Journal of Rail Transport Planning & Management 3 (2013) 1–13

Contents lists available at ScienceDirect

Journal of Rail Transport Planning & Management

journal homepage: www.elsevier.com/locate/jrtpm

A demand-based weighted train delay approach for rescheduling railway networks in real time

Jose Luis Espinosa-Aranda [*], Ricardo García-Ródenas

Escuela Superior de Informática, Universidad de Castilla-La Mancha, Paseo de la Universidad 4, Ciudad Real, Spain

ARTICLE INFO

Article history:
Received 12 April 2013
Revised 30 September 2013
Accepted 3 October 2013
Available online 30 October 2013

Keywords:
Railway traffic management
Rescheduling
Alternative graph
Heuristics
Branch-and-Cut methods

ABSTRACT

Rail systems are highly complex and their control in real time requires mathematical–computational tools. The main aim of these tools is to perform swift optimal rescheduling in response to disruptions or delays caused by events not foreseen in the original plans, so that there is no knock-on effect on other services due to these primary delays. This paper proposes a novel weighted train delay based on demand approach based on the alternative graph concept for rescheduling passenger train services. This problem is formulated as a binary integer linear programming problem which tries to maximize consumer satisfaction by minimizing total passenger delay at destinations. A heuristic method, the so-called *Avoid Most Delayed Alternative Arc* (AMDAA) algorithm, is proposed to solve the model. AMDAA is an adaptation of *Avoid Maximum Current Cmax* (AMCC) developed by Mascis and Pacciarelli (2002) to the new model. A numerical comparison is carried out with AMDAA, a Branch-and-Cut method, AMCC and the heuristic *First Come First Served* (FCFS). Numerical research carried out with data from the Renfe Cercanias Madrid rail network (Spain) shows the high computational performance in real applications of the algorithms and the suitability of this weighted train delay based on demand model versus the classical makespan minimization approach.

1. Introduction

Currently, railway systems, both of goods and of passengers, represent one of most heavily used forms of public transport in developed countries, and demand is constantly increasing. Rail networks are expanding and growing, creating very large, complex systems.

New strategies for increasing capacity are needed, including the building of new infrastructures and the improvement of existing ones. The improvement of these railway infrastructures requires heavy economic investment which may not be possible. Therefore, production-based strategies for increasing capacity by allowing more trains to be operated on the same infrastructure, or by train scheduling, are being developed. A key task for railway systems is to create an optimal timetable capable of satisfying all the requirements for assessing the proper working of the system.

The railway rescheduling problem has been defined in an on-line context without uncertainty, to try to handle an unexpected disruption that occurs in real time, Kraay and Harker (1995) and Narayanaswami and Rangaraj (2013). This situation may lead the railway system to be incapable of properly addressing the requirements of the system or satisfying the original timetable and requires network recovery to be achieved in a short time. Recovery can be carried out attempting to return train to their original timetables or generating new temporary timetables for the remainder of their journeys.

In literature about train timetabling problem, the demand is a key factor and it is exhaustively considered (Cacchiani and Toth, 2012). On the other hand, in rescheduling the existence of a passenger dissatisfaction-based timetable is assumed so its main objective is to recover the system to the pre-established timetable, not directly considering demand in this phase. In this paper is assumed that demand data is available and can be taken into account in the process of recovering the railway system.

The paper is organized as follows. Section 2 discusses previous research in rescheduling, Section 3 defines the concept of alternative graphs and the new approach proposed, Section 4 presents the heuristic algorithm used for solving the rescheduling problem, in Section 5 several computational experiments are reported to compare these solutions, and finally Section 6 concludes with a discussion of our findings and future work.

2. Past research

Many studies are currently being carried out to solve problems related to railways, we refer the reader to Caprara et al. (2002, 2007) and Cordeau et al. (1998) for surveys in railway optimization. Focussing on the train-conflict resolution problem with the

* Corresponding author. Tel.: +34 926295300.
E-mail addresses: JoseL.Espinosa@uclm.es (J.L. Espinosa-Aranda), Ricardo.Garcia@uclm.es (R. García-Ródenas).

2210-9706/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.jrtpm.2013.10.001

aim of improving the quality of the service offered, the main strategies followed are train timetabling, train dispatching, train platforming and train routing problems (Lusby et al., 2011).

This paper is focused on train-conflict resolution in real-time. Bilevel programming is a general framework used for describing this problem. The lower level problem defines an equilibrium between railway services' supply and demand. The supply model represents the infrastructure, capacities, operating rules, safety rules and design of the train services of the rail network. This model provides the real timetable when a disruption occurs following a predetermined rescheduling strategy. The demand model represents user behavior in the railway network.

The supply model is modeled as train scheduling (TS) at a microscopic level which represents how the trains move through the network. The demand model takes into account the decisions of passengers at a macroscopic level including re-routing or disconnecting strategies which consider the timetable and the possible connections between trains. This problem is known as Delay Management (DM).

The upper level problem represents rescheduling decisions within the railway network. The main points of view in the literature for modeling the objective function are the minimization of weighted train delay or the minimization of train delays. The first approach needs to include a demand model which increases its complexity when it is solved in real time. Because of this in most cases the approaches found in the literature are focused on the TS.

Table 1 shows a classification of the literature according to the approach followed.

A mathematical tool widely used for modeling the TS problem is the so-called *alternative graphs* method described by Mascis and Pacciarelli (2002). These graphs represent the feasible moves for an individual train in time and space as nodes and fixed arcs and the conflicting train paths as a pair of alternative arcs. Any solution of the graph requires that one of each alternative arc pair be selected. This approach represents the train timetabling problem as job-shop scheduling, where the railway scenario is analogous to a shop with blocking and no-waiting behavior. This formulation will lead to a minimization of the makespan of trip times in a railway context.

The main difference with the alternative graph compared to other approaches presented in the literature is the detailed but flexible representation of the network topology with regard to railway signals and operational rules. This approach can easily incorporate a number of traffic regulation rules and constraints relevant to railways, which are rarely taken into account in the literature, as observed by D'Ariano (2008).

Mascis and Pacciarelli (2002) proposes a heuristic algorithm that tries to reduce the computational costs of solving the complete alternative graph called *Avoid Maximum Current Cmax* (AMCC). This algorithm compares in each iteration two alternative arcs and avoids the arc whose selection would result in the worst solution based on the evaluation of each path.

Mazzarello and Ottaviani (2007) apply this formulation for dynamic rescheduling after delays, minimizing delays and fuel consumption. Furthermore D'Ariano et al. (2007a) apply this concept to a rescheduling problem improving AMCC algorithm with the inclusion of the concept of *static implications*. D'Ariano et al. (2008a) and Corman et al. (2010, 2011b) use alternative graphs for re-routing trains in real time. D'Ariano et al. (2008b) test the same model for dynamic timetabling for dispatching support. It is also possible to compute the optimal speed profile for each train using this model (D'Ariano et al., 2007b; Corman et al., 2009). Another approach of rescheduling presented by Corman et al. (2011a) is to modify the objective function of the model including classes of priority for the trains.

Corman et al. (2012b) deals with the coordination of multiple regional control centers. These authors demonstrate that the coordination problem can be ideally solved with a Branch and Bound procedure.

Tornquist Krasemann (2012) detects that for certain scenarios it is difficult to find good solutions within seconds using a Branch-and-Cut approach. This paper proposes a greedy algorithm which effectively delivers good solutions within the permitted time.

Currently a growing interest exists in how to represent the demand decisions, leading to the development of models that combine the TS and DM problems.

Dollevoet et al. (2009) and Schachtebeck and Schöbel (2007) propose to use an approach aimed at minimizing the sum of all passenger delays plus the sum of all missed connections. Schachtebeck and Schöbel (2007) add capacity constraints to the Delay Management formulation.

Corman et al. (2012a) describes a bi-objective TS to minimize both the delay of the trains and the number of missed connections.

Kanai et al. (2011) deals with DM and TS problems combining simulation and optimization. The simulation part consists of a train traffic simulator and a microscopic passenger flow simulator which traces the behavior of passengers one by one. The optimization approach minimizes passenger dissatisfaction.

Almodóvar and García-Ródenas (2013) proposes a model for timetable rescheduling in emergency cases, reallocating trains/buses in real time to other service lines. This model assumes that passengers use travel strategies and waiting passengers are loaded at trains/buses on a first-come-first-served basis. The infrastructure restrictions are not taken into account by the model.

Dollevoet et al. (2012) presents an integrated approach of DM and TS models. It determines which connections to maintain and

Table 1
Summary of related studies on TS and DM grouped by characteristics considered.

	TS			DM	
Reference	Re-routing	Sequencing	Speed change	Passenger delays	Connections
Kraay and Harker (1995), Tornquist and Persson (2007), Min et al. (2011), D'Ariano et al. (2008a) and Corman et al. (2010, 2011b)	X				
Mascis and Pacciarelli (2002) and D'Ariano et al. (2007a)	X	X			
Mazzarello and Ottaviani (2007)	X	X	X		
D'Ariano et al. (2008b), Corman et al. (2011a, 2012b) and Tornquist Krasemann (2012)		X			
D'Ariano et al. (2007b) and Corman et al. (2009)		X	X		
Dollevoet et al. (2009) and Schachtebeck and Schöbel (2007)		X		X	X
Corman et al. (2012a), Kanai et al. (2011) and Dollevoet et al. (2012)		X			X
Almodóvar and García-Ródenas (2013)	X			X	
Cadarso et al. (2013)		X		X	
Wang et al. (2013)	X			Passenger comfort Fuel consumption	

proposes the departure and arrival times of trains at stations using a microscopic model.

Cadarso et al. (2013) proposes a two-step approach that combines an integrated optimization model for representing the timetable and the rolling stock restrictions with a multinomial logit model which simulates passenger behavior.

Wang et al. (2013) shows a different approach to how to consider the demand, in which the objective function is a trade-off between energy consumption and comfort in transit.

The main contribution of this paper is to introduce an estimation of the demand to the alternative-graph approach by means of a linear programming formulation which tries to minimize train delays, weighted by the number of passengers that arrive at each station. This approach is focused on eliminating the flaws of existing alternative graph models when a schedule is disrupted in various services at the same time, because it will only focus on the train with the biggest delay without taking into consideration the complete system. This objective function is a particular case of the nonlinear cost function of delays at arrivals and departures presented by Kraay and Harker (1995). The model proposed in this paper tries to take into account implicitly the DM without treating demand lost at connections explicitly. Moreover the AMCC algorithm is adapted to this new model.

