Università degli Studi di Roma Tre
Dipartimento di Informatica e Automazione
Via della Vasca Navale, 79 – 00146 Roma, Italy

Bi-objective conflict detection and resolution in railway traffic management

Francesco Corman[1], Andrea D'Ariano[2], Dario Pacciarelli[2], Marco Pranzo[3]

RT-DIA-157-09 **Settembre 2009**

(1) Department of Transport and Planning, Delft University of Technology,
Stevinweg, 1 - 2628 CN Delft, The Netherlands.
`f.corman@tudelft.nl`

(2) Dipartimento di Informatica e Automazione, Università degli Studi Roma Tre,
via della vasca navale, 79 - 00146 Roma, Italy.
{`a.dariano;pacciarelli`}`@dia.uniroma3.it`

(3) Dipartimento di Ingegneria dell'Informazione, Università degli Studi di Siena,
via Roma, 56 - 53100 Siena, Italy.
`pranzo@dii.unisi.it`

This work has been partially funded by the Italian Ministry of Research, Grant number RBIP06BZW8, project FIRB "Advanced tracking system in intermodal freight transportation".

ABSTRACT

Railway conflict detection and resolution is the daily task faced by railway managers that consists of adjusting train schedules whenever disturbances make the timetable infeasible. The main objective pursued by the infrastructure managers in this task is minimization of train delays, while train operating companies are also interested in other indicators of passengers dissatisfaction. The two objectives are conflicting whenever delay reduction requires cancellation of some connections between train services, which hampers passengers transfer. In fact, the infrastructure company and the train operating companies discuss on which connection to keep or drop in order to reach a compromise solution.

In this paper we face the bi-objective problem of minimizing delays and missed connections to provide a set of feasible non-dominated schedules to support this decisional process. We use a detailed alternative graph model to ensure schedule feasibility and develop two heuristic algorithms to compute the Pareto front of non-dominated schedules. Our computational study, based on a complex and densely occupied Dutch railway network, shows that good coordination of connected train services is important to achieve real-time efficiency of railway services since the management of connections may heavily affect train punctuality. The two algorithms approximate accurately the Pareto front within a limited computational time.

Keywords: Railway; Train Scheduling; Multi-Objective Optimization; Alternative Graph; Delay Management.

1 Introduction

Railway managers usually manage performance through a carefully designed plan of operations, called timetable, that defines well in advance routes, orders and timings for all trains running in the network [3]. In the Netherlands, the timetable is designed in order to provide good connectivity between different train services for a large number of origins and destinations. Transfer connections allow the movement of passengers from one train to another. This means that, for each pair of connected train services, the waiting train must depart sufficiently later with respect to the feeder train.

During operations, train traffic can be seriously disturbed when delays, accidents or technical problems occur on railway lines. Major disturbances cause primary delays that propagate as consecutive, or secondary, delays to other trains in the network. In such case, short-term adjustments to the timetable (i.e., train schedule modifications defined shortly before execution) are required in order to limit the negative effects of the disturbances. The problem faced by railway managers in this real-time process is called Conflict Detection and Resolution (CDR).

Delay propagation is due to trains sharing part of the infrastructure or to connected services, i.e., trains constrained by transfer or rolling stock connections. Since in the Dutch timetable passenger transfers are typically up to six minutes long, if the feeder train is late this delay easily propagate to the waiting train. In fact, transfer connections are an important cause of the delay propagation in complex and densely used areas [9, 19, 23].

Transfer connections are relevant to the passenger satisfaction but do not affect the feasibility of railway operations, therefore one of the possible dispatching countermeasures to reduce delay propagation is the cancellation of some scheduled connections. This action reduces overall train delays but causes an extra delay to the passengers affected by the missed connection. Train operating companies are therefore also interested in keeping as most connections as possible even in presence of disturbed traffic conditions. In fact, infrastructure managers discuss with train operating companies on which connections must be kept when regulating railway traffic. To support this negotiation process, in this paper we deal with the *Bi-objective Conflict Detection and Resolution* (BCDR) problem of finding a set of feasible schedules with a good trade-off between the minimization of train delays and the maximization of respected transfer connections.

