) \in K$ and $(t_2, p_2) \in K$, pattern p_1 for train t_1 and pattern p_2 for train t_2 both use platform b and occupy it for overlapping intervals. The straightforward 0–1 QP formulation of the problem, using a binary variable y_b for each $b \in B$, with $y_b = 1$ if and only if platform b is used, and a binary variable $x_{t,p}$ for each $t \in T$ and $p \in P_t$, with $x_{t,p} = 1$ if and only if train t is assigned pattern p, is the following:

$$\min \sum_{b \in B} c_b y_b + \sum_{t \in T} \sum_{p \in P_t} c_{t,p}\ x_{t,p} + \sum_{(t_1,t_2) \in T^2} \sum_{p_1 \in P_{t_1}} \sum_{p_2 \in P_{t_2}} c_{t_1,p_1,t_2,p_2}\ x_{t_1,p_1}\ x_{t_2,p_2} \tag{19}$$

subject to

$$\sum_{p \in P_t} x_{t,p} = 1, \qquad t \in T, \tag{20}$$

$$\sum_{(t,p) \in K} x_{t,p} \leq y_b, \qquad K \in \mathcal{K}_b, \tag{21}$$

$$\sum_{(t,p) \in K} x_{t,p} \leq 1, \qquad K \in \mathcal{K}, \tag{22}$$

$$y_b, x_{t,p} \in \{0, 1\}, \qquad b \in B,\ t \in T,\ p \in P_t. \tag{23}$$

Constraints (20) guarantee that each train is assigned a pattern, constraints (21) impose that at most one train at a time occupies a given platform b, and if this ever happens that variable y_b takes the value 1, and constraints (22) forbid the assignment of patterns that are pairwise incompatible. Note that the objective function is *quadratic*; hence, the overall model is a *discrete nonlinear* model. These models are particularly complex, because they combine the difficulties of nonlinear functions with integrality constraints on the variables.

3.4 Solution methods

In this section, we simply aim at pointing out the main features of the model presented in the previous section. We do not intend to discuss in detail the solution methods, since they are many, but let the reader know what are the available algorithmic approaches and hopefully give some useful references.

Quadratic objective function: The model is a 0–1 QP problem. There are many ways to deal with it, but basically there are two main approaches: treat it as an integer nonlinear model or linearize it and use "standard" ILP techniques to solve it. In particular, given that the integer variables are all *binary*, one can exploit many techniques developed for the 0–1 quadratic case. A good recent survey on this topic can be found in Burer and Letchford (2012). With regards to linearization, one can apply classical linearization techniques proposed for the *quadratic assignment problem* (QAP), see Burkard et al. (2009), or use Caprara et al. (2011a)'s linearization method, which guarantees a strong continuous relaxation and a fast separation algorithm.
Clique constraints: These constraints are exponentially many, so one needs to add them dynamically. In Caprara et al. (2011a), the authors present a simple exact separation technique. Of course one could also develop fast heuristic separation algorithms to be run before the exact one.
Pattern variables: The model has a large number of *pattern* variables, so it is not convenient to deal with the whole model. Yet, the number of patterns for real-world instances allows direct enumeration. So in Caprara et al. (2011a), the authors simply enumerate all pattern variables, calculate the corresponding reduced cost and dynamically add only those having a negative reduced cost.

The overall solution method presented in Caprara et al. (2011a) is a *branch-and-cut-and-price* method because both variables and constraints are added dynamically through separation and pricing procedures, respectively. Note that in Caprara et al. (2011a), branching is aimed at quickly finding a "good" feasible solution. This makes it essentially a canonical *diving* heuristic algorithm that, rather than terminating at the end of the "dive", continues as a canonical depth-first branch-and-bound method until optimality is proved (or the time limit is reached).

3.5 Computational experience

In this section, we present some computational results on a set of real-world instances provided by Rete Ferroviaria Italiana. We consider three Italian railway stations: Palermo Centrale (PA C.LE), Genova Porta Principe (GE P.PR.) and Bari Centrale (BA C.LE). In Table 9, we compare the solutions obtained by a (computationally very fast) heuristic method used by Rete Ferroviaria Italiana with the best integer solutions produced by the branch-and-cut-and-price (B&C&P) algorithm described in Sect. 3.4 with a time limit of 1 day (24 hours), recalling that TPP is a planning problem to be solved each time a new timetable is released.