3. Problem formulation

3.1. Alternative graph

A railway network comprises *normal tracks* and *station tracks* which include platforms. The tracks are divided into several *block sections* which are delimited by block signals.

Train movements are controlled by a signaling system which ensures that each block section may host only one train at a time. The entrance times of all trains within block sections must be synchronized to avoid conflicts. Traversing a block section by a train is called an *operation* and the planning of the movements of trains requires determining the start time of the operations.

A modeling approach presented in the literature by Mascis and Pacciarelli (2002) is the alternative graph model. This model turns out to be a powerful tool for scheduling problems, since it was designed to deal with a wide range of applications in which the response times are a critical factor in the assessment of the goodness of the solution. It also allows the inclusion in the model of all the relevant physical characteristics and the restrictions on the railway network in order to create efficient and realistic plans.

The scheduling problem is defined by the alternative graph formed by the triplet $G = (N, F, A)$. This triplet comprises the set of *nodes* N, a set of *fixed* directed arcs F and a set of *alternative* directed arcs A.

- The set N comprises all the nodes of the graph. A node corresponds to an *operation*. Also, to include all the necessary information in the model, there are special types of node:
  - *Start node.* This is a dummy source node denoted by 0.
  - *Finish node.* A dummy sink node denoted by n, where $n-1$ is the number of real nodes present in the system.
  - *Entry node.* For each train an entry node is added which represents the entry of a train into the system. It is assumed that the start node is connected to each entry node by a fixed arc labeled with the entry time of its train.
  - *Exit node.* This node represents the train leaving the system, all the exit nodes are connected by a fixed arc to the finish node. This arc contains the expected exit time of the train from the system. This set of nodes is denoted by $N_{\mathtt{EXIT}}$.
  - *Rest of nodes.* These indicates the current position (track segment) of a train.

The decision variables are the times at which the operations start, that is, the time at which a train enters a given block section. The start time of a generic operation i is denoted by t_i, where $i = 1,\ldots,n$. The variable t_n indicates the moment at which all the trains have finished their routes.

- The set F is formed by fixed directed arcs (i, j), which represent a relationship of precedence, forcing the start time (t_i) of operation i before the start time (t_j) of operation j. The set F defines the sequence of operations carried out by the trains. If a time f_{ij} is required by the operation i and $(i, j) \in F$ thus, $t_j \geqslant t_i + f_{ij}$.
  The set F also includes the arcs which join the start node with the respective entry nodes to the system of each train and its weight is associated to the entry time of the train to the system, as well as the arcs which join the exit nodes of each train with the finish node and its weight is associated to the predicted leaving time of the system.
- *Alternative arcs* are contained in the set A. A pair of alternative arcs represents the precedence between two operations on the same resource, in this case a block section. Given a pair of arcs $((i, j),(h, k)) \in A$, arcs (i, j) and (h, k) are said to be paired and arc (i, j) is the alternative of arc (h, k). Arcs in this set model time constraints between operations of conflicting jobs and selecting one of the arcs that belongs to a pair of alternative arcs over the other enforces the sequencing and timing decision, so the selections of alternative arcs are the decision variables of the problem. Their weight corresponds with the *setup time* that ensures a minimum headway between consecutive trains.

This approach considers the idea that the logical order of tasks is related to the defined schedule, thus the predefined route of each train is what defines the order of each task i. This constraint is included in the creation of the fixed arc set chaining fixed arcs from node to node, and the MIP formulation defines a constraint which forces the start time t_i of operation i to be before the start time t_j of operation j. If a time f_{ij} (travel time) is required by the operation, then $t_j \geqslant t_i + f_{ij}$. That is, the route of each train is implicitly defined by set F.

Therefore this model is a microscopic model which defines each movement in each track segment, so it does not matter if is a double-tracked line or another network configuration. The most important thing is to define the route of each train completely. The proposed formulation and approach does not focus specifically on conflict detection and resolutions in stations but allows the formulation of scheduled stops, rolling stock connection, passenger connection and route booking constraints (D'Ariano, 2008).

Fig. 1 shows an example of a railway network with three trains. The trains will request the use of track segments 3 and 4 before actually occupying them. While traversing a sequence of track sections 3 and 4, trains will successively release each one once the tail of the train has exited, and a small setup time has elapsed. Released track sections may then be claimed by other trains. Fig. 1 (at bottom) shows its alternative graph to model these precedences, in which fixed arcs are represented in black with continuous lines and each pair of alternative arcs is in color with discontinuous lines. Alternative arcs model the possible sequencing decisions between trains 1 and 2 on track segment 3 and the precedence relationship in track segment 4 between all the trains.

The classical formulation finds a feasible schedule which minimizes the so-called *makespan*

$$t_n - t_0 = \max_{j \in N_{\mathtt{EXIT}}} \{t_j\} - t_0$$

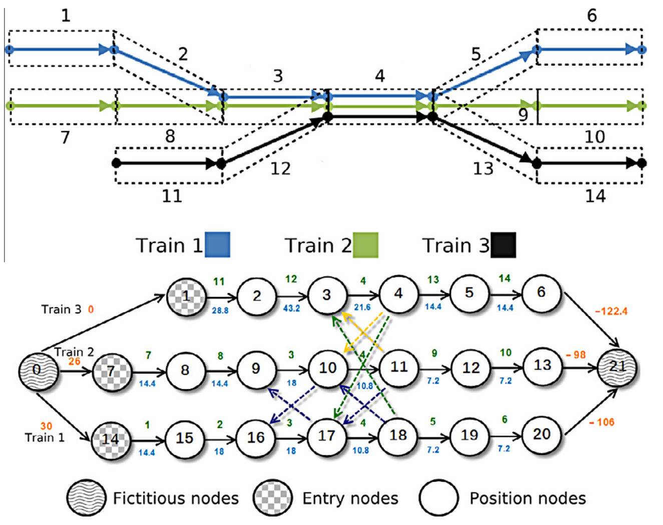


Fig. 1. An example of a physical railway network and its associated alternative graph, fixed arcs represent track segments occupied on top and travel time underneath.

where the times t_j with $j \in N_{\texttt{EXIT}}$ represent the finishing time of the last operation of the set of trains.

This problem is formulated as an integer linear problem with disjunctive constraints:

$$\begin{aligned} &\text{Minimize} \quad Z_{\texttt{MAKESPAN}} = t_n - t_0 \\ &\text{Subject to}: \\ &\qquad t_j - t_i \geqslant f_{ij}; \qquad (i,j) \in F \\ &\qquad (t_j - t_i \geqslant f_{ij}) \vee (t_k - t_h \geqslant f_{hk}); \qquad ((i,j),(h,k)) \in A \end{aligned} \tag{1}$$

3.2. A weighted train delay based on demand approach

The makespan approach described in the previous section attempts to minimize the total operation time in the system, by calculating the overall time and subtracting the starting time of the operations.

This model is used to solve job-shop problems and it is suitable when the objective is to finalize a project (minimizing the final moment t_n) but is open to question in the management of rail traffic. If a train experiences a delay and arrives at a different intermediate point later than the moment planned, the passengers who are in this station will suffer a delay which could not be mitigated even though the train regained its original timetable. Also in a case where the schedule has been disrupted in various services at same time, this approach will only focus on the train with the biggest delay which could mean the system not taking into account services with minor delays.

The approach proposed in this paper tries to address these drawbacks. The main idea is to use the concept of alternative graphs presented below, but changing the objective function, focusing on minimizing the sum of delay accumulated during arrival at the stations and including the demand, seeking to reduce the overall time of passenger delay produced in the infrastructure.

It is assumed that $N_{\texttt{STOP}}$ is the set of operations associated with scheduled stops in stations (i.e. in block sections) and g_i with $i \in N_{\texttt{STOP}}$ is an estimation of the number of passengers that leave the train in the station i represented by an origin–destination (OD) matrix disaggregated by train services. Liu et al. (2010) present a review of methods used to obtain this data. These OD-matrices are frequently used in the Delay Management problem in an off-line context and for the rescheduling problem new technologies are required to calculate these estimations in real time. These off-line estimations could be assumed to be the values of g_i. However, if the OD matrix is not available it is possible to set all the values g_i to 1 which focuses only on train delay minimization at stations.

Moreover, it is assumed that the problem is posed in a rescheduling context and that the instants in which operations are scheduled $\widehat{t}_i$ with $i \in N_{\texttt{STOP}}$ are known.

Kraay and Harker (1995) propose a nonlinear objective of delays at arrivals and departures for MISLP, each individually priced.

$$Z = \sum_{i \in N_{\texttt{STOP}}} \beta_i \left| t_i - \widehat{t}_i \right|^{\phi} + \sum_{i \in N_{\texttt{DEPARTURE-STATION}}} \alpha_i \left| t_i - \widehat{t}_i \right|^{\phi} \tag{2}$$

where $N_{\texttt{DEPARTURE-STATION}}$ is the set of operations associated with the departure time for trains from stations.