The BCDR problem is closely related to the Delay Management (DM) problem introduced by Schöbel [20]. The latter problem adopts a passenger point of view, and aims at the minimization of the sum of all delays over all passengers at their final destination. In this paper we choose a train point of view. A value is associated to each connection (e.g., expressed in terms of number of passengers who get the connection) and one objective function is maximization of the total value of respected connections. A further difference is that the DM problem does not take into account the limited capacity of the railway network (i.e., does not deal with the CDR problem), which is the main issue of this paper.

We formulate the BCDR problem as an alternative graph [15], in which signaling and safety aspects are taken into account according to the blocking time theory [10]. In an alternative graph variables are starting times of the relevant events. *Fixed arcs* are used to represent fixed relations between starting times. Alternative arcs are used to represent decisions. We distinguish two types of alternative arcs: A *pair of alternative arcs* is used to represent a sequencing decision, a *connection arc* represents enforcement of a connection.

The solution procedure consists of computing a maximal set of non-dominated solutions for the BCDR problem, called the *Pareto front*. The solution strategy adopted in this paper consists of iteratively solving the CDR problem (with fixed connections) and then searching for a different set of connections to be enforced. The CDR problem is solved by the branch and bound algorithm described in [5], whose objective function is minimization of the maximum consecutive delay at scheduled stops and at the exit points from the area under study. For the selection of the connections to be enforced, we develop and test two new algorithms for the BCDR problem.

The computational study is based on a complex and densely occupied Dutch railway network. A set of disturbances is generated for different values of train delays. For each perturbed situation, non-dominated solutions to the BCDR problem are computed. Comparison between the exhaustive search and the two heuristic algorithms, in terms of train delays, value of respected transfer connections and computation time, demonstrates the effectiveness of the two heuristics. We also discuss which connections are more critical to delay minimization and which non-dominated solutions exhibit interesting theoretical properties.

The paper is organized as follows. Section 2 reviews the literature most related with the BCDR problem. Section 3 describes the BCDR problem, its formulation and the solution algorithms. Section 4 reports on our computational experiments. Section 5 discusses the main achievements and future research directions.

2 Related literature

The literature on railway traffic management experienced little attention to the management of transfer connections. A stream of research, weakly related to this paper, deals with *timetable synchronization*, i.e., the problem of designing a timetable in which train departures and arrivals at each station are coordinated to offer effective connections and reduced waiting time to transferring passengers [21, 22, 25].

Models closer to this paper deal with the reaction to disturbances performed by railway infrastructure managers. This short-term activity requires to take two strongly related decisions. One deals with choosing which connections to keep and which to drop in order to reduce delay and passenger inconvenience. This decision requires coordination with train operating companies and possibly with other public transportation companies (e.g., bus companies). The other decision is how to adjust the timetable to make it compliant with traffic regulations and with the current train positions. The main issue here is the feasibility of the train schedules, while the goals include delay minimization and energy consumption.

Models and algorithms to support the former decision have been proposed by several authors in the context of delay management (DM) [20, 11, 8]. Schöbel [20] proposes a mixed integer linear programming model for the DM problem with minimization of the sum of the delays faced by passengers at their destinations. Heilporn et al. [11] solve the same problem by a branch and cut procedure and by a constraint generation approach. Ginkel and Schöbel [8] present a bi-criteria model for a different version of the DM problem, with minimization of the delay of all vehicles at all stations and the weighted number of missed connections. To formulate the problem they use an event-activity network and then find all the Pareto solutions by extending a solution procedure proposed by Demeulemeester et al. [7].

The timetable adjustment must take into consideration the limited capacity of the railway network and the development of feasible schedules for the trains taking into account the maintained connection constraints. Albrecht and Oettich [2] present an algorithm for the dynamic modification of train running times in such a way that the probability of arriving on-time to transfer to other means of public transport can be increased and the overall energy consumption of train operation is minimal. Wegele and Schnieder [24] use genetic algorithms to reschedule trains with the objective of minimizing passenger annoyance, e.g., delays, change of platform stops and missed connections. Examples of application on a part of the German railways are reported. D'Ariano et al. [6, 4, 5] develop the real-time traffic management system ROMA (Railway traffic Optimization by Means of Alternative graphs), able to provide reliable solutions to the CDR problem in real-time for a dispatching area of practical size. In [6, 4], the authors also analyze the effects of transfer connections in combination with different scheduling algorithms for solving the CDR problem. The analysis, limited to the two extreme cases with all connections maintained or no connection enforced, shows that transfer connections are an important cause of delay propagation.

