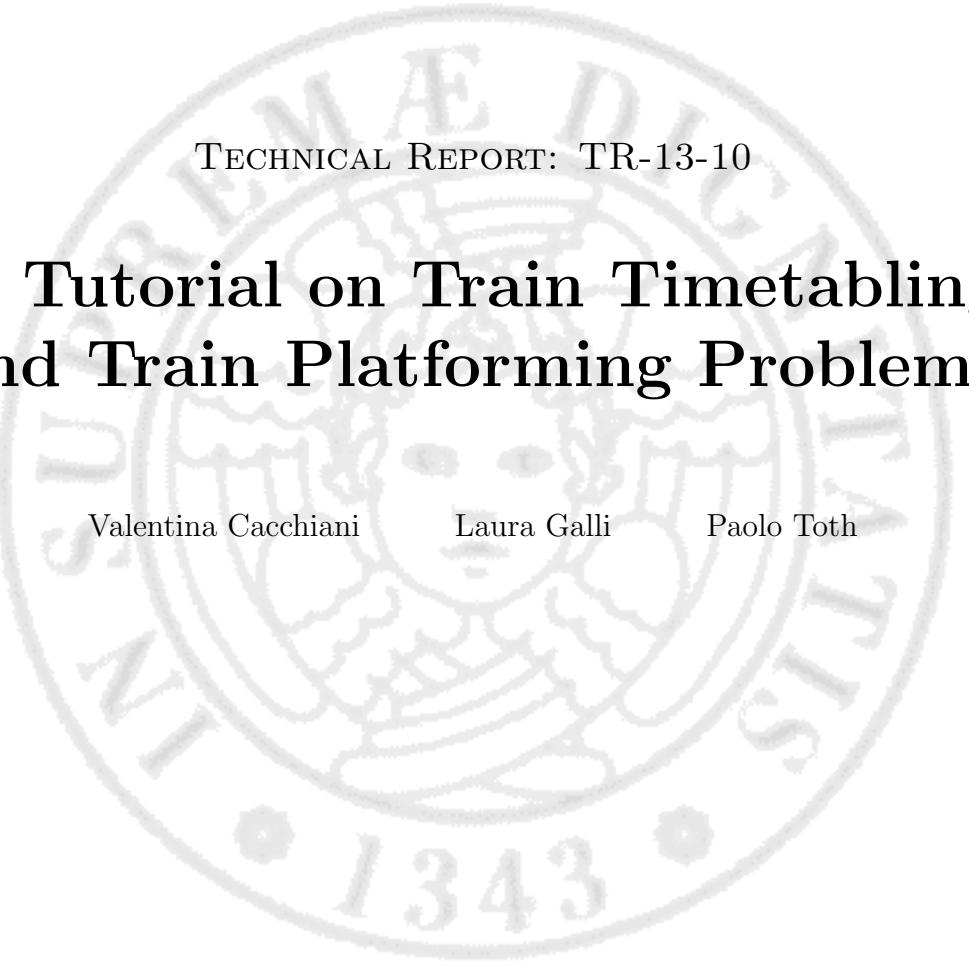
Università di Pisa
Dipartimento di Informatica



Technical Report: TR-13-10

A Tutorial on Train Timetabling and Train Platforming Problems

Valentina Cacchiani Laura Galli Paolo Toth

address: Largo B. Pontecorvo 3, 56127 Pisa, Italy. tel: +39 050 2212700 fax: +39 050 2212726

A Tutorial on Train Timetabling and Train Platforming Problems

Valentina Cacchiani* Laura Galli† Paolo Toth *

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) and the Train Platforming Problem (TPP). These problems correspond to two main phases that are usually optimized in close sequence. 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 articles for technical details.

Keywords: *Train Timetabling, Train Platforming, Integer Linear Programming, Integer Quadratic Programming, Heuristic, Branch-and-Cut-and-Price*

1 Introduction

Railway systems are packed 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 sub-problems, which are solved one after the other (i.e., the output of a problem becomes the input of the following one). These sub-problems 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. [17] for an overview on passenger railway optimization problems. In this tutorial we concentrate on two early phases, namely *Train Timetabling Problem* (TTP) and *Train Platforming Problem* (TPP). These two phases come right after the Line Planning Problem, hence they assume that the routes for the

*DEI, University of Bologna, Viale Risorgimento 2, 40136 Bologna, Italy. E-mail: {valentina.cacchiani, paolo.toth}@unibo.it

†Department of Computer Science, University of Pisa, Largo B. Pontecorvo 3, 56127 Pisa, Italy. E-mail: galli@di.unipi.it

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 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 (Sections 2.1 and 3.1) and the problem description, specifying the input, the constraints and the objective that are taken into account (Sections 2.2 and 3.2). Then, in Section 2.4, we present two Integer Linear Programming (ILP) models for the TTP (based on a graph representation of the problem presented in Section 2.3). In Section 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 (Sections 2.5 and 3.4). For each problem a set of examples is presented.

2 The Train Timetabling Problem

2.1 Literature review

TTP has received considerable attention in the literature. We refer the reader to [7], [12], [14], [17], [18], [27], and [38] for recent surveys. Two are the main variants of train timetabling: one is to consider a cyclic (or periodic) schedule of the trains that is repeated every given time period (for example every hour), and the other one is to consider a more congested network and/or a competitive market for which a non-cyclic 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 cyclic schedules. The cyclic timetabling was introduced by the seminal paper [46], 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. [30], [33], [34], [36], [37], [39], [40], [41], [42] and [45]). One of the first papers dealing with the non-cyclic TTP is presented in [47], in which a job-shop scheduling formulation for TTP on a single track railroad is proposed. Other works on non-cyclic TTP propose exact methods ([9], [28]) or heuristic methods ([4], [13], [19], [21], [26]). There are works dealing with a single one-way line ([9], [10], [13], [26], [47]) or with a general railway network ([2], [3], [11]).