This goal appears in cargo trains and it minimizes the costs emanating from the actual deviation of planned train schedules from real delivery time. These costs are delivery penalty and inventory costs in intermediate stations. From the point of view of the passengers only the delay at destination is relevant. For this reason the proposed objective function is:

$$Z = \sum_{i \in N_{\texttt{STOP}}} g_i \max\{0, t_i - \widehat{t}_i\}^{\phi} \tag{3}$$

The parameter ϕ has been set to 1 in order to address the computationally issue. The proposed optimization model is formulated as follows:

$$\begin{aligned} &\text{Minimize} \quad Z_{\texttt{DELAY}} = \sum_{i \in N_{\texttt{STOP}}} g_i \max\{0, t_i - \widehat{t}_i\} \\ &\text{Subject to}: \\ &\qquad t_j - t_i \geqslant f_{ij}; \qquad (i,j) \in F \\ &\qquad (t_j - t_i \geqslant f_{ij}) \vee (t_k - t_h \geqslant f_{hk}); \qquad ((i,j),(h,k)) \in A \end{aligned} \tag{4}$$

The objective function of problem (4) is still related to train delay minimization, even if weighted with the number of passengers per train. It can be reformulated as the following binary integer linear programming problem:

$$\begin{aligned} &\text{Minimize} \quad Z_{\texttt{DELAY}} = \sum_{i \in N_{\texttt{STOP}}} g_i d_i \\ &\text{Subject to}: \\ &\qquad t_i - \hat{t}_i \leqslant d_i; \qquad i \in N_{\texttt{STOP}} \\ &\qquad t_j - t_i \geqslant f_{ij}; \qquad (i,j) \in F \\ &\qquad t_j - t_i \geqslant f_{ij} - M(1 - y_{ij}); \qquad ((i,j),(h,k)) \in A \\ &\qquad t_k - t_h \geqslant f_{hk} - My_{ij}; \qquad ((i,j),(h,k)) \in A \\ &\qquad d_i \geqslant 0; \qquad i \in N_{\texttt{STOP}}; \\ &\qquad y_{ij} \in \{0,1\}; \qquad ((i,j),(h,k)) \in A \end{aligned} \tag{5}$$

Theorem 1. *Let $M = \sum_{(i,j)\in F} f_{ij} + 2\sum_{((i,j),(h,k))\in A}(f_{ij} + f_{hk})$, thus the linear integer programming problem (5) is equivalent to (4).*

Proof. We begin by showing that the variable d_i for all $i \in N_{\texttt{STOP}}$ takes the value defined by $d_i = \max\{0, t_i - \widehat{t}_i\}$. From constraints $d_i \geqslant 0$ and $d_i \geqslant t_i - \widehat{t}_i$, we obtain $d_i \geqslant \max\{0, t_i - \widehat{t}_i\}$. Moreover, the coefficients g_i in the objective function are non-negative because they represent the passengers and taking into account that it is a minimization problem, the solution will be reached in minimum values of d_i. Therefore, $d_i = \max\{0, t_i - \widehat{t}_i\}$ holds.

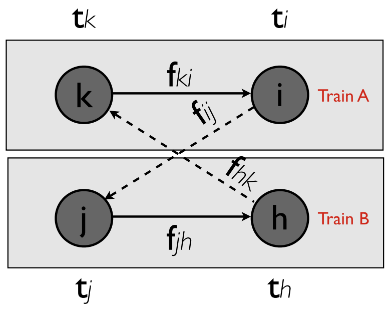


Fig. 2. Example of pair of alternative arcs.

The disjunctive constraints are modeled using the binary variables y_{ij}. Fig. 2 illustrates the proof. Let $\{t_i\}_{i\in N}$ the operational times of a feasible solution. Then, for each pair of operations h and k holds

$$|t_k - t_h| \leqslant t_n - t_0 \tag{6}$$

Assume that $y_{ij} = 1$ for an alternative arc $((i,j),(h,k)) \in A$. In this case the constraints associated with this pair of alternative arcs transform into

$$t_j - t_i \geqslant f_{ij} \tag{7}$$

$$t_k - t_h \geqslant f_{hk} - M \tag{8}$$

Now it will be proved that Eq. (8) is always satisfied, i.e. the constraint is redundant. In a feasible solution, various jobs occurs simultaneously, so the time necessary for carrying out all the operations, $t_n - t_0$, would be less than a sequential execution of the complete schedule. That is:

$$t_n - t_0 \leqslant \sum_{(i,j)\in F} f_{ij} + \sum_{((i,j),(h,k))\in A} (f_{ij} + f_{hk}) \tag{9}$$

Constraints (6) and (9) imply:

$$-\sum_{(i,j)\in F} f_{ij} - \sum_{((i,j),(h,k))\in A} (f_{ij}+f_{hk}) \leqslant t_k - t_h \leqslant \sum_{(i,j)\in F} f_{ij} + \sum_{((i,j),(h,k))\in A} (f_{ij}+f_{hk}) \tag{10}$$

Constraint (8) is guaranteed due to the choice of M which is a sufficiently large constant similar to the used in Samà et al. (2013):

$$f_{hk} - M = f_{hk} - \sum_{(i,j)\in F} f_{ij} - 2\sum_{((i,j),(h,k))\in A} (f_{ij}+f_{hk}) \leqslant -\sum_{(i,j)\in F} f_{ij} - \sum_{((i,j),(h,k))\in A} (f_{ij}+f_{hk}) \leqslant t_k - t_h \tag{11}$$

In this case train A is sequenced before B in the track segment (see Fig. 2). In the other case $y_{ij} = 0$ train B will be sequenced before train A because the other disjunctive constraint $t_k - t_h \geqslant f_{hk}$ is satisfied.
□

4. Heuristic algorithms

The formulation shown in the previous section using integer linear programming is useful for applying exact algorithms like Branch-and-Bound or Branch-and-Cut. However, the problem addressed in this paper is a particular case of *job-shop problem with limited buffer capacities* called *blocking job-shop problem* in which all buffers have capacity 0. Papadimitriou and Kanellakis (1980) showed that even the two-machine flow-shop problem with a limited buffer between the first and the second machine is strongly $\mathcal{NP}$-hard. This fact suggests solving the blocking job-shop problem with heuristics in situations in which computational time is a key factor.

The literature contains several heuristics which could be adapted and used for the railway conflict resolution problem. Two popular approaches are: *First Come First Served* (FCFS) and *Avoid Most Critical Completion Time* (AMCC) (Mascis and Pacciarelli, 2002; D'Ariano, 2008).

In FCFS a train has access to a resource in strict accordance with its arrival time, and the other trains which wish to enter a block section must wait until it is released. The traditional way in which a block section is given to a train is by strict order of arrival. This method has the advantage of simplicity, but it can lead to major disadvantages in the form of delays. For example, if a slow train reaches a block section first and its operation time is 10 min, this will cause a serious delay to a later train which arrives at the same block section a minute later and has an operation time of 5 min. This method does not take note of any delays in the rescheduling process, but an effect of this neglect is that delays can spread easily. Another problem with this principle is that it more easily leads to deadlock.

FCFS is, however, useful for detecting conflicts, since it corresponds to normal working, without the involvement of the dispatcher. This algorithm is used by the SIMEIFER tool (Espinosa-Aranda and García-Ródenas, 2012) to simulate the working of the system and to detect conflicts.

AMCC is a greedy algorithm based on global system information which tries to find a feasible solution to the scheduling problem. Firstly the finishing times for each route are studied for each pair of alternative arcs on the graph. Once the times are calculated, the higher time is identified and its use is forbidden, and the other arc is chosen.

4.1. Avoid Most Delayed Alternative Arc (AMDAA)

In this section, AMCC is adapted to the alternative graph model presented in Section 3. This heuristic is called *Avoid Most Delayed Alternative Arc* (AMDAA). AMDAA attempts to reduce the total passenger delay at the destination station.

A *selection S* is a set of arcs obtained from A by choosing at most one arc per alternative pair. The selection is called *complete* if only one arc of the pair is chosen. Given a pair of arcs $a = (u, v) \in A$, u is chosen in S if $u \in S$, and u is forbidden in S if $v \in S$. If neither of the two arcs is chosen, the pair is called *unselected pair*. The set of unselected pairs is denoted as A'. For each alternative arc w^*, we therefore associate a list of implied alternative arcs $Stat(w^*)$.

Two static implication rules are defined to add an arc to the set $Stat(w^*)$. Let B_1 and B_2 be the block sections associated with pairs $((a,b),(c,d))$ and $((i,j),(h,k))$ respectively, and let T_1 and T_2 be two trains. T_1 executes b before i, it means that nodes b and i are associated with train T_1 and are connected by a directed path of fixed arcs. T_2 executes j before a, which means that nodes j and a are associated with train T_2 and are connected by a directed path of fixed arcs. T_1 and T_2 pass through the two block sections that $((a,b),(c,d))$ and $((i,j),(h,k))$ refer to.

The two rules that define a static implication and the addition of an arc to the set $Stat(w^*)$, developed by D'Ariano (2008) are:

1. *(Trains travelling in the same direction).* If B_1 and B_2 are adjacent block sections, traversed by T_1 and T_2 in the same order, then nodes a and j, and c and k, respectively, coincide. Thus, $(a,b) \in Stat((h,k))$, $(h,k) \in Stat((a,b))$, $(c,d) \in Stat((i,j))$ and $(i,j) \in Stat((c,d))$.

2. *(Trains travelling in opposite direction).* If T_1 and T_2 pass both through B_1 and B_2 in opposite directions then, $(h,k) \in Stat((a,b))$ and $(c,d) \in Stat((i,j))$.

Given a selection S, $G(S)$ shows the graph $(N, F \cup S)$. The selection will be consistent if in graph *G(S)* there are no cycles, that is, a number of adjacent arcs where the same arc is not traversed twice, and which returns to the starting point. A solution is defined as a *consistent selection*, that is, a selection which has taken one arc from each pair of arcs present in A and there are no cycles.

The selection S is augmented in two ways. The first is by choosing a pair of unselected alternative arcs $a = (u,v)$ such that the total delays produced by this pair on passengers in the system is maximum, selecting the arc w^* that minimizes it. The second is that all the static implications on the other alternative arcs contained in a given set $Stat(w^*)$ are added to S. Each time the selection S is augmented, the start time of the operations related to the alternative arcs added to S is calculated.

More formally, let $a = (u,v) \in A'$, and let S be the current selection, then

$$Z_u = \sum_{i \in N_{\text{STOP}}} g_i \max\{0, t_i^u - \hat{t}_i\} \tag{12}$$

$$Z_v = \sum_{i \in N_{\text{STOP}}} g_i \max\{0, t_i^v - \hat{t}_i\} \tag{13}$$

$$Z_a = \max\{Z_u, Z_v\} \tag{14}$$

where t_i^u, t_i^v are the start time of the operation i on $G(S \cup \{u\})$ and $G(S \cup \{v\})$ respectively. AMDAA chooses the unselected pair as:

$$a^* = \underset{a=(u,v)\in A'}{\text{Arg maximize}}\, Z_a \tag{15}$$

Thus the new selection is

$$S = S \cup \{w^*\}$$

where

$$a^* = (u^*, v^*) \text{ and } Z_{w^*} = \text{Minimize}\{Z_{u^*}, Z_{v^*}\}. \tag{16}$$

The main difference between the AMCC and AMDAA algorithms is how they choose the arcs that will be selected. While in each iteration the AMCC algorithm enhances selection S by choosing a pair of unselected alternative arcs so as to minimize the makespan value, the AMDAA algorithm chooses alternative arcs which minimize the delay caused to passengers.