Summarizing this overview of the literature, we can observe that existing contributions tend to focus on one aspect of the real-time railway management problem and do not capture the overall process of building feasible schedules while taking into account the needs of different stakeholders (infrastructure company, train operating companies, passengers, bus operating companies and so on). There is therefore a need of research contributions focusing on the development of feasible and reliable train schedules while taking into account many different objectives, such as energy consumption, optimal use of infrastructure, minimization of train and passenger delays and so on.

3 BCDR problem

This section describes the BCDR problem, presents its alternative graph formulation, gives an illustrative example and introduces algorithms for its resolution.

3.1 Definitions and problem description

In its basic form a rail network is composed of stations, links and block sections separated by signals. Signals, interlocking and Automatic Train Protection (ATP) ensure safety of railway traffic by imposing a minimum safety separation between trains, setting up feasible routes and enforcing speed restrictions on running trains. Signals are located before every junction as well as along the lines and inside the stations. A *block section* is a track segment between two signals and may host at most one train at a time.

The passage of a train through a particular block section is called an *operation*. A *route* of a train is a sequence of operations to be processed during a *service* (train run). The *timing* of a route specifies the starting time t_i of each operation in the route. Each operation requires a traveling time, called *running time*, which depends on the actual speed profile followed by the train while traversing the block section. A speed profile is constrained by the rolling stock characteristics (maximum speed, acceleration and braking rates), physical infrastructure characteristics (maximum allowed speed and signaling system) and driver behavior (coasting, braking and acceleration profiles when approaching the variable aspects of the signaling system in use).

The running time of a train on a block section starts when its head (the first axle) enters the block section, ends when its head enters the subsequent block section, and is computed on the basis of its speed profile. We assume that the running time is known in advance since all trains travel at their scheduled speed whenever possible, and recover small delays by using the time margins inserted in the timetable.

In yards or complex station interlocking areas, the routes for the individual trains need to be setup before entering and cleared after leaving. Safety regulations impose a minimum distance separation between the trains running in the network, which translates into a minimum *setup time* between the exit of a train from a block section and the entrance of the subsequent train into the same block section. The setup time considers the time between the entrance of the train head in a block section and the exit of its tail (the last axle) from the previous one, plus additional time margins to release the occupied route and to take into account the sighting distance (see, e.g., [10]).

We consider a timetable which describes the movements of all trains running within a given time period of traffic prediction, specifying, for each train, the planned arrival/passing times at a set of relevant points along its route (e.g., stations, junctions, and the exit point of the network). At stations, a train is not allowed to depart from a platform stop before its scheduled departure time and is considered late if arriving at the platform later than its scheduled arrival time. At a platform stop, the scheduled waiting time of each train is called *dwell time*. Additional connection constraints relate to rolling stock circulation and to passenger satisfaction. The latter requires minimum transfer times between connected passenger trains in order to allow passengers to alight from one train, move to another platform track and board the other train. The former requires that a train departure must be postponed if part of its rolling stock is actually used to run another service. Waiting time may also include the time for coupling-decoupling units.

Timetables are designed to satisfy traffic regulations and other additional constraints such as connection constraints. However, unexpected events occur during operations, which cause delays with respect to the operations scheduled in the timetable. A *conflict* occurs when two or more trains claim the same block section simultaneously, and a decision on the train ordering has to be taken.

The delay may propagate causing a domino effect of increasing disturbances. We define an *entrance perturbation* as a set of train delays at their entrance in a dispatching area. A set of trains causes a *deadlock* when each train in the set claims a block section ahead which is not available, e.g., due to the occupation/reservation for another train in the set. Running time prolongation may occur because of conflicts between trains or technical failures. Dwell time perturbations are due to traffic delays at stations, passenger boarding/alighting as well as personnel temporary unavailability.