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 ([29]), Light Robustness ([24], [25]), Recoverable Robustness ([22], [32]), Delay Management ([35]), Bi-objective Methods ([8], [43], [44]).

In this tutorial, we focus on non-cyclic nominal train timetabling (from now on shortly TTP). TTP calls for providing a timetable for a set of trains on a railway network, that 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 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. [11]).

2.2 Problem Description

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 every day of a given time horizon along the corridor. Let $S = \{1, \dots, s\}$ represent the set of stations, numbered according to the order in which they appear along the corridor for the running direction considered, and $T = \{1, \dots, t\}$ 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 $S^j := \{f_j, \dots, l_j\} \subseteq S$ 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 $f_j + 1, \dots, l_j - 1$. Each train is assigned by the Train Operator an *ideal timetable*, representing the most desirable timetable for the train.

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 Figure 1 and consists of a corridor with 5

stations ($S = \{1, 2, 3, 4, 5\}$).

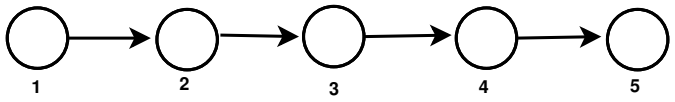


Figure 1: The considered corridor with 5 stations.

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.

	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	

Table 1: Example of three ideal timetables.

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.

The minimum time interval between arrivals or departures is called *headway time*. Let us consider $a_i = 4$ minutes for all stations $i \in S \backslash \{1\}$ and $d_i = 2$ minutes for all stations $i \in S \backslash \{s\}$ in our instance. An example of violated overtaking, arrival and departure constraints is shown in Figure 2. 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 Figure 2, 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 Figure 2, the two trains are arriving at station s_2 too close in time to each other (only 3 minutes 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 minute is in between the two departures (instead of 2 minutes).

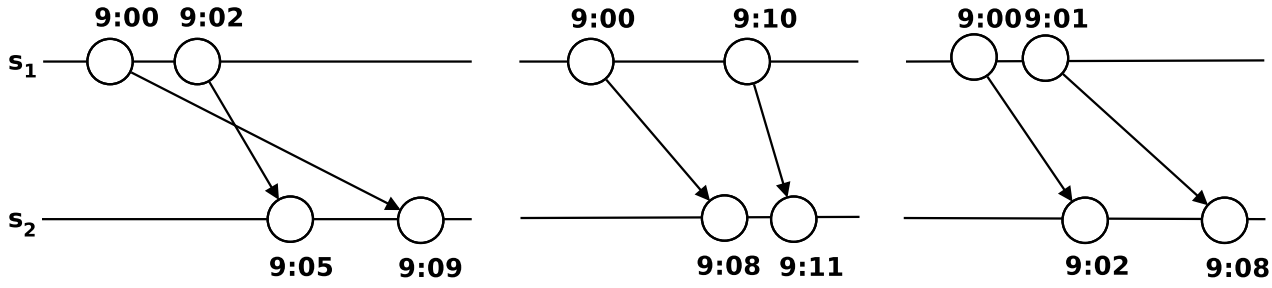


Figure 2: Examples of violated overtaking, arrival and departure constraints.

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 [19] 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 Section 2.2.3). We consider the version of the problem in which the travel time of a given train between consecutive stations cannot be modified. We refer to [15] and [19] 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 $\pi_j - \alpha_j \nu_j - \gamma_j \mu_j$, 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, therefore it is proportional to the importance given to the train;
- ν_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 (nonnegative) 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 nonnegative parameters.

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. 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 minutes and therefore its profit corresponds to the ideal profit of 200, decreased by $6 * 2$. Actual Timetable$_2$ receives a stretch of 1 minute at station 2 and its profit corresponds to $200 - 10$. Finally, Actual Timetable$_3$ is shifted of 2 minutes and stretched of 2 minutes at station 2, getting an overall profit of $200 - 6 * 2 - 10 * 2$.

	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 2: Example of ideal profits and shift/stretch penalties.

	Actual Timetable$_1$ A		Actual Timetable$_2$ A		Actual Timetable$_3$ A	
Stations	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	

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

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 $q := 1440$ (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. I.e., 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 $p_{(\sigma,v)} := \pi_j - \alpha_j \nu(v)$, with $\nu(v) := \min\{\Delta(v^*, v), \Delta(v, v^*)\}$, 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 Figure 3, 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 Figure 4, 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.

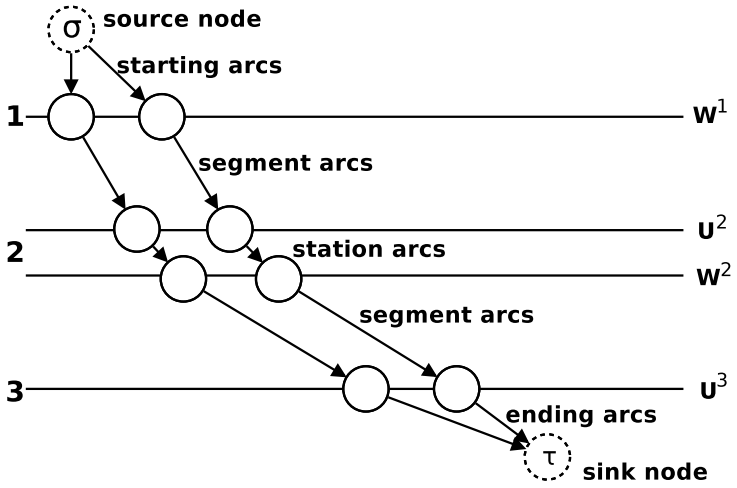


Figure 3: Multigraph representation.