Potential deadlocks are handled in the AMCC and AMDAA algorithms the same way, defining the existence of a positive length cycle in selection S as a deadlock situation and modifying the alternative arc selected which this cycle produces. Given the nodes $i,j \in N(F)$, $l^S(i,j)$ denotes the length of the longest path from i to j on $G(S)$. Consider a selection S and two unselected alternative pairs $((a,b),(c,d))$ and $((i,j),(h,k))$. If $f_{ab} + l^S(b,i) + f_{ij} + l^S(j,a) \geqslant 0$, then arc (h,k) is implied by selection $S \bigcup \{(a,b)\}$ and arc (c,d) is implied by selection $S \bigcup \{(i,j)\}$. Consequently the selection $S \bigcup \{(a,b),(i,j)\}$ contains a positive length cycle.

The complete AMDAA pseudocode can be seen in Algorithm 1 which computes one feasible solution. This algorithm needs at first a pre-generation process in which the alternative graph G is created using the predefined routes of each train as the aspect which represents the directed arcs F between nodes N and the alternative arcs A, and calculating all the values of these arcs considering the length of track segments and the speed of the trains.

Algorithm 1. AMDAA algorithm.

```
Require: An alternative graph $G = (N,F,A)$. Let $S = \{\emptyset\}$ be the
   initial selection, and let $A' = A$, $cycle$ = false,
 1: while $A' \neq \emptyset$ $cycle$= false do
 2:   Let $a^*$ be the alternative defined in Eq. (15)
 3:   Choose the arc $w^*$ defined in Eq. (16).
 4:   Let $S' := S \bigcup \{w^*\}$,  $A' := A \backslash \{a^*\}$,
 5:   Select all arcs in set $Stat(w^*)$,
 6:   if there is a cycle in $G(S')$ an arc from $Stat(w^*)$ is forbidden
   then
 7:     Let $S' := S \bigcup \{\widetilde{w}^*\}$, where $a^* = (w^*, \widetilde{w}^*)$
 8:     Select all arcs in the set $Stat(\widetilde{w}^*)$
 9:     if there is a cycle in $G(S')$ an arc from $Stat(\widetilde{w}^*)$ is
   forbidden then
10:         $cycle$ = (The procedure failed to compute a feasible
   solution).
11:     end if
12:     $S := S'$.
13:   end if
14: end while A consistent selection $S$ infeasible solution
```

4.2. Illustrative example

This section shows an example that will help in illustrating the difference between AMCC and AMDAA algorithms.

Consider first the network shown in Fig. 3. This network has 5 stations and 2 trains which coincide in track segment 3 and their travel time in each block section is 105*s* and 100*s* respectively. Also the demand for trips $S1 \to S3$, $S1 \to S4$, $S2 \to S3$, $S2 \to S5$ is known. The demand is G_1 for the origin–destination pairs $S1 \to S3$, $S1 \to S4$, and G_2 for the pairs $S2 \to S3$, $S2 \to S5$.

The feasible set comprises only two solutions. One is that train 1 has precedence over train 2 in station 3, so alternative arc $(4,9)$ is selected. This selection produces a delay of 115 s in train 2 as the planed time to reach the station is 200 but it must wait until train 1 leaves the station at instant 315. The other solution is that train 2 has precedence over train 1 in station 3, i.e. the alternative arc $(10,3)$ is chosen. This selection produces a delay of 90 s in train 1 as the planned time to reach the station is 210 but it must wait until train 2 leaves the station at instant 300. Both heuristics explore the feasible set selecting the solution which minimizes its objective function.

The AMCC algorithm evaluates the solutions without considering demand and the results are 115 s for (4,9) selection and 90 s for (10,3) selection, so this algorithm selects alternative arc (10,3).

In the case of AMDAA algorithm the alternative arc selected will depend on the demand values. If the selected alternative arc is (4,9) then train 1 will run ahead of train 2 and it will cause a delay of 115 s in each station to train 2, as the number of passengers who leave in each station is G_2 then the passengers will suffer $115*2*G_2 = 230G_2$ seconds of delay, $115*G_2$ for each of the two destination stations. Otherwise, the selected arc would be (10,3) and train 2 will run ahead of train 1 and it will cause a delay of 90 s in each station of train 1, as the number of passengers who leave in each station is G_1 then its total delay will be $90*2*G_1 = 180G_1$ seconds, $90*G_1$ for each of the two destination stations. Mathematically,

$$Z_u = 230G_2$$

$$Z_v = 180G_1$$

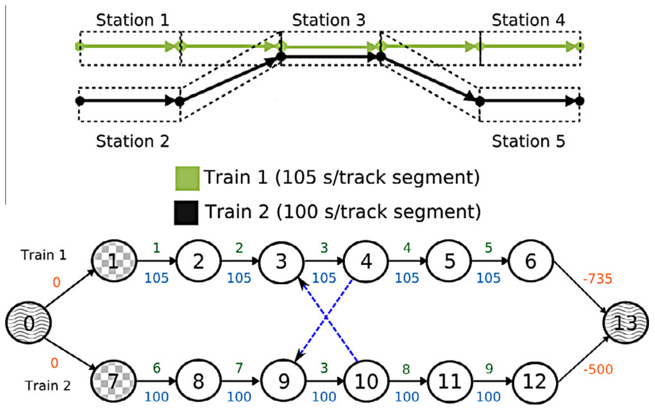


Fig. 3. Illustrative example notation.

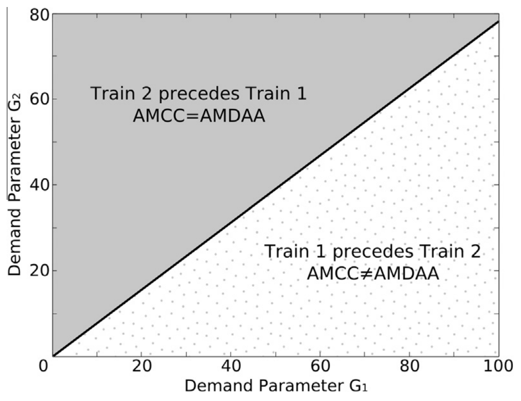


Fig. 4. AMDAA alternative arc selection depending on demand.

where $u = (4,9)$ and $v = (10,3)$. Therefore the solution selected by AMDAA between the two alternative arcs will depend on the values of the demand parameters G_1 and G_2. If $230G_2 < 180G_1$ arc (4,9) will be selected. Otherwise, arc (10,3) will be selected. Fig. 4 depicts the solution found by AMDAA depending on the parameters G_1 and G_2. The black line represent the solutions of the equation $230G_2 = 180G_1$ dividing the complete set of feasible solutions depending on the values given to G_1 and G_2 for AMDAA algorithm. The gray part presents the infinite set of solutions in which the AMDAA algorithm will obtain the same solution as the AMCC algorithm, and the white part represents the infinite set in which they differ. It is worth noting that the case when $G_1 = G_2$ represents a situation in which only the delays of all trains at stations are taken into account. This case belongs to the gray part of the picture.

5. Computational experiments

The network used for the computational experiments is the Madrid local rail network, called Renfe Cercanias Madrid. It makes up 60% of the local networks in Spain, 40% of rail traffic and is able to move more than a million passengers per day. One of the most important actions which have been undertaken recently consisted of the building of new lines to Barajas airport from the station of *Chamartin* and three stations, *Manoteras*, *Valdebebas* and *Barajas T-4* on the route.

This part of the network, routes followed by trains and conflict zones are showed in Fig. 5. This infrastructure comprises 93 block sections and is used by 475 trains in an operational period of 20 h, which means an average of 24 trains running on the infrastructure per hour, with maxima of 36 and minima of 3 trains per hour.

Because the relevant interval for a rail dispatcher for real-time purposes in regional railway systems, because of their journey time, is estimated at something less than an hour, the numerical comparison will study the following two time windows, firstly the time horizon is divided into 20 intervals of 1 h and solved individually, next the time horizon is divided into 10 intervals of 2 h and solved individually. It is a rolling time horizon approach, and in this case the connection between time interval and number of trains is related to the predefined schedule, showing the number of trains that will start their journey in each interval. The case study has been obtained from the perturbation of the original timetable. A set of random events is applied to 65% of trains causing some of the trains to make their journey at a slower speed (5–30% reduction) than the speed planned for them. The data concerning the network and the complete disturbance scenario has been uploaded to http://bit.ly/16AdgiH in PDF and to http://bit.ly/15OIi5K in XML formats.

To carry out the computational experiments the SOFGA tool[1] ("Secuenciacion de Operaciones Ferroviarias mediante Grafos Alternativos" which means "Railway Operations Sequencing with Alternative Graphs") was developed to implement the models and algorithms mentioned in previous sections. The tool has been codified using MATLAB. Fig. 6 shows a capture of the window of SOFGA tool. Area 1 represents the part the user uses for loading the XML files which contain the data of the initial schedule. Area 2 is used for selecting the intervals in which the user wants to work. Area 3 is used for applying the different algorithms to the selected interval and view the obtained solution.

The FCFS, AMCC and AMDAA algorithms have been implemented with the SOFGA tool, and run on an Intel Core i3 2.40 GHz with 4 GB of RAM, and CPLEX implemented in GAMS with a standard configuration and without a time limit for the computation for the integer linear programming models on an Intel i7 3.06 GHz with 16 GB of RAM. CPLEX uses a Branch-and-Cut algorithm for integer linear programming. In these experiments $BC_{\texttt{MAKESPAN}}$ and $BC_{\texttt{DELAY}}$ denote the Branch-and-Cut algorithm applied respectively to objective functions $Z_{\texttt{MAKESPAN}}$ and $Z_{\texttt{DELAY}}$.

The main goals of this computational study are to answer the following two questions:

1. What are the strengths and weaknesses of each algorithm for minimizing both passenger delay and makespan?
2. Why is the performance the way it is for each algorithm and how are the scenarios affecting it?

The answer to the first question is addressed in *Experiment 1* and to the second in *Experiment 2*.

5.1. Experiment 1

This experiment is divided into two parts. The first test will be aimed at train delay minimization so the values of all the demands in the system have been set to $g_i = 1$. This value means that all trains have the same priority and all stations have the same relevance. This experiment focuses on comparing the performance of the algorithms in a schedule in which the demands are not known. The second test is focused on proving the capacity of the new model and AMDAA on a schedule with a known demand. Due to the fact that there is no available real data of the origin–destination demand matrix for the passengers, the demand variable g_i has been defined for each train as a random number generated from 0 to 1. The file with the generated data has been uploaded to http://bit.ly/1eK5LM3.