We tested all the possible integer values of the dynamic threshold π. These values range from 0, in which case the occupation of incompatible paths at the same time is

Table 9 Results of the branch-and-cut-and-price method for the instances PA C.LE, GE P.PR, BA C.LE

Instance	π	Curr	LP		First		Best	
			Value	Time	Value	Time	Value	Time
PA C.LE.	0	749,012	334,038	27	349,037	48	339,039	56
PA C.LE.	1	410,139	10,159	56	120,155	96	120,155	96
PA C.LE.	2	380,182	10,159	53	10,176	86	10,176	86
PA C.LE.	3	11,431	10,159	54	10,172	106	10,172	106
GE P.PR.	0	745,000	306,020	67	306,020*	115	306,020*	115
GE P.PR.	1	705,005	147,069	192	147,087	490	147,080	825
GE P.PR.	2	458,065	8,116	274	8,121	499	8,116*	4,617
GE P.PR.	3	336,340	8,116	572	8,121	1,057	8,116*	13,647
GE P.PR.	4	8,692	8,116	434	8,126	866	8,116*	1,040
BA C.LE.	0	1,576,300	653,264	122	808,255	350	808,255	350
BA C.LE.	1	1,398,330	373,486	123	438,685	642	438,685	642
BA C.LE.	2	1,197,485	128,896	199	148,867	542	148,867	542
BA C.LE.	3	838,235	8,885	216	8,924	656	8,924	656
BA C.LE.	4	11,290	8,877	198	8,912	880	8,911	52,943

forbidden and the quadratic part in the objective function vanishes, to $g_d^{\max}$ (maximum travelling time for direction d), in which case two patterns are incompatible if and only if they occupy the same platform at the same time, whereas simultaneous occupation of incompatible paths does not affect the problem feasibility but only the quadratic part in the objective function.

In Table 9, we report the value of π, the solution value found by the heuristic method currently used by Rete Ferroviaria Italiana (*curr*), and the rounded-up value and the associated computing time of the LP relaxation at the root problem (*LP*). Moreover, we report the value and computing time of the first feasible solution found by the B&C&P algorithm (*first*), and of the best feasible solution found by the B&C&P algorithm within the time limit (*best*). A "*" means that the solution found is optimal.

The table shows that in all cases, the B&C&P algorithm was able to improve significantly over the current heuristic solution. In most cases, it found the best solution, or a solution of value very close to the best one, after a fairly small running time (some minutes). In 4 out of 14 cases, the best solution found is provably optimal (and the first solution found is very close to the optimal one), in other 5 cases the percentage gap between the first solution value and the LP lower bound is $<1\%$.

4 Conclusion

In this tutorial, we present two important railway planning problems: train timetabling and platforming. These problems are strictly connected, so we treat

them together. The timetabling and platforming problems are solved in closed sequence by the railway infrastructure manager, who is responsible for both. To our knowledge, this tutorial is the first joint treatment of (non-periodic) timetabling and platforming. We believe that the importance of a joint treatment lies in showing the reader that these problems are not independent one another. These are two strictly connected phases of the complex railway planning process and work on the same resources (e.g. railway lines, station capacity etc.) On one hand, we concentrate on the *non-periodic* version, because the periodic case is based on a different paradigm called *periodic event scheduling problem* and should be treated separately. On the other hand, even though there exist many variants of non-periodic TTP and TPP, we believe the topics we present are general enough to be easily extended to other similar contexts. The purpose of this tutorial is to introduce the non-expert reader to general variants of non-periodic TTP and TPP and show some standard, but successful, solution approaches based on integer programming.

Appendix

TTP variables and parameters

In the following, we list parameters and variables of the TTP models presented in Sect. 2.4.

The input consists of:

- $S = \{1, \ldots, s\}$ is the set of stations, numbered according to the order in which they appear along the corridor for the running direction considered
- $T = \{1, \ldots, t\}$ is the set of candidate trains
- For each train $j \in T$, a first (departure) station f_j and a last (destination) station l_j ($l_j > f_j$) are given
- $S^j := \{f_j, \ldots, l_j\} \subseteq S$ is the ordered set of stations visited by train $j \in T$
- a_i is the minimum time interval for station $i \in S$ between two consecutive arrivals of trains
- d_i is the minimum time interval for station $i \in S$ between two consecutive departures of trains
- π_j is the ideal profit of train $j \in T$
- v_j is the shift of train $j \in T$
- μ_j is the stretch of train $j \in T$
- α_j is the penalty for each minute of shift for train $j \in T$
- γ_j is the penalty for each minute of stretch for train $j \in T$.