In particular, train A and B depart at the same time (9:00), therefore violating the minimum headway of 2 minutes 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 minutes apart from each other instead of at least 4 minutes).

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

	Actual Timetable A		Actual Timetable B		Actual Timetable C	
Stations	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	

Table 4: Example of actual feasible timetables.

In Figure 5, we show the paths corresponding to the actual timetables of Table 4.

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 Section 2.3. The first one uses binary variables for each arc of the graph, while the second one uses binary variables for each path of the graph.

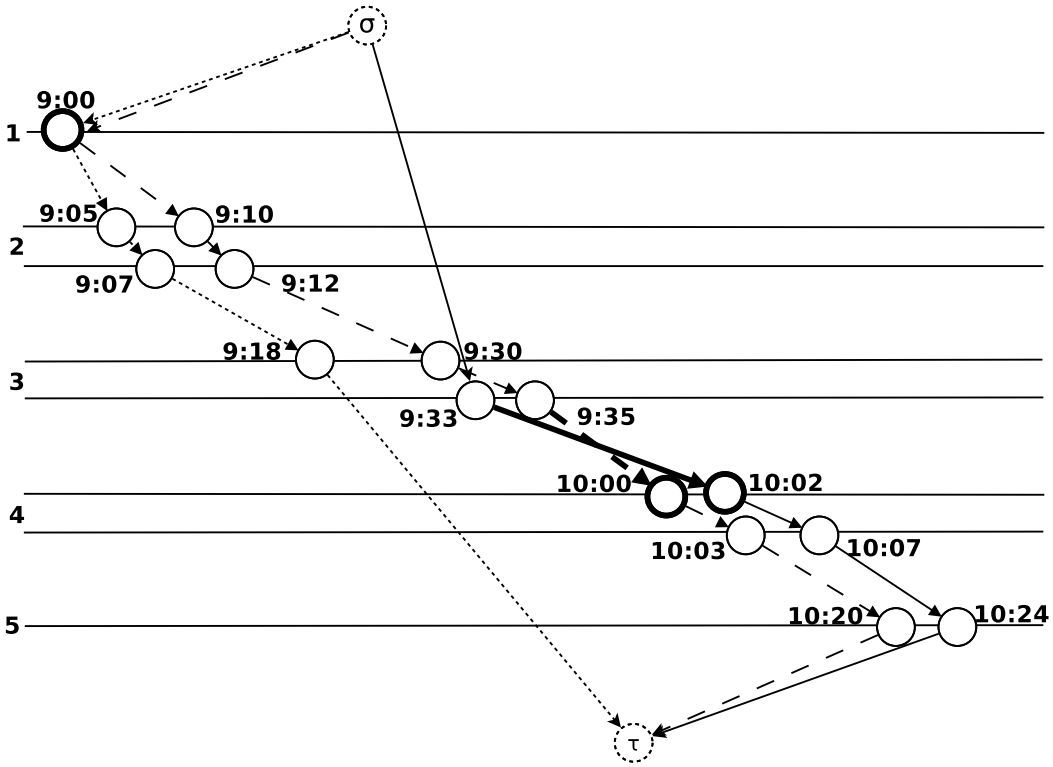


Figure 4: Ideal paths of the presented example.

2.4.1 Arc Model

We present an ILP formulation for TTP using arc variables. The model was presented in [15] and was extended to deal with a general railway network in [11]. 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 Section 2.3. 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:

$$\max \sum_{a \in A} p_a x_a \tag{1}$$

subject to

$$\sum_{a \in \delta_j^+(\sigma)} x_a \leq 1, \qquad j \in T, \tag{2}$$

$$\sum_{a \in \delta_j^-(v)} x_a = \sum_{a \in \delta_j^+(v)} x_a, \qquad j \in T, v \in V^j \setminus \{\sigma, \tau\} \tag{3}$$