[1] A video showing the working of SOFGA tool may be seen at http://youtu.be/771_wAqcphI.

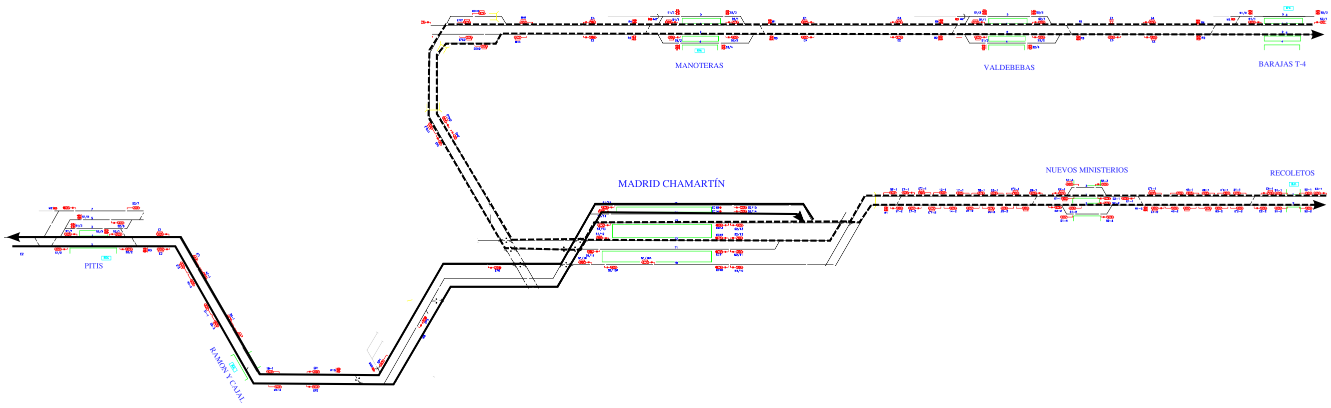
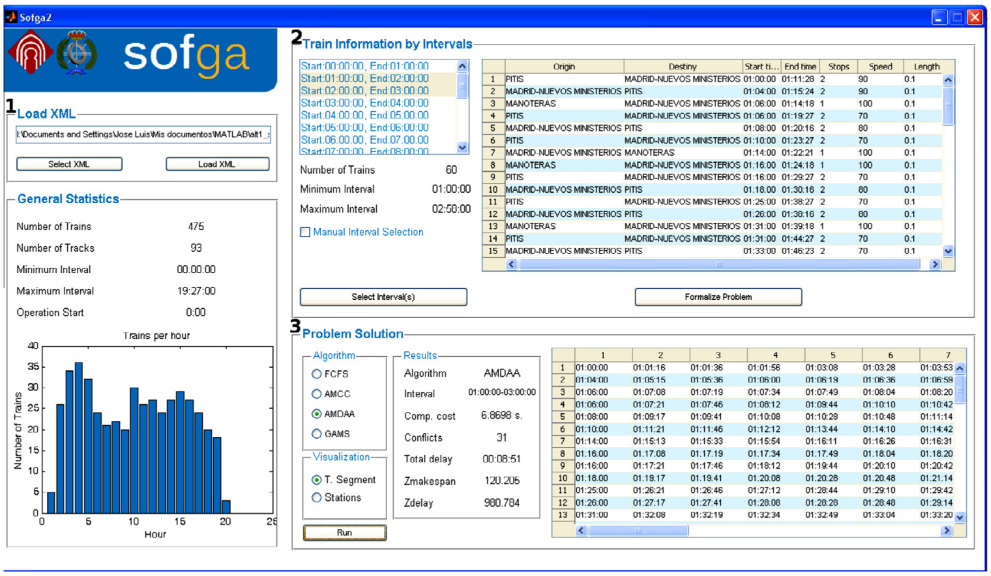


Fig. 5. Case study rail map.



1 **Load XML**

I:\Documents and Settings\Jose Luis\Mis documentos\MATLAB\alt1_

Select XML | Load XML

General Statistics

Number of Trains	475
Number of Tracks	93
Minimum Interval	00:00:00
Maximum Interval	19:27:00
Operation Start	0:00

2 **Train Information by Intervals**

Start:00:00:00, End:01:00:00
Start:01:00:00, End:02:00:00
Start:02:00:00, End:03:00:00
Start:03:00:00, End:04:00:00
Start:04:00:00, End:05:00:00
Start:05:00:00, End:06:00:00
Start:06:00:00, End:07:00:00
Start:07:00:00, End:08:00:00

Number of Trains	60
Minimum Interval	01:00:00
Maximum Interval	02:58:00

☐ Manual Interval Selection

	Origin	Destiny	Start ti...	End time	Stops	Speed	Length
1	PITIS	MADRID-NUEVOS MINISTERIOS	01:00:00	01:11:28	2	90	0.1
2	MADRID-NUEVOS MINISTERIOS	PITIS	01:04:00	01:15:24	2	90	0.1
3	MANOTERAS	MADRID-NUEVOS MINISTERIOS	01:06:00	01:14:18	1	100	0.1
4	PITIS	MADRID-NUEVOS MINISTERIOS	01:06:00	01:19:27	2	70	0.1
5	MADRID-NUEVOS MINISTERIOS	PITIS	01:08:00	01:20:16	2	80	0.1
6	PITIS	MADRID-NUEVOS MINISTERIOS	01:10:00	01:23:27	2	70	0.1
7	MADRID-NUEVOS MINISTERIOS	MANOTERAS	01:14:00	01:22:21	1	100	0.1
8	MANOTERAS	MADRID-NUEVOS MINISTERIOS	01:16:00	01:24:18	1	100	0.1
9	PITIS	MADRID-NUEVOS MINISTERIOS	01:16:00	01:29:27	2	70	0.1
10	MADRID-NUEVOS MINISTERIOS	PITIS	01:18:00	01:30:16	2	80	0.1
11	PITIS	MADRID-NUEVOS MINISTERIOS	01:25:00	01:38:27	2	70	0.1
12	MADRID-NUEVOS MINISTERIOS	PITIS	01:26:00	01:38:16	2	80	0.1
13	MANOTERAS	MADRID-NUEVOS MINISTERIOS	01:31:00	01:39:18	1	100	0.1
14	PITIS	MADRID-NUEVOS MINISTERIOS	01:31:00	01:44:27	2	70	0.1
15	MADRID-NUEVOS MINISTERIOS	PITIS	01:33:00	01:46:23	2	70	0.1

Select Interval(s) | Formalize Problem

3 **Problem Solution**

Algorithm: FCFS [✓ AMDAA] AMCC GAMS

Visualization: [✓ T. Segment] Stations

Results

Algorithm	AMDAA
Interval	01:00:00-03:00:00
Comp. cost	6.8698 s.
Conflicts	31
Total delay	00:08:51
Zmakespan	120.205
Zdelay	980.784

Run

	1	2	3	4	5	6	7
1	01:00:00	01:01:16	01:01:36	01:01:56	01:03:08	01:03:28	01:03:53
2	01:04:00	01:05:15	01:05:36	01:06:00	01:06:19	01:06:36	01:06:59
3	01:06:00	01:07:08	01:07:19	01:07:34	01:07:49	01:08:04	01:08:20
4	01:06:00	01:07:21	01:07:46	01:08:12	01:09:44	01:10:10	01:10:42
5	01:08:00	01:09:17	01:09:41	01:10:08	01:10:28	01:10:48	01:11:14
6	01:10:00	01:11:21	01:11:46	01:12:12	01:13:44	01:14:10	01:14:42
7	01:14:00	01:15:13	01:15:33	01:15:54	01:16:11	01:16:26	01:16:31
8	01:16:00	01:17:08	01:17:19	01:17:34	01:17:49	01:18:04	01:18:20
9	01:16:00	01:17:21	01:17:46	01:18:12	01:19:44	01:20:10	01:20:42
10	01:18:00	01:19:17	01:19:41	01:20:08	01:20:28	01:20:48	01:21:14
11	01:25:00	01:26:21	01:26:46	01:27:12	01:28:44	01:29:10	01:29:42
12	01:26:00	01:27:17	01:27:41	01:28:08	01:28:28	01:28:48	01:29:14
13	01:31:00	01:32:08	01:32:19	01:32:34	01:32:49	01:33:04	01:33:20

Fig. 6. The SOFGA tool.

Table 2
Relative error summary of results for 1-h interval.

	$BC_{\texttt{MAKESPAN}}$	$BC_{\texttt{DELAY}}$		FCFS	AMCC	AMDAA	
		$g_i = 1$	rand g_i			$g_i = 1$	rand g_i
$Z_{\texttt{MAKESPAN}}$	0	0.0519	1.5683	0.0435	0.0336	0.0336	1.2524
$Z_{\texttt{DELAY}}$ $g_i = 1$	1.6031	0	–	0.0039	0.0311	0.0311	–
$Z_{\texttt{DELAY}}$ rand g_i	2.9694	–	0	0.5278	0.5350	–	0.1767

Results can be seen in Tables 4 and 5 for intervals of 1 h, and in Table 6 for intervals of 2 h. These tables show the number of trains which make their services in each time interval and the computational cost, which is the average time taken to compute the optimal solution in seconds (CPU) taking into account that the AMDAA and $BC_{\texttt{DELAY}}$ are launched twice, once per experiment, and the other algorithms only once, due to their insensitivity to demand values. Furthermore the values of the objective functions $Z_{\texttt{MAKESPAN}}$ and $Z_{\texttt{DELAY}}$, evaluated in seconds, for each algorithm and for each set of values of g_i. In order to clarify these results, Tables 2 and 3 show the *mean relative error* MRE:

$$\mathrm{MRE} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{Z^i - Z^i_{BC}}{Z^i_{BC}}\right) \tag{17}$$

for each algorithm with respect to the optimal value obtained. The parameter n is the number of intervals evaluated, Z^i the objective function value found by the algorithm tested in an interval i and Z^i_{BC} the objective function value found by the BC method in the interval i.

As can be seen in Tables 2 and 3, these results show significant differences in more crowded intervals. With regard to the exact methods when $g_i = 1$ it is shown how the $BC_{\texttt{DELAY}}$ optimizes also

Table 3
Relative error summary of results for 2-h interval.