Real-time railway traffic management copes with temporary infeasibility by adjusting the timetable of each train, in terms of routing and timing, and/or by resequencing the trains at the entrance of each merging/crossing point. The railway traffic is predicted over a given time horizon of traffic prediction. The task of dispatchers is to regulate traffic in a given dispatching area with the main objective of minimizing train delays in such a way that the new schedule is compliant with railway operating rules and with the entrance position of each train. The latter information is taken into account in the computation of the *release time* of each train that is the expected time, with respect to the starting time t_0 of traffic prediction, at which the current train enters its first block section in the area under study. The *total delay* is the positive difference between the estimated train arrival time and the scheduled arrival time at a relevant point in the network, and can be divided into two parts. The *initial delay* (primary delay) is caused by original failures and disturbances and can only be recovered by exploiting available running time reserves, i.e., by letting the trains traveling at their maximum speed. The *consecutive delays* (secondary delays) are caused by the hinder from other trains.

The BCDR problem can be defined as follows: given a railway network, a set of train routes and passing/stopping times at each relevant point in the network, and the position and speed of each train being known at time t_0, find a set of non-dominated deadlock-free schedules, compatible with the initial position of each train and such that all conflicts between consecutive trains are solved, each train enters the network at its release time, no train departs from a relevant point before its minimum scheduled departure time, the given rolling stock constraints are respected, the constraints due to the enforced (passengers) transfer connections are also respected, trains arrive at the relevant points with the smallest possible consecutive delay and the selected transfer connections return the highest possible connection value.

3.2 Alternative graph formulation

The BCDR problem can be formulated as a special bi-objective job shop scheduling problem. This formulation corresponds to a particular *disjunctive program*, i.e., a linear program with logical conditions involving operation "or" ($\vee$, disjunction).

$$
\begin{array}{lll}
\min & \left(t_n - t_0 \;;\; -\sum_{(i,j)\in C} v_{ij}\delta(t_j - t_i - w_{ij})\right) & \\
s.t. & t_j - t_i \geq w_{ij} & (i,j) \in F \\
 & (t_j - t_i \geq w_{ij}) \vee (-) & (i,j) \in C \\
 & (t_j - t_{\sigma(i)} \geq w_{\sigma(i)j}) \vee (t_i - t_{\sigma(j)} \geq w_{\sigma(j)i}) & ((\sigma(i),j),(\sigma(j),i)) \in A
\end{array}
\tag{1}
$$

In Problem (1), a variable t_i, for $i = 1, \ldots, n-1$, is the starting time of operation i and corresponds to the entrance time of a train in the associated block section. Operation 0 is a dummy operation called *start* that precedes all the other operations. Similarly, n is a dummy operation called *end* that follows all the other operations. The function $\delta(x)$ is equal to 1 if $x \geq 0$ and is equal to 0 if $x < 0$. The set C is the set of connections. We associate a value v_{ij} to each connection constraint $(i,j) \in C$.

Fixed constraints in F model feasible timing for each train on its specific route. For each operation i, let $\sigma(i)$ be the operation which follows i on the route of the associated train. In a feasible solution, the precedence relation $t_{\sigma(i)} \geq t_i + w_{i\sigma(i)}$ must hold, where $w_{i\sigma(i)} > 0$ is the minimum running time of operation i. Other fixed constraints are used to model release times and train delays, as in [5].

Sets A and C are disjunctive, i.e., are composed by pairs of constraints, possibly dummy. In a feasible solution exactly one constraint from each pair has to be chosen. The dummy constraint corresponds to adding no constraint to the problem. Connection constraints in C impose or not a connection between two operations. If a connection is enforced between operations i and j, then a minimum time w_{ij} must be ensured between t_i and t_j. In our formulation, if the connection is not enforced, then a dummy constraint $(-)$ is added to the problem. This means that inequality $t_j - t_i \geq w_{ij}$ can be satisfied also when the connection is not enforced. Alternative constraints in A represent the ordering decision between trains. For each pair i and j of operations associated with the entrance of two trains in the same block section, we introduce the disjunction $(t_j - t_{\sigma(i)} \geq w_{\sigma(i)j}) \vee (t_i - t_{\sigma(j)} \geq w_{\sigma(j)i})$, where $w_{\sigma(i)j} > 0$ and $w_{\sigma(j)i} > 0$ are the minimum required setup times.