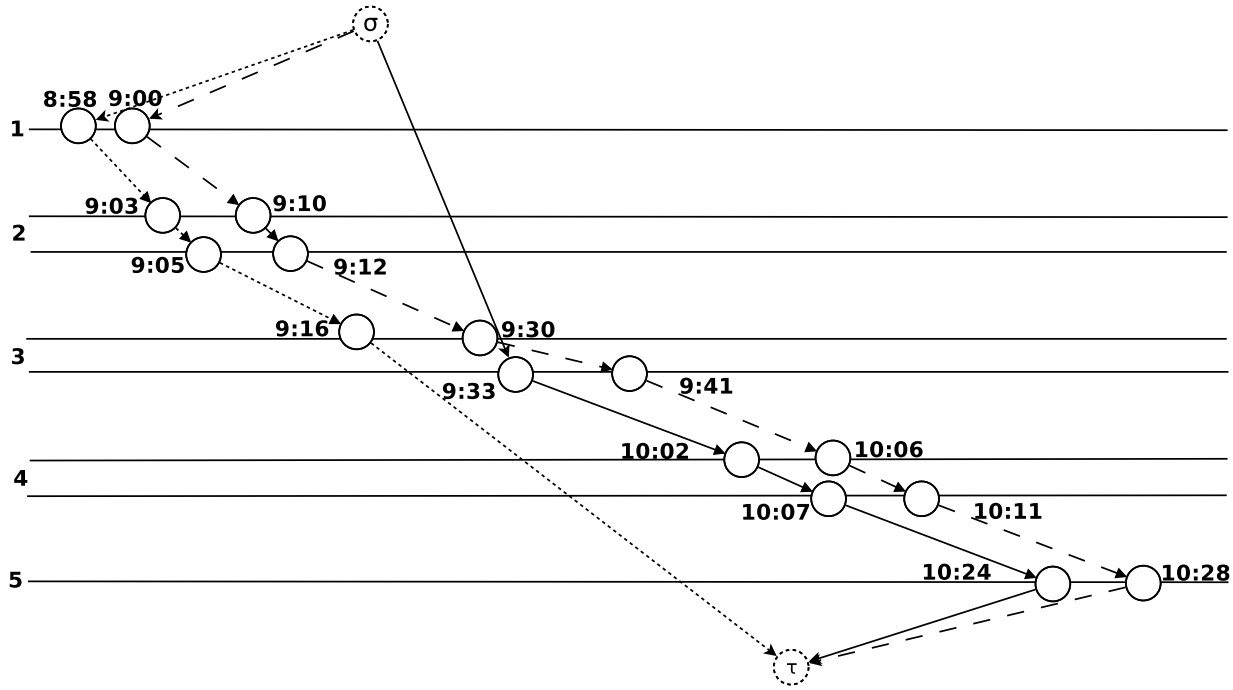


Figure 5: Actual paths of the presented example.

$$z_{jv} = \sum_{a \in \delta_j^-(v)} x_a, \qquad j \in T, v \in V^j \setminus \{\sigma, \tau\} \tag{4}$$

$$y_v = \sum_{j \in T : v \in V^j} z_{jv}, \qquad v \in V \setminus \{\sigma, \tau\} \tag{5}$$

$$\sum_{w \in U^i : w \succeq u, \Delta(u,w) < a_i} y_w \leq 1, \qquad i \in S \setminus \{1\}, u \in U^i, \tag{6}$$

$$\sum_{w \in W^i : w \succeq v, \Delta(v,w) < d_i} y_w \leq 1, \qquad i \in S \setminus \{s\}, v \in W^i, \tag{7}$$

$$z_{jv_1} + z_{kv_2} \leq 1,$$
$$i \in S \setminus \{s\}, \quad j,k \in T \quad (\text{with } j \neq k; i, i+1 \in S^j \cap S^k), \quad r_j^i \geq r_k^i,$$
$$v_1, v_2 \in W^i \quad v_1 \preceq v_2, v_2 < v_1 + r_j^i + a_{i+1} - r_k^i \tag{8}$$

$$x_a \geq 0, \qquad a \in A, \tag{9}$$

$$y_v \geq 0, \qquad v \in V, \tag{10}$$

$$z_{jv} \geq 0, \qquad j \in T, v \in V, \tag{11}$$

$$x_a \text{ integer}, \qquad a \in A, \tag{12}$$

$$y_v \text{ integer}, \qquad v \in V, \tag{13}$$

$$z_{jv} \text{ integer}, \qquad j \in T, v \in V. \tag{14}$$

The objective function (1) asks for the maximization of the profits of the arcs 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 timetable for each train, i.e. at most one arc among the starting arcs from the source σ for train $j \in T$. Note that we do not need equality constraints in (2) since we allow train cancellation. Constraints (3) 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 (4) and (5) 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* (6) and the *departure time constraints* (7) 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 particular, constraints (6) 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 Figure 6, we show station i and a set of arrival nodes, together with a set of arcs (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 arc 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'.

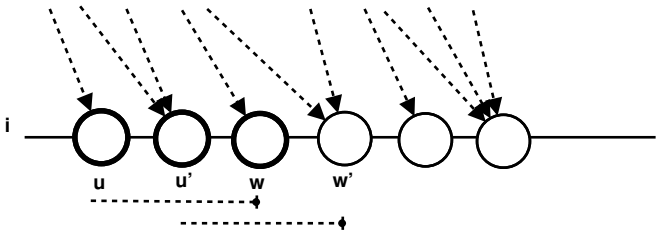


Figure 6: Departure/arrival constraint.

The same holds for constraints (7): 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 (8) are the *overtaking constraints* and impose not to have overtaking between pairs of trains along the corridor. 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 Figure 7). 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 the two corresponding arcs. The two trains are overtaking each other if v_2 is after v_1 and is before the first departure $(v_1 + r^i_j + a_{i+1} - r^i_k)$ of the fast train whose corresponding arrival is compatible with the arrival of the slow train (dotted node in Figure 7). Indeed, recall that the travel times cannot be

modified. The overtaking constraints (8) 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.

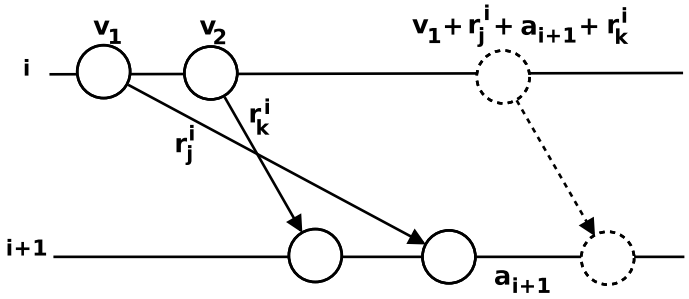


Figure 7: Overtaking constraint.

Constraints (8) can be strengthened by considering not only the arc of train j leaving node v_1 and the arc of train k leaving node v_2, but all the arcs of train j (k resp.) that leave a node in an interval v_1-v_3 (v_2-v_4 resp.) defined as follows. These intervals contain all incompatible arcs of train j and k, respectively, due to either overtaking or departure or arrival constraints. In particular, v_3 is the first departure node of the slow train compatible with the departure of the fast train, and v_4 is the first departure node of the fast train whose arrival is compatible with the arrival of the slow train (see Figure 8). The strengthened overtaking constraints can be expressed as follows:

$$\sum_{w \in W^i \cap V^j : v_1 \preceq w \prec v_3} z_{jw} + \sum_{w \in W^i \cap V^k : v_2 \preceq w \prec v_4} z_{kw} \leq 1,$$

$$i \in S \setminus \{s\}, \quad j,k \in T \quad (\text{with } j \neq k; i, i+1 \in S^j \cap S^k), \quad r^i_j \geq r^i_k,$$

$$v_1, v_2, v_3, v_4 \in W^i \quad v_1 \preceq v_2,$$

$$v_2 < v_1 + r^i_j + a_{i+1} - r^i_k, \quad \Delta(v_2, v_3) = d_i, \quad \Delta(v_1, v_4) = a_{i+1} + r^i_j - r^i_k, \tag{15}$$