Both models are based on the *graph representation* discussed in Sect. 2.3, consisting of a (directed, acyclic) space-time multigraph $G = (V, A)$.

The node set V has the form $\{\sigma, \tau\} \cup (U^2 \cup \ldots \cup U^s) \cup (W^1 \cup \ldots \cup W^{s-1})$, where

- σ and τ are an *artificial source node* and an *artificial sink node*, respectively

- set U^i, $i \in S \setminus \{1\}$, represents the set of time instants in which some train can arrive at station i; the nodes in $U^2 \cup \ldots \cup U^s$ are called *arrival* nodes.
- set W^i, $i \in S \setminus \{s\}$, represents the set of time instants in which some train can depart from station i; the nodes in $W^1 \cup \ldots \cup W^{s-1}$ are called *departure* nodes.

The arc set A is partitioned into sets $A^1, \ldots, A^t$, one for each train $j \in T$.

- $\theta(v)$ is the time instant associated with node $v \in V$
- $\Delta(u, v)$ is the time distance between two nodes u and v, with $u, v \in V$
- V^j denotes the set of nodes associated with time instants corresponding to possible arrivals/departures of train j.

TTP path model

- $\mathcal{P}$ is the collection of all possible paths in graph $G = (V, A)$
- $\mathcal{P}^j$ is the collection of possible paths for train j in graph $G = (V, A)$
- $p_P := \sum_{a \in P} p_a$ is the profit for path $P \in \mathcal{P}$ (p_a is the profit associated with each arc $a \in A$, as defined in Sect. 2.3)
- $\mathcal{P}^j_w \subseteq \mathcal{P}^j$ is the (possibly empty) subcollection of paths for train j that visit node $w \in V^j$
- $\mathcal{P}_w := \mathcal{P}^1_w \cup \ldots \cup \mathcal{P}^t_w$ is the subcollection of paths that visit node $w \in V$
- r^i_j is the travel time of train $j \in T$ from station i to station $i + 1$ $(i, i + 1 \in S^j)$
- x_P is a binary variable for each possible path $P \in \mathcal{P}$ for a train, equal to 1 if, and only if, the path is chosen in the solution.

TTP arc model

- p_a is the profit associated with each arc $a \in A$, as defined in Sect. 2.3
- $\delta^+_j(v)$ is the set of arcs of train j leaving node v and $\delta^-_j(v)$ is the set of arcs of train j entering node v
- r^i_j is the travel time of train $j \in T$ from station i to station $i + 1$ $(i, i + 1 \in S^j)$
- x_a is a binary variable, for each train $j \in T$ and each arc $a \in A^j$, equal to 1 if, and only if, arc a is selected in an optimal solution
- y_v is a binary variable equal to 1 if, and only if, node $v \in V$ is visited by any train
- z_{jv} is a binary variable equal to 1 if, and only if, train $j \in T$ visits node $v \in V$.

.

TPP variables and parameters

In the following, we list parameters and variables of the TPP model presented in Sect. 3.3.

The input consists of:

- T is the set of trains

- B is the set of platforms
- D is the set of *directions* for train arrivals and departures
- $\mathcal{R}$ is the set of *paths* connecting directions and platforms
- g_d is the *travel time* for all paths connecting direction $d \in D$ to any platform
- $L_{d_1,d_2} \subseteq B$ is the *preference list* of preferred platforms for the ordered pair $(d_1, d_2) \in D \times D$
- $\mathcal{R}_{d,b} \subseteq \mathcal{R}$ is the subset of paths linking direction $d \in D$ to platform $b \in B$
- $\mathcal{I}_R \subseteq \mathcal{R}$ is a list of *incompatible* paths
- u_t^{a} is the *ideal arrival time* of train $t \in T$ at a platform
- u_t^{d} is the *ideal departure time* of train $t \in T$ from a platform
- s_t^{a} is the maximum *arrival shift* of train $t \in T$
- s_t^{d} is the maximum *departure shift* of train $t \in T$
- $d_t^{\mathrm{a}} \in D$ is the arrival direction of train $t \in T$
- $d_t^{\mathrm{d}} \in D$ is the departure direction of train $t \in T$
- $C_t \subseteq B$ is the set of candidate platforms where train $t \in T$ may stop.

The model is based on the concept of *pattern* and *pattern incompatibility graph* defined in Sect. 3.3.