	$BC_{\mathtt{MAKESPAN}}$	$BC_{\mathtt{DELAY}}$		FCFS	AMCC	AMDAA	
		$g_i = 1$	rand g_i			$g_i = 1$	rand g_i
$Z_{\mathtt{MAKESPAN}}$	0	0.1030	1.8384	0.0877	0.0350	0.0350	0.8456
$g_i = 1$ $Z_{\mathtt{DELAY}}$	3.4496	0	–	0.0026	0.0163	0.0163	–
rand g_i $Z_{\mathtt{DELAY}}$	8.4051	–	0	0.4631	0.4682	–	0.1199

Table 4
Numerical results for 1-h interval (1/2).

Temporal interval	Number of trains	Algorithm	CPU (in seconds)	Exp. $g_i = 1$		Exp. rand g_i	
				$Z_{\mathtt{MAKESPAN}}$	$Z_{\mathtt{DELAY}}$	$Z_{\mathtt{MAKESPAN}}$	$Z_{\mathtt{DELAY}}$
1	5	$BC_{\mathtt{MAKESPAN}}$	0.31	0	0	0	0
		$BC_{\mathtt{DELAY}}$	0.67	0	0	0	0
		FCFS	0.11	0	0	0	0
		AMCC	0.14	0	0	0	0
		AMDAA	0.22	0	0	0	0
2	26	$BC_{\mathtt{MAKESPAN}}$	0.92	27.82	55.64	27.82	54.76
		$BC_{\mathtt{DELAY}}$	0.86	27.82	55.64	136.33	21.96
		FCFS	0.1	27.82	55.64	27.82	30.55
		AMCC	0.64	27.82	55.64	27.82	30.55
		AMDAA	0.77	27.82	55.64	136.33	21.96
3	34	$BC_{\mathtt{MAKESPAN}}$	2.39	89.21	2168.76	98.91	1635.45
		$BC_{\mathtt{DELAY}}$	1.63	89.91	774.56	138	525.95
		FCFS	0.14	89.91	774.56	89.91	559.95
		AMCC	1.06	120.21	846.28	120.21	548.19
		AMDAA	1.29	120.21	846.28	138	525.95
4	36	$BC_{\mathtt{MAKESPAN}}$	2.59	110.54	3180.53	110.54	1418.90
		$BC_{\mathtt{DELAY}}$	1.72	168.09	661.63	168.09	296.99
		FCFS	0.15	134.4	685.23	134.4	378.38
		AMCC	1.26	148.02	775.36	148.02	378.8
		AMDAA	1.53	148.02	775.36	110.54	338.54
5	32	$BC_{\mathtt{MAKESPAN}}$	1.5	90.69	2827.17	90.69	1172.34
		$BC_{\mathtt{DELAY}}$	1.23	139.64	508.91	90.69	329.89
		FCFS	0.13	139.64	508.91	139.64	357.79
		AMCC	0.92	92.09	586.12	92.09	342.31
		AMDAA	1.13	92.09	586.12	90.69	329.89
6	24	$BC_{\mathtt{MAKESPAN}}$	1.16	92.42	522.42	92.42	281.52
		$BC_{\mathtt{DELAY}}$	1.36	92.42	450.93	147.82	205.57
		FCFS	0.08	92.42	450.93	92.42	222.04
		AMCC	0.51	92.42	450.93	92.42	226.25
		AMDAA	0.61	92.42	450.93	147.82	220.67
7	21	$BC_{\mathtt{MAKESPAN}}$	0.66	28.38	85.14	28.38	30.55
		$BC_{\mathtt{DELAY}}$	0.77	28.38	56.76	28.38	14.58
		FCFS	0.07	28.38	56.76	28.38	14.58
		AMCC	0.37	28.38	56.76	28.38	14.58
		AMDAA	0.45	28.38	56.76	28.38	14.58
8	22	$BC_{\mathtt{MAKESPAN}}$	0.81	23.28	264.48	23.28	148.48
		$BC_{\mathtt{DELAY}}$	0.89	23.28	171.38	23.28	93.45
		FCFS	0.08	23.28	171.38	23.28	93.45
		AMCC	0.41	23.28	171.38	23.28	93.45
		AMDAA	0.5	23.28	171.38	23.28	93.45
9	20	$BC_{\mathtt{MAKESPAN}}$	0.81	28.38	113.52	28.38	26.18
		$BC_{\mathtt{DELAY}}$	0.46	28.38	28.38	130	8.28
		FCFS	0.06	28.38	28.38	28.38	18.73
		AMCC	0.36	28.38	28.38	28.38	18.73
		AMDAA	0.4	28.38	28.38	130	8.28
10	30	$BC_{\mathtt{MAKESPAN}}$	1.16	78	3316.68	78	1379.46
		$BC_{\mathtt{DELAY}}$	1.03	78	432.05	149.9	79.93
		FCFS	0.11	78	432.05	78	349.08
		AMCC	0.82	78	432.05	78	349.08
		AMDAA	0.99	78	432.05	120.21	293.94

the objective function $Z_{\mathtt{MAKESPAN}}$ in most cases. It doesn't occur so often vice versa, showing the main deficiency of $BC_{\mathtt{MAKESPAN}}$ when $g_i = 1$, focusing only on the makespan of the worst train of the schedule, so if a train is highly delayed and there is no possibility of facing that delay, the solution given could not take into account the disruptions occurring between other trains.

With respect to the results for the exacts methods when the demand g_i is generated randomly, it can be seen that the solutions found by the two exact approaches are meaningfully different, finding solutions based only on the objective function, which are worse if compared with the opposite objective.

Table 5
Numerical results for 1-h interval (2/2).

Temporal interval	Number of trains	Algorithm	CPU (in seconds)	Exp. $g_i = 1$		Exp. rand g_i	
				$Z_{\mathtt{MAKESPAN}}$	$Z_{\mathtt{DELAY}}$	$Z_{\mathtt{MAKESPAN}}$	$Z_{\mathtt{DELAY}}$
11	26	$BC_{\mathtt{MAKESPAN}}$	1.66	79.64	692.87	79.64	214.86
		$BC_{\mathtt{DELAY}}$	0.67	79.64	226.21	90.69	142.45
		FCFS	0.09	79.64	226.21	79.64	177.99
		AMCC	0.58	79.64	226.21	79.64	177.99
		AMDAA	0.71	79.64	226.21	90.69	142.45
12	27	$BC_{\mathtt{MAKESPAN}}$	1.33	27.82	248.65	27.82	87.58
		$BC_{\mathtt{DELAY}}$	0.75	27.82	161.66	148.38	60.02
		FCFS	0.1	27.82	161.66	27.82	68.04
		AMCC	0.63	27.82	181.66	27.82	73.48
		AMDAA	0.77	27.82	181.66	148.38	65.46
13	24	$BC_{\mathtt{MAKESPAN}}$	1.39	27.82	186.91	27.82	22.56
		$BC_{\mathtt{DELAY}}$	0.67	27.82	37.82	136.33	19.64
		FCFS	0.08	27.82	37.82	27.82	19.79
		AMCC	0.49	27.82	37.82	27.82	19.79
		AMDAA	0.6	27.82	37.82	136.33	19.64
14	27	$BC_{\mathtt{MAKESPAN}}$	1.38	77.18	505.85	77.18	175.5
		$BC_{\mathtt{DELAY}}$	0.81	77.18	229.42	218.99	167.5
		FCFS	0.1	87.82	239.21	87.82	180.11
		AMCC	0.63	77.18	229.42	77.18	178.68
		AMDAA	0.77	77.18	229.42	77.18	178.68
15	29	$BC_{\mathtt{MAKESPAN}}$	1.19	32.42	234.78	32.42	186.54
		$BC_{\mathtt{DELAY}}$	0.83	32.42	190.52	147.81	106.44
		FCFS	0.11	32.42	190.52	32.42	146.44
		AMCC	0.74	32.42	190.52	32.42	146.44
		AMDAA	0.9	32.42	190.52	32.42	146.44
16	27	$BC_{\mathtt{MAKESPAN}}$	1.34	32.42	253.3	32.42	105.66
		$BC_{\mathtt{DELAY}}$	0.8	32.42	243.3	32.42	80.74
		FCFS	0.1	32.42	243.3	32.42	80.74
		AMCC	0.64	32.42	263.3	32.42	89.52
		AMDAA	0.78	32.42	263.3	32.42	89.52
17	24	$BC_{\mathtt{MAKESPAN}}$	0.78	20.63	61.9	20.63	44.93
		$BC_{\mathtt{DELAY}}$	1.14	20.63	20.63	140.27	1.66
		FCFS	0.08	20.63	20.63	20.63	8.24
		AMCC	0.49	20.63	20.63	20.63	8.24
		AMDAA	0.6	20.63	20.63	140.27	1.66
18	20	$BC_{\mathtt{MAKESPAN}}$	0.56	23.28	93.1	23.28	54.59
		$BC_{\mathtt{DELAY}}$	0.77	23.28	46.55	147.82	36.14
		FCFS	0.07	23.28	46.55	23.28	43.86
		AMCC	0.33	23.28	46.55	23.28	43.86
		AMDAA	0.4	23.28	46.55	147.82	36.14
19	18	$BC_{\mathtt{MAKESPAN}}$	0.7	20.27	26.14	20.27	14.57
		$BC_{\mathtt{DELAY}}$	0.45	20.27	26.14	20.27	14.57
		FCFS	0.06	20.27	26.14	20.27	14.57
		AMCC	0.29	20.27	26.14	20.27	14.57
		AMDAA	0.35	20.27	26.14	20.27	14.57
20	3	$BC_{\mathtt{MAKESPAN}}$	0.31	0	0	0	0
		$BC_{\mathtt{DELAY}}$	0.53	0	0	0	0
		FCFS	0.01	0	0	0	0
		AMCC	0.02	0	0	0	0
		AMDAA	0.02	0	0	0	0

Table 6
Numerical results for 2-h interval.