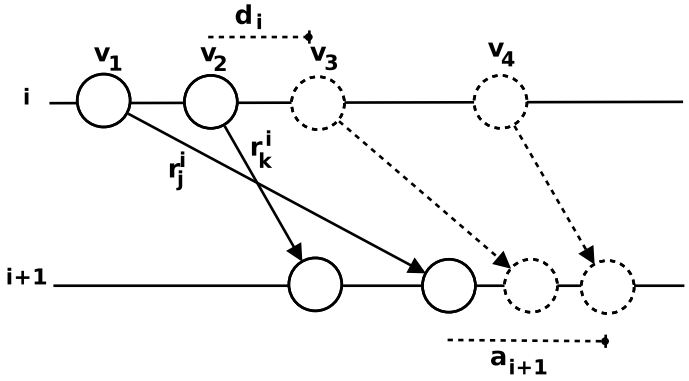


Figure 8: Strong overtaking constraint.

We wish to underline that the above ILP model would be valid also without constraints (6) and (7), as arrival/departures too close in time are forbidden also by (15). However, including arrival and departure time constraints leads to a stronger LP relaxation. In addition, these

constrains are much faster to separate in practice and this helps in reducing the computing time of the solution process. Constraints (9)-(14) impose the variables to assume positive integer values.

2.4.2 Path Model

We next present an ILP formulation for TTP using path variables. It is a simplified version of the one proposed in [9] 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 [10]).

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. The arc variables of model (1)-(14) are related with the path variables as follows: $x_a = \sum_{P \in \mathcal{P}: a \in P} x_P$. 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

$$\max \sum_{P \in \mathcal{P}} p_P x_P \tag{16}$$

subject to

$$\sum_{P \in \mathcal{P}^j} x_P \leq 1, \qquad j \in T, \tag{17}$$

$$\sum_{w \in U^i : w \succeq u, \Delta(u,w) < a_i} \; \sum_{P \in \mathcal{P}_w} x_P \leq 1, \qquad i \in S \setminus \{1\}, u \in U^i, \tag{18}$$

$$\sum_{w \in W^i : w \succeq v, \Delta(v,w) < d_i} \; \sum_{P \in \mathcal{P}_w} x_P \leq 1, \qquad i \in S \setminus \{s\}, v \in W^i, \tag{19}$$

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

$$i \in S \setminus \{s\}, \quad j,k \in T \quad (\text{with } j \neq k; i, i+1 \in S^j \cap S^k), \quad r^i_j \geq r^i_k,$$

$$v_1, v_2 \in W^i \quad v_1 \preceq v_2, v_2 < v_1 + r^i_j + a_{i+1} - r^i_k \tag{20}$$

$$x_P \geq 0, \qquad P \in \mathcal{P}, \tag{21}$$

$$x_P \text{ integer}, \qquad P \in \mathcal{P}. \tag{22}$$

The meaning of objective function and constraints is analogous to the one presented for the arc model. The objective function (16) asks for the maximization of the profits of the

paths selected in the solution. Constraints (17) require to select at most one path for each train. The *arrival time constraints* (18) and the *departure time constraints* (19) prevent two consecutive arrivals and departures at the same station i to be too close in time. In Figure 6, the dotted arcs can be interpreted as arcs belonging to a set of paths (of any train) arriving at station i at some of these nodes. The constraints impose to select at most one path among all the ones visiting the nodes in the time window. Constraints (19) can be explained in a similar way. Constraints (20) are the *overtaking constraints* and impose to have no overtaking between pairs of trains along the corridor. 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 Figure 7). The overtaking constraints (20) 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. Note that in the path model we have in general more than one path visiting the same node for the same train, with respect to exactly one arc as in the arc model. Constraints (21) and (22) impose the variables to assume positive integer values.

Constraints (20) can be strengthened with similar considerations as for the arc model, by considering not only the paths of train j passing through node v_1 and the paths of train k passing through node v_2, but all the paths of train j (k resp.) that pass through a node in an interval v_1-v_3 (v_2-v_4 resp.). The strengthened overtaking constraints can be expressed as follows:

$$\sum_{w \in W^i \cap V^j : v_1 \preceq w \prec v_3} \sum_{P \in \mathcal{P}^j_w} x_P + \sum_{w \in W^i \cap V^k : v_2 \preceq w \prec v_4} \sum_{P \in \mathcal{P}^k_w} x_P \leq 1,$$

$$i \in S \setminus \{s\}, \quad j,k \in T \quad (\text{with } j \neq k; i, i+1 \in S^j \cap S^k), \quad r^i_j \geq r^i_k,$$

$$v_1, v_2, v_3, v_4 \in W^i \quad v_1 \preceq v_2,$$

$$v_2 < v_1 + r^i_j + a_{i+1} - r^i_k, \quad \Delta(v_2, v_3) = d_i, \quad \Delta(v_1, v_4) = a_{i+1} + r^i_j - r^i_k, \tag{23}$$

As already explained for the arc model, constraints (18) and (19) are not needed but are useful to get a stronger LP relaxation. Other ILP models for TTP, calling for a maximum-profit collection of compatible paths in a suitable graph, are presented in [10]. These models look for a maximum-weight clique in a(n exponentially-large) compatibility graph. We refer the reader to [10] for an analysis of these models.