- P_t is the set of *patterns* that can be assigned to train $t \in T$
- $c_{t,p}$ is the *train platforming* cost if pattern $p \in P_t$ is assigned to train $t \in T$
- c_{t_1,p_1,t_2,p_2} is the *path incompatibility* cost if patterns $p_1 \in P_{t_1}$ and $p_2 \in P_{t_2}$ are assigned to trains $t_1 \in T$ and $t_2 \in T$, respectively
- $\mathcal{K}$ is the whole collection of cliques in the *pattern incompatibility graph*
- $\mathcal{K}_b$ is the collection of cliques in the pattern incompatibility graph associated with the sets of patterns that use platform $b \in B$ at the same time
- y_b is a binary variable for each $b \in B$, with $y_b = 1$ if and only if platform b is used
- $x_{t,p}$ is a binary variable for each $t \in T$ and $p \in P_t$, with $x_{t,p} = 1$ if and only if train t is assigned pattern p.

References

Adenso-Díaz B, Oliva González M (1999) On-line timetable rescheduling in regional train services. Transp Res Part B 33:387–398

Barber F, Ingolotti L, Lova A, Marin A, Mesa J, Ortega F, Tormos P (2008) Integrating timetabling, network and line design. Technical report, ARRIVAL project

Billionnet A (2003) Using integer programming to solve the train platforming problem. Transp Sci 37:213–222

Borndörfer R, Grötschel M, Lukac S, Mitusch K, Schlechte T, Schultz S, Tanner A (2006) An auctioning approach to railway slot allocation. Compet Regul Netw Ind 7(2):163–197

Borndörfer R, Schlechte T (2008) Solving railway track allocation problems. In: Kalcsics J, Nickel S (eds) Operations research proceedings 2007. Springer, Berlin, pp 117–122

Brännlund U, Lindberg PO, Nöu A, Nilsson JE (1998) Railway timetabling using Lagrangian relaxation. Transp Sci 32:358–369

Burdett R, Kozan E (2010) A sequencing approach for creating new train timetables. OR Spectr 32(1):163–193

Burer S, Letchford AN (2012) Non-convex mixed-integer nonlinear programming: a survey. Surv Oper Res Manag Sci 17(2):97–106
Burkard RE, Dell'Amico M, Martello S (2009) Assignment problems. SIAM, Philadelphia. ISBN: 978-0-898716-63-4
Cacchiani V (2009) Models and algorithms for combinatorial optimization problems arising in railway applications. 4OR A Q J Oper Res 7(1):109–112
Cacchiani V, Caprara A, Fischetti M (2012) A Lagrangian heuristic for robustness, with an application to train timetabling. Transp Sci 46(1):124–133
Cacchiani V, Caprara A, Toth P (2008) A column generation approach to train timetabling on a corridor. 4OR Q J Oper Res 6(2):125–142
Cacchiani V, Caprara A, Toth P (2010) Non-cyclic train timetabling and comparability graphs. Oper Res Lett 38(3):179–184
Cacchiani V, Caprara A, Toth P (2010) Scheduling extra freight trains on railway networks. Transp Res Part B 44B(2):215–231
Cacchiani V, Caprara A, Toth P (2013) Finding cliques of maximum weight on a generalization of permutation graphs. Optim Lett 7(2):289–296
Cacchiani V, Huisman D, Kidd M, Kroon L, Toth P, Veelenturf L, Wagenaar J (2014) An overview of recovery models and algorithms for real-time railway rescheduling. Transp Res Part B 63:15–37
Cacchiani V, Toth P (2012) Nominal and robust train timetabling problems. Eur J Oper Res 219(3):727–737
Cadarso L, Marín A (2012) Integration of timetable planning and rolling stock in rapid transit networks. Ann Oper Res 199(1):113–135
Cai X, Goh CJ (1994) A fast heuristic for the train scheduling problem. Comput Oper Res 21:499–510
Caprara A (2010) Almost 20 years of combinatorial optimization for railway planning: from Lagrangian relaxation to column generation. In: Erlebach T, Lübbecke M (eds) Proceedings of the 10th workshop on algorithmic approaches for transportation modelling, optimization, and systems (ATMOS). Schloss Dagstuhl, Germany, pp 1–12
Caprara A, Fischetti M, Toth P (2002) Modeling and solving the train timetabling problem. Oper Res 50:851–861
Caprara A, Galli L, Toth P (2011) Solution of the train platforming problem. Transp Sci 45(2):246–257
Caprara A, Galli L, Stiller S, Toth P (2014) Delay robust event scheduling. Oper Res, forthcoming
Kroon L, Monaci M, Peeters M, Toth P (2007) Passenger railway optimization. In: Barnhart C, Laporte G (eds) Transportation, handbooks in operations research and management science. Elsevier, Amsterdam, pp 129–187
Caprara A, Kroon L, Toth P (2011) Optimization problems in passenger railway systems. Wiley Encycl Oper Res Manag Sci 6:3896–3905
Caprara A, Monaci M, Toth P, Guida PL (2006) A Lagrangian heuristic approach to real-world train timetabling problems. Discret Appl Math 154:738–753
Carey M, Carville S (2003) Scheduling and platforming trains at busy complex stations. Transp Res Part A 37:195–224
Carey M, Lockwood D (1995) A model, algorithms and strategy for train pathing. J Oper Res Soc 46:988–1005
Cicerone S, D'Angelo G, Di Stefano G, Frigioni D, Navarra A (2009) Recoverable robust timetabling for single delay: complexity and polynomial algorithms for special cases. J Comb Optim 18:229–257
Cordeau JF, Toth P, Vigo D (1998) A survey of optimization models for train routing and scheduling. Transp Sci 32:380–404
D'Ariano A, Corman F, Pacciarelli D, Pranzo M (2008) Reordering and local rerouting strategies to manage train traffic in real time. Transp Sci 42(4):405–419
D'Ariano A, Pacciarelli D, Pranzo M (2007) A branch and bound algorithm for scheduling trains on a railway network. Eur J Oper Res 183(2):643–657
De Luca Cardillo D, Mione N (1999) k L-list T colouring of graphs. Eur J Oper Res 106:160–164
Dollevoet T, Huisman D, Schmidt M, Schöbel A (2012) Delay management with rerouting of passengers. Transp Sci 46:74–89
Fischer F, Helmberg C, Janßen J, Krostitz B (2008) Towards solving very large scale train timetabling problems by Lagrangian relaxation. In: Fischetti M, Widmayer P (eds) 8th Workshop on algorithmic approaches for transportation modeling, optimization, and systems (ATMOS08). Schloss Dagstuhl, Germany, pp 1–12