Temporal interval	Number of trains	Algorithm	CPU (in seconds)	Exp. $g_i = 1$		Exp. rand g_i	
				$Z_{\mathtt{MAKESPAN}}$	$Z_{\mathtt{DELAY}}$	$Z_{\mathtt{MAKESPAN}}$	$Z_{\mathtt{DELAY}}$
1	31	$BC_{\mathtt{MAKESPAN}}$	1.3	27.82	681.96	27.82	411.65
		$BC_{\mathtt{DELAY}}$	1.03	27.82	55.64	136.33	21.96
		FCFS	0.14	27.82	55.64	27.82	30.55
		AMCC	0.97	27.82	55.64	27.82	30.55
		AMDAA	1.25	27.82	55.64	27.82	30.55
2	70	$BC_{\mathtt{MAKESPAN}}$	13.67	110.54	8697.22	110.54	4367.85
		$BC_{\mathtt{DELAY}}$	36.44	168.09	1436.19	253.33	843.89
		FCFS	0.53	134.39	1459.79	134.39	938.33
		AMCC	5.79	148.02	1649.66	148.02	942.66
		AMDAA	6.91	148.02	1649.66	165.82	1051.37
3	56	$BC_{\mathtt{MAKESPAN}}$	6.09	92.42	6131.14	92.42	2764.65
		$BC_{\mathtt{DELAY}}$	5.45	139.64	959.84	147.82	535.46
		FCFS	0.32	139.64	959.84	139.64	579.84
		AMCC	3.9	92.42	1037.05	92.42	568.57
		AMDAA	4.59	92.42	1037.05	92.42	551.92

Table 6 (*continued*)

Temporal interval	Number of trains	Algorithm	CPU (in seconds)	Exp. $g_i = 1$		Exp. rand g_i	
4	43	$BC_{\texttt{MAKESPAN}}$	2.39	28.38	859.47	28.38	424.19
		$BC_{\texttt{DELAY}}$	1.78	28.38	228.14	28.38	108.02
		FCFS	0.2	28.38	228.14	28.38	108.02
		AMCC	1.66	28.38	228.14	28.38	108.02
		AMDAA	2.05	28.38	228.14	28.38	108.02
5	50	$BC_{\texttt{MAKESPAN}}$	3.42	78	4036.61	78	2191.02
		$BC_{\texttt{DELAY}}$	2.8	78	460.63	149.91	88.21
		FCFS	0.26	78	460.43	78	367.82
		AMCC	2.43	78	460.43	78	367.82
		AMDAA	2.95	78	460.43	149.91	88.21
6	53	$BC_{\texttt{MAKESPAN}}$	3.9	79.64	4128.53	79.64	1846.47
		$BC_{\texttt{DELAY}}$	3.86	79.64	387.87	148.38	202.47
		FCFS	0.29	79.64	387.87	79.64	246.04
		AMCC	2.77	79.64	407.87	79.64	251.48
		AMDAA	3.92	79.64	407.87	90.69	215.94
7	51	$BC_{\texttt{MAKESPAN}}$	3.45	77.18	4339.31	77.18	1981.74
		$BC_{\texttt{DELAY}}$	3.24	77.18	267.23	218.99	187.15
		FCFS	0.27	87.82	277.03	87.82	199.91
		AMCC	2.52	77.18	267.23	77.18	198.46
		AMDAA	3.08	77.18	267.23	218.99	187.15
8	56	$BC_{\texttt{MAKESPAN}}$	4.03	32.42	1725.23	32.42	758.29
		$BC_{\texttt{DELAY}}$	4.13	32.42	433.82	187.19	187.18
		FCFS	0.32	32.42	433.82	32.42	227.18
		AMCC	3.17	32.42	453.82	32.42	235.96
		AMDAA	3.86	32.42	453.82	32.42	235.96
9	44	$BC_{\texttt{MAKESPAN}}$	1.86	23.28	628.45	23.28	381.67
		$BC_{\texttt{DELAY}}$	1.9	23.28	67.19	147.81	37.79
		FCFS	0.21	23.28	67.19	23.28	52.09
		AMCC	1.74	23.28	67.19	23.28	52.09
		AMDAA	2.16	23.28	67.19	140.27	45.52
10	21	$BC_{\texttt{MAKESPAN}}$	0.73	20.27	40.54	20.27	34.17
		$BC_{\texttt{DELAY}}$	0.56	20.27	26.14	20.27	14.58
		FCFS	0.07	20.27	26.14	20.27	14.58
		AMCC	0.4	20.27	26.14	20.27	14.58
		AMDAA	0.49	20.27	26.14	20.27	14.58

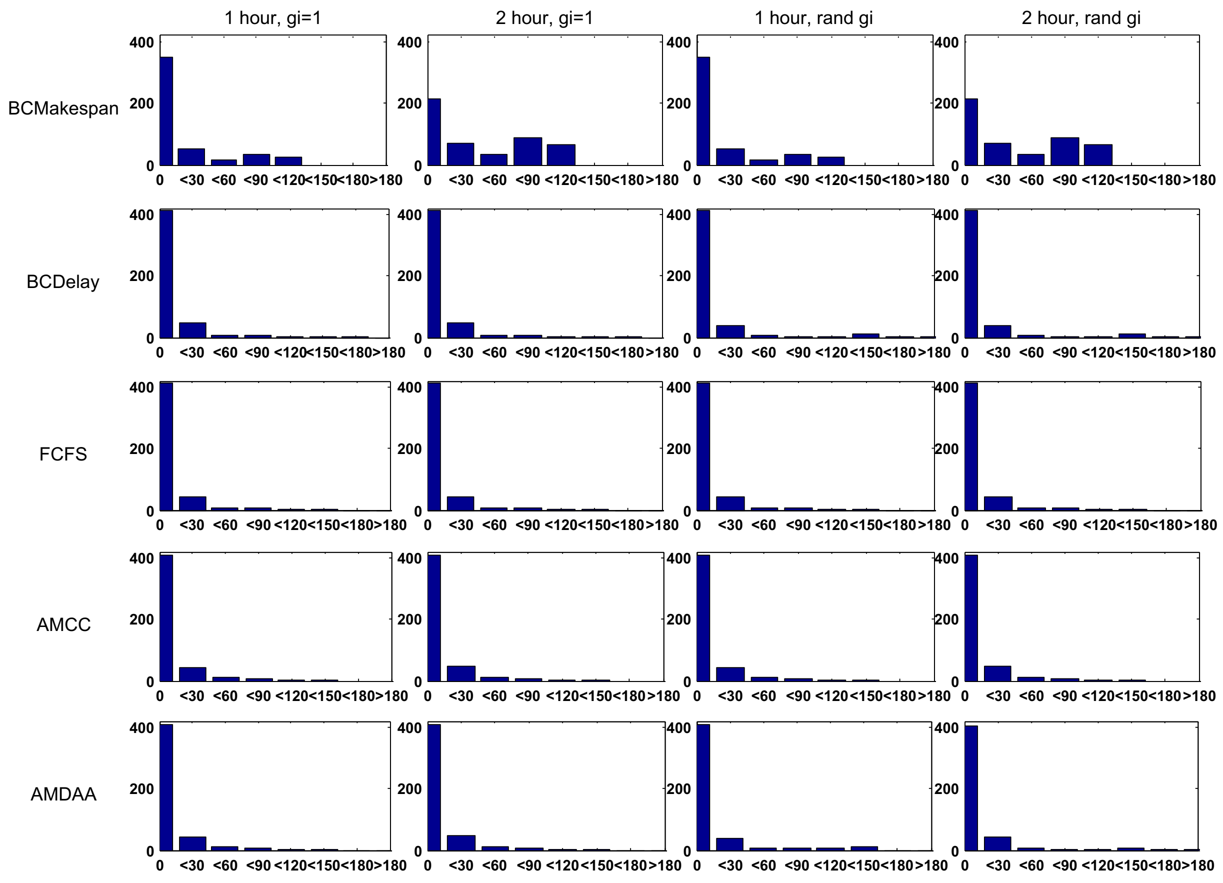


Fig. 7. Number of trains delayed per algorithm.

Table 7
Computational cost versus number of trains and amplitude of the time window.

Amplitude (h) of time window	Number of trains	Alternative Arcs	FCFS	AMDAA	AMCC	BC_{MAKESPAN}	BC_{DELAY}
1	5	160	0.11	0.22	0.14	0.31	0.67
2	26	3235	0.14	1.25	0.97	1.3	1.03
3	65	13931	0.7	7.64	4.99	11.53	10.19
4	101	34296	1.93	18.8	16.27	39.75	154.75
5	133	59236	3.96	41.98	37.43	95.8	294.38
6	157	81856	6.97	146.75	140.41	166.78	467.89
7	178	104255	9.77	240.7	221.71	217.82	738.44
8	200	130897	14.09	367.63	342.48	317.7	1165.13
9	220	157474	19.7	552.6	488.42	1736.42	6256.83

With regard to heuristics, in the case of $g_i = 1$, it can be shown that the FCFS performs relatively well for this problem because the railway network is simple and presents few overtaking points, and it finds near-optimal solutions for Z_{DELAY} objective function, but worse results if compared to Z_{MAKESPAN}. There are a few locations where the train ordering decision may be critical in terms of delay propagation. In this case, AMCC and AMDAA obtain the same results for all the cases.

In the case of the generated random demand, similar conclusions to those obtained with the exact algorithms are found. The main difference is that the FCFS algorithm finds worse solutions for the Z_{DELAY} objective function than AMDAA. Moreover, comparing AMCC and AMDAA, the results show how each algorithm focuses on the improvement of its own objective function without taking others into account, obtaining different solutions in most cases.

In conclusion, heuristic algorithms are sufficiently robust when considering both objective functions. It can be seen that AMDAA and AMCC obtain good solutions in this test problem. It is worth noting that AMCC is a little faster than AMDAA because the minimization of the average delay requires more intensive data structure updates compared to the minimization of the maximum delay. Exact methods can be applied to a real time context, but if they are applied, it should be considered that minimizing Z_{MAKESPAN} could imply poor results for Z_{DELAY} and vice versa.

Tables 4–6 show how the optimization problem of the Z_{MAKESPAN} objective function could obtain multiple optima due to the fact that other algorithms are capable of finding different solutions, because a different Z_{DELAY} value is obtained, but having the optimal Z_{MAKESPAN} value. This fact could motivate a bi-objective approach using both objective functions.

Fig. 7 depicts a set of histograms which represent the number of trains in the solution obtained with a given final delay. Each histogram is associated with an algorithm and test, and is grouped by intervals of 30 s. These histograms are used for analyzing the behavior of the algorithms qualitatively. The main conclusion obtained is that the BC_{MAKESPAN} algorithm adds more trains with an individual delay to the schedule while the other algorithms present a schedule with fewer delayed trains. Particularly in some cases the BC_{DELAY} and AMDAA algorithms allow few trains with a delay more than 180 s to minimize the overall Z_{DELAY} objective function.