A train schedule corresponds to the set of the starting time of each operation. The schedule is feasible if it satisfies all conjunctions (set F) and at least a constraint for each disjunctive pair (sets A and C). The quality of a schedule is measured in terms of punctuality and total value of the satisfied transfer connections (with negative value in order to have two minimum cost functions). Punctuality is measured with the maximum consecutive delay of all trains at a set of relevant points (scheduled stops and the exit of the network). A feasible solution is non-dominated if there is no other feasible solution with a smaller value of both objective functions.

Mascis and Pacciarelli [15] show that the alternative graph is a suitable model for the job shop scheduling problem with additional constraints, such as blocking and no-wait, also occurring in the railway context. The BCDR problem can be formulated as the alternative graph $\mathcal{G} = (N, F, C \cup A)$, where N is the set of nodes, F is the set of fixed arcs, C is the set of connection arcs, and A is the set of alternative arcs. In terms of alternative graph a solution is a graph $(N, F, S^C \cup S^A)$ in which S^C is a selection of arcs from C and S^A is a selection of arcs from A, obtained by choosing one arc from each pair in the corresponding set. The solution is feasible if the graph contains no positive length cycles. The objective functions are the length of the longest path between the dummy nodes 0 and n and the total value of the arcs in S^C (the dummy arcs having weight zero).

3.3 Illustrative example

Figure 1 presents 4 trains (A, B, C and D) running through a small network composed of 10 block sections and a station (Q). All trains have a scheduled stop at the station (A at block section 7, B at block section 6, C and D at block section 8). Three transfer connections (A-D, D-A and C-B) are considered to represent the minimum waiting time for a feeder train (e.g., C) to let passenger moving to the waiting train (e.g., B).

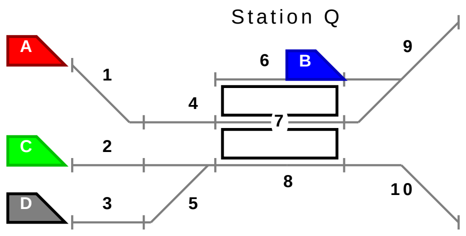


Figure 1: Trains running in a small railway area.

The BCDR problem is modeled by the alternative graph formulation of Figure 2. For the sake of clarity, a node of the graph can be identified by a pair (train, block section), by a pair (train, scheduled stop) or by a pair (train, exit point), except for the dummy nodes 0 and n. There are three types of arcs:

Fixed arcs are depicted in solid black color in Figure 2 and can be divided in running time arcs, dwell time arcs, release time arcs and due date arcs (the latter are used to compute the delay of each train at its relevant points). The weight on each fixed arc (i, j) is the associated time w_{ij} of Problem (1). For example, for train A there is one

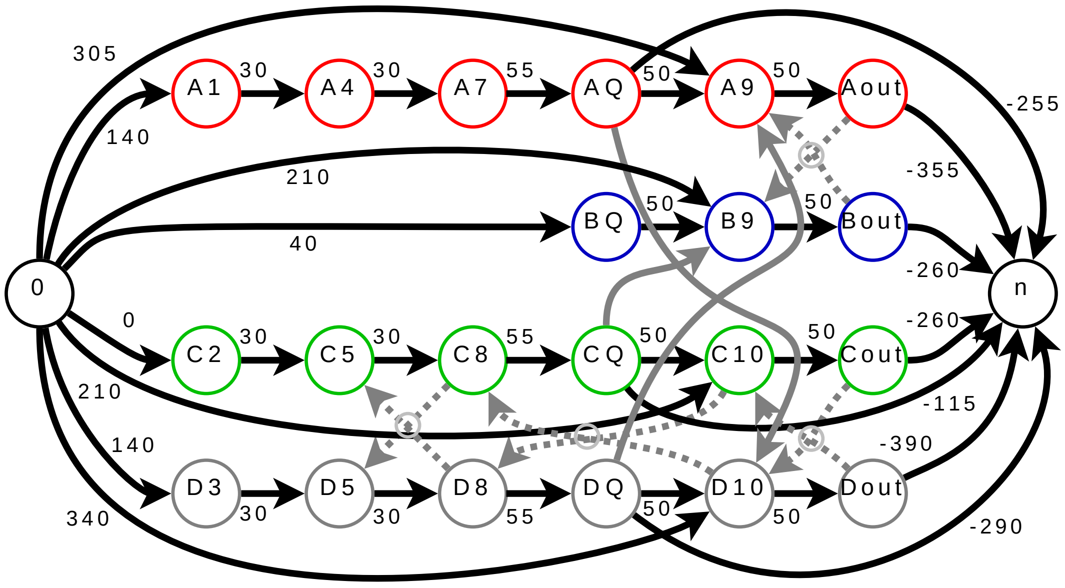


Figure 2: Alternative graph for the illustrative example.