Fischer F, Helmberg C (2013) Dynamic graph generation for the shortest path problem in time expanded networks. Math Program. doi:10.1007/s10107-012-0610-3
Fischetti M, Monaci M (2009) Light robustness. In: Ahuja RK, Moehring R, Zaroliagis C (eds) Robust and online large-scale optimization, lecture notes in computer science, vol 5868. Springer, Berlin, pp 61–84
Fischetti M, Salvagnin D, Zanette A (2009) Fast approaches to improve the robustness of a railway timetable. Transp Sci 43:321–335
Galli L (2011) Combinatorial and robust optimisation models and algorithms for railway applications. 4OR 9:215–218
Harrod SS (2012) A tutorial on fundamental model structures for railway timetable optimization. Surv Oper Res Manag Sci 17(2):85–96
Higgings A, Kozan E, Ferreira L (1997) Heuristic techniques for single line train scheduling. J Heuristics 3:43–62
Huisman D, Kroon LG, Lentink RM, Vromans MJCM (2005) Operations research in passenger railway transportation. Stat Neerl 59:467–497
Jovanovic D, Harker PT (1991) Tactical scheduling of rail operations: the SCAN I system. Transp Sci 25:46–64
Kroon LG, Dekker R, Maróti G, Vromans MJCM (2008) Stochastic improvement of cyclic railway timetables. Transp Res Part B 42(6):553–570
Kroon LG, Peeters LWP (2003) A variable trip time model for cyclic railway timetabling. Transp Sci 37:198–212
Kroon LG, Romeijn HE, Zwaneveld PJ (1997) Routing trains through railway stations: complexity issues. Eur J Oper Res 98:485–498
Liebchen C (2006) Periodic timetable optimization in public transport. Dissertation.de - verlag im Internet GmbH. ISBN: 3-86624-150-X
Liebchen C, Lübbecke M, Möhring R, Stiller S (2009) The concept of recoverable robustness, linear programming recovery, and railway applications. In: Ahuja RK, Moehring R, Zaroliagis C (eds) Robust and online large-scale optimization, lecture notes in computer science 5868. Springer, Berlin Heidelberg, pp 1–27
Liebchen C, Möhring R et al (2007) The modeling power of the periodic event scheduling problem: railway timetables—and beyond. In: Geraets F (ed) Algorithmic methods for railway optimization, lecture notes in computer science, vol 4359. Springer, Berlin
Liebchen C, Proksch M, Wagner FH (2007) Performance of algorithms for periodic timetable optimization. In: Hickman P, Wagner FH, Voss S (eds) Computer-aided systems in public transport (CASPT 2004), lecture notes in economics and mathematical systems, vol 600. Springer, Berlin
Liebchen C, Schachtebeck M, Schöbel A, Stiller S, Prigge A (2010) Computing delay resistant railway timetables. Comput Oper Res 37(5):857–868
Lindner T (2000) Train schedule optimization in public rail transport. Ph.D. Thesis, University of Technology, Braunschweig
Lindner T, Zimmermann UT (2005) Cost optimal periodic train scheduling. Math Methods Oper Res 62:281–295
Lusby RM, Larsen J, Ehrgott M, Ryan D (2011) Railway track allocation: models and methods. OR Spectr 33(4):843–883
Lusby RM, Larsen J, Ehrgott M, Ryan DM (2013) A set-packing inspired method for real-time junction train routing. Comput Oper Res 40:713–724
Mannino C, Mascis A (2009) Optimal real-time traffic control in metro stations. Oper Res 57(4):1026–1039
Mascis A, Pacciarelli D (2002) Job-shop scheduling with blocking and no-wait constraints. Eur J Oper Res 143(3):498–517
Nachtigall K (1994) A branch-and-cut approach for periodic network programming. Technical report 29, Hildesheimer Informatik-Berichte
Nemhauser GL, Wolsey LA (1988) Integer and combinatorial optimization (Vol. 18). Wiley, New York
Odijk M (1996) A constraint generation algorithm for the construction of periodic railway timetables. Transp Res Part B 30:455–464
Oliveira E, Smith BM (2000) A job-shop scheduling model for the single-track railway scheduling problem. Technical Report 2000.21, School of Computing Research Report, University of Leeds
Peeters LWP (2003) Cyclic railway timetable optimization. Ph.D Thesis, Erasmus Research Institute of Management, Erasmus University Rotterdam