5.2. Experiment 2

Experiment 2 discusses the viability of using the heuristic and exact methods in real problems. For this purpose each heuristic and exact method has been tested with 9 intervals of 1–9 h to evaluate, starting each of them at the initial hour of the schedule seen in Experiment 1, adding in each new experiment the trains of the next 1 h interval. The results obtained can be seen in Table 7. Each row presents the number of intervals or hours evaluated, the number of trains and alternative arcs evaluated and the computational cost in seconds of each algorithm for this interval.

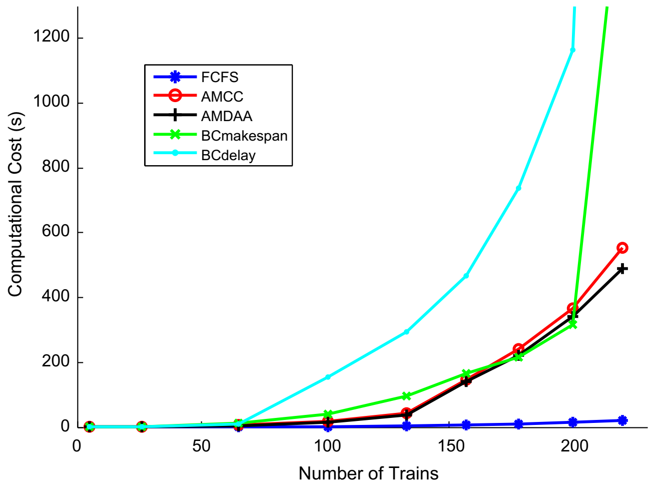


Fig. 8. Computational cost comparison.

This result shows that exact methods are viable for computing medium-sized problems in a real time horizon and that heuristics could obtain near-optimal results for large problems with a reduced computational cost.

Fig. 8 shows graphically the evolution of computational cost for each algorithm with regard to the number of trains (or alternative arcs) evaluated. It can be seen that exact algorithms' computational costs have exponential growth related to the number of alternative arcs of the interval, showing that the exact methods are viable for some sizes of the problem but for large sizes the computational cost may become prohibitive because of the number of operations required to find the exact solution, thus heuristic algorithms are needed.

6. Conclusions

This paper presents a novel weighted train delay based on demand approach for rescheduling railway systems. The model seeks to minimize the sum of the accumulated delays that passengers suffer on the railway network. This approach allows us, if an origin–destination matrix is available, to take demand into account in the rescheduling process.

In this work the AMCC algorithm, which was designed for the optimization of makespan, has been adapted to the AMDAA algorithm for minimizing the delays and so maximizing customer satisfaction.

A computational study with AMCC, FCFS, AMDAA and Branch-and-Cut methods has been carried out using a perturbation of the original timetable planned by Renfe Cercanias Madrid in order to test the performance of the algorithms for resolving railway clashes and to ascertain their goodness.

These trials have shown that if demand is unknown (i.e. $g_i = 1$) heuristic algorithms provide good solutions to both makespan and delay objectives, but are unable to reach the goodness of the solutions given by exact methods. Moreover, an exact method for minimizing makespan can lead to unsatisfactory solutions from the point of view of costumer satisfaction in most cases. In the case of randomly generated demand values, each algorithm or exact method only considers its main purpose, obtaining poor solutions for the rest, so for future work a multiobjective approach should be studied.

From the optimization standpoint, the heuristic and exact methods provide acceptable solutions within a reasonable time window for the temporal horizon of 1 h, which is the time horizon of practical interest to rail traffic controllers for real-time purposes in regional railway systems because of their journey time. It has also been proved that with exact methods the computational cost can grow very quickly, making them impractical, and requiring heuristic algorithms such as those proposed in this paper.

Acknowledgements

This research has been financed by the Ministerio de Economia y Competitividad of Spain with the TRA-2011-27791-C03-03 research project. Also thanks to Javier Ruiz Lazaro for his help in carrying out the computational experiments.

References

Almodóvar, M., García-Ródenas, R., 2013. On-line reschedule optimization for passenger railways in case of emergencies. Computers and Operations Research 40 (3), 725–736.

Cacchiani, V., Toth, P., 2012. Nominal and robust train timetabling problems. European Journal of Operational Research 219 (3), 727–737.

Cadarso, L., Marín, T., Maróti, G., 2013. Recovery of disruptions in rapid transit networks. Transportation Research Part E: Logistics and Transportation Review 53 (1), 15–33.

Caprara, A., Fischetti, M., Toth, P., 2002. Modeling and solving the train timetabling problem. Operations Research 50 (5), 851–861+916.

Caprara, A., Kroon, L., Monaci, M., Peeters, M., Toth, P., 2007. Chapter 3. Passenger railway optimization. Handbooks in Operations Research and Management Science 14.

Cordeau, J.-F., Toth, P., Vigo, D., 1998. A survey of optimization models for train routing and scheduling. Transportation Science 32, 380–404.

Corman, F., D'Ariano, A., Pacciarelli, D., Pranzo, M., 2009. Evaluation of green wave policy in real-time railway traffic management. Transportation Research Part C: Emerging Technologies 17 (6), 607–616.

Corman, F., D'Ariano, A., Pacciarelli, D., Pranzo, M., 2010. A tabu search algorithm for rerouting trains during rail operations. Transportation Research Part B: Methodological 44 (1), 175–192.

Corman, F., D'Ariano, A., Pranzo, M., Hansen, I., 2011b. Effectiveness of dynamic reordering and rerouting of trains in a complicated and densely occupied station area. Transportation Planning and Technology 34 (4), 341–362.

Corman, F., D'Ariano, A., Hansen, I.A., Pacciarelli, D., 2011a. Optimal multi-class rescheduling of railway traffic. Journal of Rail Transport Planning and Management 1 (1), 14–24.

Corman, F., D'Ariano, A., Pacciarelli, D., Pranzo, M., 2012a. Bi-objective conflict detection and resolution in railway traffic management. Transportation Research Part C: Emerging Technologies 20 (1), 79–94.

Corman, F., D'Ariano, A., Pacciarelli, D., Pranzo, M., 2012b. Optimal inter-area coordination of train rescheduling decisions. Transportation Research Part E: Logistics and Transportation Review 48 (1), 71–88.

D'Ariano, A., 2008. Improving real-time train dispatching: models, algorithms and applications (Ph.D. thesis). The Netherlands TRAIL Research School, tRAIL Thesis Series No. T2008/6.

D'Ariano, A., Pacciarelli, D., Pranzo, M., 2007a. A branch and bound algorithm for scheduling trains in a railway network. European Journal of Operational Research 183 (2), 643–657.

D'Ariano, A., Pranzo, M., Hansen, I.A., 2007b. Conflict resolution and train speed coordination for solving real-time timetable perturbations. IEEE Transactions on Intelligent Transportation Systems 8 (2), 208–222.

D'Ariano, A., Corman, F., Pacciarelli, D., Pranzo, M., 2008a. Reordering and local rerouting strategies to manage train traffic in real time. Transportation Science 42 (4), 405–419.

D'Ariano, A., Pacciarelli, D., Pranzo, M., 2008b. Assessment of flexible timetables in real-time traffic management of a railway bottleneck. Transportation Research Part C: Emerging Technologies 16 (2), 232–245.

Dollevoet, T., Huisman, D., Schmidt, M., Schöbel, A., 2009. Delay management with re-routing of passengers. OpenAccess Series in Informatics 12.

Dollevoet, T., Corman, F., D'Ariano, A., Huisman, D., 2012. An iterative optimization framework for delay management and train scheduling, Tech. Rep., Erasmus School of Economics (ESE).

Espinosa-Aranda, J.L., García-Ródenas, R., 2012. A Discrete Event-Based Simulation Model for Real-Time Traffic Management in Railways. Journal of Intelligent Transportation Systems 16 (2), 94–107.

Kanai, S., Shiina, K., Harada, S., Tomii, N., 2011. An optimal delay management algorithm from passengers' viewpoints considering the whole railway network. Journal of Rail Transport Planning and Management 1 (1), 25–37.

Kraay, D.R., Harker, P.T., 1995. Real-time scheduling of freight railroads. Transportation Research Part B 29 (3), 213–229.

Liu, J., Li J., Chen, F., Zhou, Y., 2010. Review on station-to-station OD matrix estimation model and algorithm for urban rail transit. In: ICCMS 2010 – 2010 International Conference on Computer Modeling and Simulation, vol. 3, pp. 149–153.

Lusby, R.M., Larsen, J., Ehrgott, M., Ryan, D., 2011. Railway track allocation: models and methods. OR Spectrum 33 (4), 843–883.

Mascis, A., Pacciarelli, D., 2002. Job-shop scheduling with blocking and no-wait constraints. European Journal of Operational Research 143 (3), 498–517.

Mazzarello, M., Ottaviani, E., 2007. A traffic management system for real-time traffic optimisation in railways. Transportation Research Part B: Methodological 41 (2), 246–274.

Min, Y., Park, M., Hong, S., Hong, S., 2011. An appraisal of a column-generation-based algorithm for centralized train-conflict resolution on a metropolitan railway network. Transportation Research Part B: Methodological 45 (2), 409–429.

Narayanaswami, S., Rangaraj, N., 2013. Modelling disruptions and resolving conflicts optimally in a railway schedule. Computers and Industrial Engineering 64 (1), 469–481.

Papadimitriou, C., Kanellakis, P., 1980. Flow shop scheduling with limited temporary storage. Journal Association Computing Machine 27, 533–549.

Samà, M., D'Ariano, A., Pacciarelli, D., 2013. Rolling horizon approach for aircraft scheduling in the terminal control area of busy airports. Procedia – Social and Behavioral Sciences 80 (0), 531–552.

M. Schachtebeck, A. Schöbel, 2007. Algorithms for delay management with capacity constraints, Tech. Rep., ARRIVAL Project.

Tornquist, J., Persson, J.A., 2007. N-tracked railway traffic re-scheduling during disturbances. Transportation Research Part B 41, 342–362.

Tornquist Krasemann, J., 2012. Design of an effective algorithm for fast response to the re-scheduling of railway traffic during disturbances. Transportation Research Part C: Emerging Technologies 20 (1), 62–78.

Wang, Y., De Schutter, B., van den Boom, T., Ning, B., 2013. Optimal trajectory planning for trains – a pseudospectral method and a mixed integer linear programming approach. Transportation Research Part C: Emerging Technologies 29, 97–114.