dwell time arc (AQ, A9) and four running time arcs (A1, A4), (A4 , A7), (A7, AQ) and (A9, Aout). Release time arcs (respectively, due date arcs) link operations with the dummy node 0 (respectively, n). A release arc is weighted with the scheduled time the associated train enters the network or leaves the associated station, while each due date arc is weighted with the scheduled time the corresponding train should exit the network or should enter the associated platform (but with negative value). For example, for train A there are two release arcs (0, A1) and (0, A9), and two due date arcs (AQ, n) and (Aout, n).

Connection arcs are depicted in Figure 2 with gray solid arcs and represent the minimum separation time between the start of the dwell time of the feeder train and the departure of the waiting train. The weight of each connection arc corresponds to the minimum separation time between connected services. In this example, there are three connection arcs (DQ, A9), (AQ, D10) and (CQ , B9), with weight equal to 50 for each arc. For readability, connection arc weights are not reported in figure.

Alternative arcs are depicted with pairs of dotted gray arcs with a small circle in Figure 2, and are used to avoid train conflicts at each block section by specifying train orders and setup times. The alternative arcs are paired. Each arc of the pair enforces the order between two trains and is weighted with the corresponding setup time. In the example there are four alternative pairs. The weight of each alternative arc is 10 and it is not reported in figure for readability. For example, trains A and B share block section 9 and require the alternative pair ((Aout, B9), (Bout, A9)).

Figure 3 reports four time-distance diagrams of BCDR solutions for the example of Figure 1. In Figure 3 (a) the timetable is shown. Train A is scheduled to enter the network at time 140, arrive at the station (platform 7) at time 255, leave the station at time 305 and leave the network at time 355. Train B is scheduled to enter the station at platform 6 at time 40, leave the station at time 210 and leave the network at time 260. Train C is scheduled to enter the network at time 0, arrive at the station (platform 8)