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 (16)-(22) (see Section 2.5.1). This method is effective for small/medium size real-world instances (see [9], 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 Section 2.5.2. A Lagrangian-based heuristic algorithm, proposed in [15] and based on model (1)-(14), is presented in Section 2.5.3. The algorithm presented in [15] was extended in [11] 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

We first of all describe how the LP-relaxation of model (16)-(22) can be solved effectively.

LP-relaxation solution The ILP model (16)-(22) 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. 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 separation algorithms. In particular, the separation of constraints (18), (19) and (20) ((23) resp.) can be done by enumeration, taking into account that the variables of the model can be fractional. The effect of adding new constraints with nonnegative 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 [10]).

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, and constraints (17) and (21);
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 (18) and (19): if constraints violated by x^* are found, add them to the reduced LP and go to Step 2.;
5. Apply separation for constraints (20) ((23) resp.): 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 (16)–(21).

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.

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 some heuristic algorithms based on the optimal LP-solution. We refer to model (16)-(22), but similar considerations can be applied to the arc model. The construction of heuristic solutions from the LP relaxation can be done by fixing to 1 or 0 some variables x_P associated with train paths and then solve again the LP-relaxation, until an integer solution is found (or 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). 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

The arc model (1)-(14) 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 (1)-(14) are very large for real-world TTP instances. As mentioned in [15], 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 [15], a Lagrangian based heuristic algorithm, combined with a subgradient optimization procedure, was developed. Constraints (6), (7) and (8) ((15) resp.) 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. 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.

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 Train Platforming Problem (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. [16]. This version has many features in common with the Netherland Railways version addressed in Kroon et al. [31], Zwaneveld [48], Zwaneveld et al. [49], Zwaneveld et al. [50]. 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 [23] and Billionet [1], 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. [16]), in which arrival and departure times and arrival and departure paths are not fixed a priori is addressed in Zwaneveld [48], Zwaneveld et al. [50], Zwaneveld et al. [49], Kroon et al. [31]. Finally, the version addressed in Carey and Carville [20] 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.

3.2 Problem Description

In the following, we describe TPP by specifying its input, constraints and objective function, using a small example.

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 Figure 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 Figure 9 consists of 3 platforms, 2 directions, 5 arrival paths and 4 departure paths, that will be explained in the following.

There are several points at which a train may stop within the station to download/upload 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 Figure 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 $g_d = 5$ minutes 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 $\mathcal{R}_{d,b} = \mathcal{R}^{\mathrm{a}}_{d,b} \cup \mathcal{R}^{\mathrm{d}}_{d,b}$, 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

Whereas, the arrival paths from direction $D1$ passing through node a are *compatible* with the arrival path from direction $D1$ passing through node e.

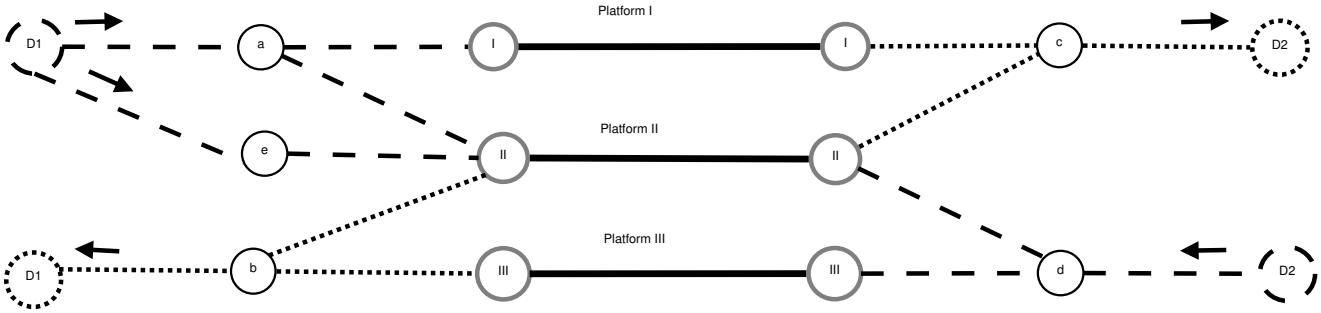


Figure 9: A station topology example.

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_t^{a} at a platform, along with a maximum *arrival shift* s_t^{a}, and an associated *ideal departure time* u_t^{d} from the platform, along with a maximum *departure shift* s_t^{d}, meaning that the train must arrive to a platform in the interval $[u_t^{\mathrm{a}} - s_t^{\mathrm{a}}, u_t^{\mathrm{a}} + s_t^{\mathrm{a}}]$ and depart in the interval $[u_t^{\mathrm{d}} - s_t^{\mathrm{d}}, u_t^{\mathrm{d}} + s_t^{\mathrm{d}}]$. Each train also specifies a *minimum stopping time*, in our example 1 minute. 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 5. For all trains, we have $s_t^{\mathrm{a}} = s_t^{\mathrm{d}} = 1$, i.e., the maximum arrival and departure shifts are one minute. The set of candidate platforms is indicated, for each train, in column 'Platf.' of Table 5.

	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

Table 5: Example of a schedule.

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 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 $\pi = 2$ minutes.

Conventionally, a train occupies a platform for the interval $[v_t^{\mathrm{a}} - h, v_t^{\mathrm{d}} + h]$, 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 $[v_t^{\mathrm{a}} - g_{d_t^{\mathrm{a}}}, v_t^{\mathrm{a}} - 1]$ and the departure path R^{d} for the interval $[v_t^{\mathrm{d}} + 1, v_t^{\mathrm{d}} + g_{d_t^{\mathrm{d}}}]$, 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 6. 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 6. 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 6. 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 6. 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 minute) 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 minutes (i.e., 9:13, 9:14, 9:15), whereas the threshold π is 2 minutes. Again, we need to use shifts, in this case it is enough to shift forward (by 1 minute) the departure time of train 3. The complete platforming information for our example are shown in Table 7. The arrival and departure times of train 3 are, respectively, 9:10 and 9:13.