Peeters LWP, Kroon LG (2001) A cycle based optimization model for the cyclic railway timetabling problem. In: Daduna J, Voss S (eds) Computer-aided transit scheduling, lecture notes in economics and mathematical systems, vol 505. Springer, Berlin, pp 275–296

Schlechte T, Borndörfer R (2010) Balancing efficiency and robustness-A Bi-criteria optimization approach to railway track allocation. In: Ehrgott M, Naujoks B, Stewart TJ, Wallenius J (eds) Multiple criteria decision making for sustainable energy and transportation systems, lecture notes in economics and mathematical systems, vol 634. Springer, Berlin, pp 105–116

Schöbel A (2009) Capacity constraints in delay management. Public Transport 1(2):135–154

Schöbel A, Kratz A (2009) A bicriteria approach for robust timetabling. In: Ahuja RK, Moehring R, Zaroliagis C (eds) Robust and online large-scale optimization, lecture notes in computer science 5868. Springer, Berlin Heidelberg, pp 119–144

Schrijver A, Steenbeek A (1994) Timetable construction for railned. Technical report, CWI, Amsterdam, (in Dutch)

Serafini P, Ukovich W (1989) A mathematical model for periodic event scheduling problems. SIAM J Discret Math 2:550–581

Szpigel B (1973) Optimal train scheduling on a single track railway. In: Ross M (ed) OR'72. North-Holland, Amsterdam, pp 343–351

Veelenturf LP, Potthoff D, Huisman D, Kroon LG (2012) Railway crew rescheduling with retiming. Transp Res Part C 20:95–110

Walker CG, Snowdon JN, Ryan DM (2005) Simultaneous disruption recovery of a train timetable and crew roster in real time. Comput Oper Res 32:2077–2094

Zwaneveld PJ (1997) Railway planning and allocation of passenger lines. Ph.D. Thesis, Rotterdam School of Management

Zwaneveld PJ, Kroon LG, van Hoesel CPM (2001) Routing trains through a railway station based on a node packing model. Eur J Oper Res 128:14–33

Zwaneveld PJ, Kroon LG, Romeijn HE, Salomon M, Dauzere-Peres S, van Hoesel CPM, Ambergen HW (1996) Routing trains through railway stations: model formulation and algorithm. Transp Sci 30:181–194