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 6: Occupation time intervals.

Train	Arrival Path	Platform	Departure Path	Arrival Shift	Departure Shift
1	$D1$-a-II	II	II-c-$D2$	0	0
2	$D2$-d-III	III	III-b-$D1$	0	0
3	$D1$-e-II	II	II-b-$D1$	+1	+1

Table 7: Solution example.

3.2.3 Problem objective

The objective function is computed by using the following coefficients, for which we also report the numerical values to give an idea of their relative importance: $\alpha_1 = 1000$, $\alpha_2 = 100000$, $\alpha_3 = 1$, $\alpha_4 = 100$, $\alpha_5 = 10000$, $\alpha_6 = 5$.

The objective function consists of three parts: fixed *platform* costs, *train platforming* costs, and *path incompatibility* costs. Each platform cost is given by $c_b = \alpha_1$ 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 $2 * \alpha_1 = 2 * 1000 = 2000$. Train 1 and 2 use regular platforms, no shift, no preference list, so their cost is 0. Train 3 uses a regular platform, 1 minute of shift on arrival and 1 minute of shift on departure (2 minutes 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 minutes 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 $2000 + 2 + 10 = 2012$.

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 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 a 0-1 Quadratic Programming (0-1 QP) model. 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. I.e., 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:

$$\min \sum_{b \in B} c_b y_b + \sum_{t \in T} \sum_{p \in P_t} c_{t,p}\ x_{t,p} + \sum_{(t_1,t_2) \in T^2} \sum_{p_1 \in P_{t_1}} \sum_{p_2 \in P_{t_2}} c_{t_1,p_1,t_2,p_2}\ x_{t_1,p_1}\ x_{t_2,p_2} \tag{24}$$

subject to

$$\sum_{p \in P_t} x_{t,p} = 1, \qquad t \in T, \tag{25}$$

$$\sum_{(t,p) \in K} x_{t,p} \leq y_b, \qquad K \in \mathcal{K}_b, \tag{26}$$

$$\sum_{(t,p) \in K} x_{t,p} \leq 1, \qquad K \in \mathcal{K}, \tag{27}$$

$$y_b, x_{t,p} \in \{0, 1\}, \qquad b \in B,\ t \in T,\ p \in P_t. \tag{28}$$

Constraints (25) guarantee that each train is assigned a pattern, constraints (26) 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 (27) forbid the assignment of patterns that are pairwise incompatible. Note that the objective function is *quadratic*, hence the overall model is a *discrete non-linear* model. These models are particularly complex, because they combine the difficulties of non-linear 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. 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 presented in the previous section 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 non-linear 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 [5]. With regards to linearization, one can apply classical linearization techniques proposed for the *Quadratic Assignment Problem* (QAP), see [6], or use [16]'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 [16] 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 patters for real-world instances allows direct enumeration. So in [16] 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 [16] 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 [16] 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).

References

[1] Billionnet A (2003) Using Integer Programming to Solve the Train Platforming Problem. Transportation Science, 37:213–222

[2] Borndörfer R, Grötschel M, Lukac S, Mitusch K, Schlechte T, Schultz S, Tanner A (2006) An Auctioning Approach to Railway Slot Allocation. Competition and Regulation in Network Industries, 7(2):163–197

[3] Borndörfer R, Schlechte T (2008) Solving railway track allocation problems. Operations Research Proceedings, 117–122

[4] Brännlund U, Lindberg PO, Nöu A, Nilsson JE (1998) Allocation of Scarce Track Capacity Using Lagrangian Relaxation. Transportation Science, 32:358–369

[5] Burer S, Letchford AN (2012) Non-convex mixed-integer nonlinear programming: a survey. Surveys in Operations Research and Managment Science, 17(2):97–106

[6] Burkard RE, Dell'Amico M, Martello S (2009) Assignment Problems. SIAM

[7] Cacchiani V (2009) Models and Algorithms for Combinatorial Optimization Problems arising in Railway Applications. 4OR A Quarterly Journal of Operations Research, 7(1):109–112

[8] Cacchiani V, Caprara A, Fischetti M (2012) A Lagrangian Heuristic for Robustness, with an Application to Train Timetabling. Transportation Science, 46(1):124–133

[9] Cacchiani V, Caprara A, Toth P (2008) A Column Generation Approach to Train Timetabling on a Corridor. 4OR A Quarterly Journal of Operations Research, 6(2):125–142

[10] Cacchiani V, Caprara A, Toth P (2010) Non-Cyclic Train Timetabling and Comparability Graphs. Operations Research Letters, 38(3):179–184

[11] Cacchiani V, Caprara A, Toth P (2010) Scheduling Extra Freight Trains on Railway Networks. Transportation Research Part B, 44B(2):215–231

[12] Cacchiani V, Toth P (2012) Nominal and Robust Train Timetabling Problems. European Journal of Operational Research, 219(3):727-737

[13] Cai X, Goh CJ (1994) A Fast Heuristic for the Train Scheduling Problem. Computers and Operations Research, 21:499–510

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

[15] Caprara A, Fischetti M, Toth P (2002) Modeling and Solving the Train Timetabling Problem. Operations Research, 50:851–861

[16] Caprara A, Galli L, Toth P (2011) Solution of the Train Platforming Problem. Transportation Science 45(2):246–257

[17] Caprara A, 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, 14, Elsevier, 129–187

[18] Caprara A, Kroon L, Toth P (2011) Optimization Problems in Passenger Railway Systems. Wiley Enciclopedia, DOI: 10.1002/9780470400531.eorms0647

[19] Caprara A, Monaci M, Toth P, Guida PL (2006) A Lagrangian Heuristic Approach to Real-World Train Timetabling Problems. Discrete Applied Mathematics, 154:738–753

[20] Carey M, Carville S (2003) Scheduling and Platforming Trains at Busy Complex Stations. Transportation Research A, 37:195–224

[21] Carey M, Lockwood D (1995) A Model, Algorithms and Strategy for Train Pathing. Journal of the Operational Research Society, 46:988–1005

[22] 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. Journal of Combinatorial Optimization, 18:229–257

[23] De Luca Cardillo D, Mione N (1999) k L-List T Colouring of Graphs. European Journal of Operational Research, 106:160–164

[24] Fischetti M, Monaci M (2009) Light Robustness. in R.K. Ahuja, R. Moehring, C. Zaroliagis, Eds., Robust and Online Large-Scale Optimization, *Lecture Notes in Computer Science 5868* Springer-Verlag, Berlin Heidelberg, 61–84

[25] Fischetti M, Salvagnin D, Zanette A (2009) Fast Approaches to Improve the Robustness of a Railway Timetable. Transportation Science, 43:321–335

[26] Higgings A, Kozan E, Ferreira L (1997) Heuristic Techniques for Single Line Train Scheduling. Journal of Heuristics, 3:43–62

[27] Huisman D, Kroon LG, Lentink RM, Vromans MJCM (2005) Operations Research in Passenger Railway Transportation. Statistica Neerlandica, 59:467–497

[28] Jovanovic D, Harker PT (1991) Tactical Scheduling of Rail Operations: the SCAN I System. Transportation Science, 25:46–64

[29] Kroon LG, Dekker R, Maroti G, Retel Helmrich MR, Vromans MJCM (2008) Stochastic Improvement of Cyclic Railway Timetables. Transportation Research Part B, 42(6):553–570

[30] Kroon LG, Peeters LWP (2003) A Variable Trip Time Model for Cyclic Railway Timetabling. Transportation Science, 37:198–212

[31] Kroon LG, Romeijn HE, Zwaneveld PJ (1997) Routing Trains Through Railway Stations: Complexity Issues. European Journal of Operations Research, 98:485–498

[32] Liebchen C, Lübbecke M, Möhring R, Stiller S (2009) The Concept of Recoverable Robustness, Linear Programming Recovery, and Railway Applications. in R.K. Ahuja et al. (Eds.): *Robust and Online Large-Scale Optimization*, LNCS 5868, 1–27

[33] Liebchen C, Möhring R (2007) The Modeling Power of the Periodic Event Scheduling Problem: Railway Timetables - and Beyond. in F. Geraets et al. (eds.) Algorithmic Methods for Railway Optimization, *Lecture Notes in Computer Science* 4359 Springer

[34] Liebchen C, Proksch M, Wagner FH (2007) Performance of Algorithms for Periodic Timetable Optimization. in M. Hickman, P. Mirchandani, and S. Voss (eds.) Computer-Aided Systems in Public Transport (CASPT 2004), LNEMS 600 Springer

[35] Liebchen C, Schachtebeck M, Schöbel A, Stiller S, Prigge A (2010) Computing delay resistant railway timetables. Computers and Operations Research, 37(5):857-868

[36] Lindner T (2000) Train Schedule Optimization in Public Rail Transport. Ph.D. Thesis, University of Technology, Braunschweig

[37] Lindner T, Zimmermann UT (2005) Cost Optimal Periodic Train Scheduling. Mathematical Methods of Operations Research, 62:281–295

[38] Lusby RM, Larsenz J, Ehrgott M, Ryan D (2011) Railway Track Allocation: Models and Methods. OR Spectrum, 33(4):843–883

[39] Nachtigall K (1994) A branch-and-cut approach for periodic network programming. Technical Report 29, Hildesheimer Informatik-Berichte

[40] Odijk M (1996) A Constraint Generation Algorithm for the Construction of Periodic Railway Timetables. Transportation Research Part B, 30:455–464

[41] Peeters LWP (2003) Cyclic Railway Timetable Optimization. Ph.D Thesis, Erasmus Research Institute of Management, Erasmus University Rotterdam

[42] Peeters LWP, Kroon LG (2001) A Cycle Based Optimization Model for the Cyclic Railway Timetabling Problem. in J. Daduna, S. Voss (eds.), Computer-Aided Transit Scheduling, Lecture Notes in Economics and Mathematical Systems 505, Springer-Verlag, Berlin, 275–296

[43] Schlechte T, Borndörfer R (2010) Balancing Efficiency and Robustness–A Bi-criteria Optimization Approach to Railway Track Allocation. Multiple Criteria Decision Making for Sustainable Energy and Transportation Systems, 105–116

[44] Schöbel A, Kratz A (2009) A Bicriteria Approach for Robust Timetabling. in R.K. Ahuja et al. (Eds.): Robust and Online Large-Scale Optimization, LNCS 5868, 119144

[45] Schrijver A, Steenbeek A (1994) Timetable Construction for Railned. Technical Report, CWI, Amsterdam, (in Dutch)

[46] Serafini P, Ukovich W (1989) A Mathematical Model for Periodic Event Scheduling Problems. SIAM Journal on Discrete Mathematics, 2:550–581

[47] Szpigel B (1973) Optimal Train Scheduling on a Single Track Railway. in M. Ross (eds) *OR'72*, North-Holland, Amsterdam, 343–351

[48] Zwaneveld PJ (1997) Railway Planning and Allocation of Passenger Lines. Ph.D. Thesis, Rotterdam School of Management

[49] Zwaneveld PJ, Kroon LG, van Hoesel CPM (2001) Routing Trains through a Railway Station based on a Node Packing Model. European Journal of Operations Research, 128:14–33

[50] 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. Transportation Science, 30:181–194