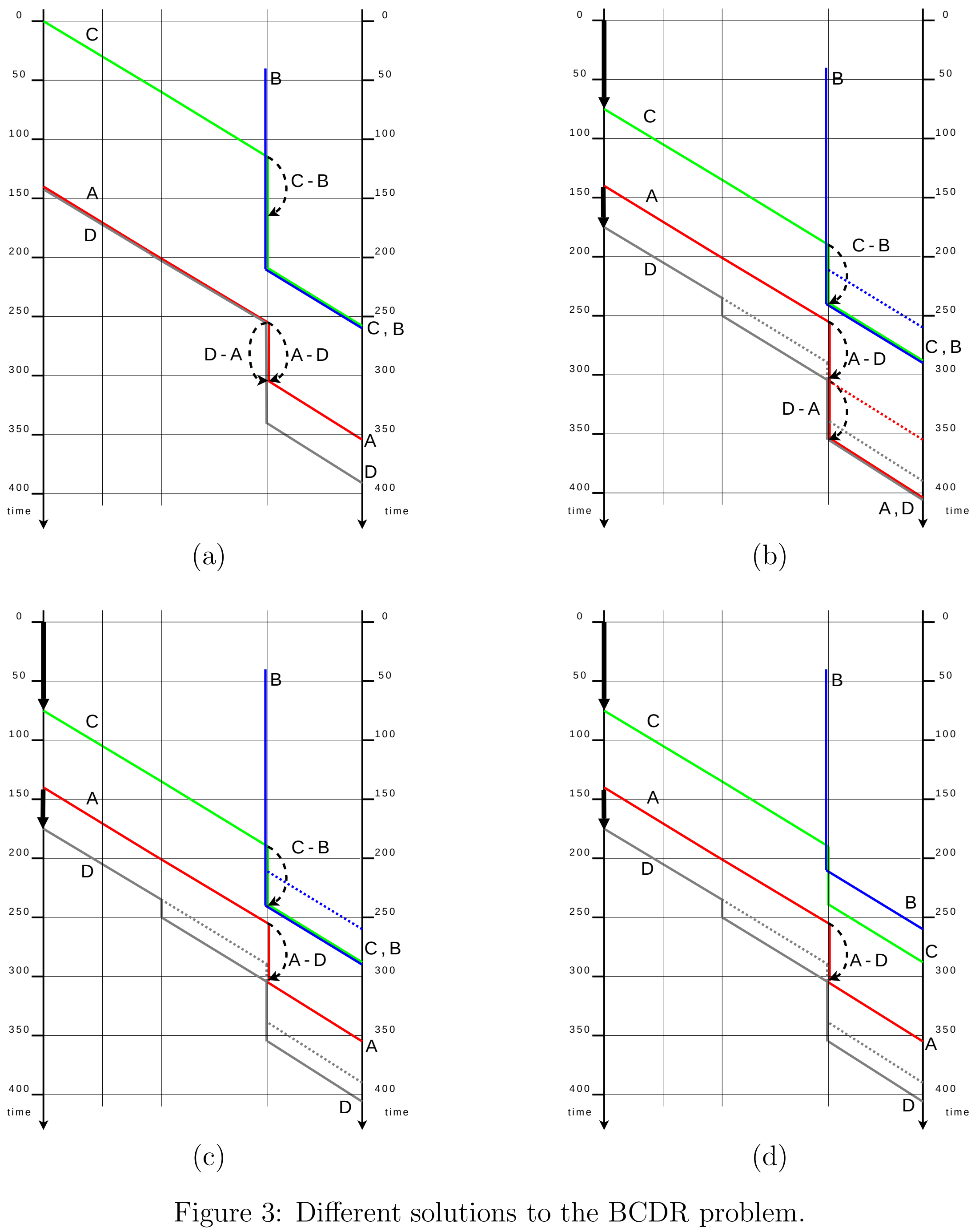


Figure 3: Different solutions to the BCDR problem.

at time 115, leave the station at time 210 and leave the network at time 260. Train D is scheduled to enter the network at time 140, arrive at the station (platform 8) at time 255, leave the station at time 340 and leave the network at time 390.

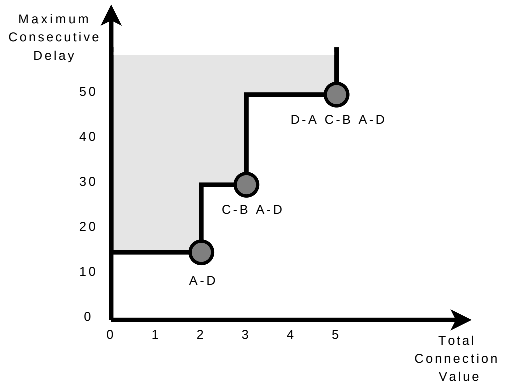
The minimum dwell time of each train at its scheduled stop is 50, even if the scheduled dwell times in the timetable include some time margins to satisfy the three transfer connections and to recover from delays. The connection time is 50 for each transfer connection, while the connection value is 2 for A-D and D-A, 1 for C-B.

Figures 3 (b), (c) and (d) show three solutions for the same entrance perturbation in which train C and train D are delayed at their entrance in the network by 75 and 35 time units respectively (see the thick black arrows).

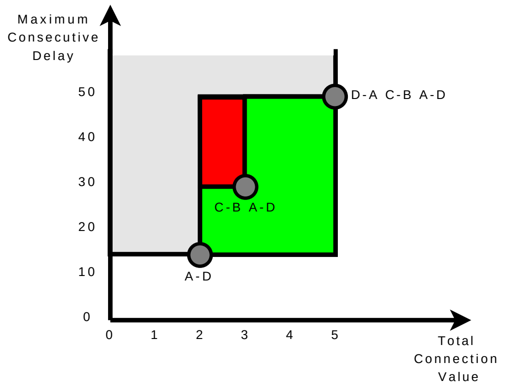
Figure 3 (b) shows the optimal solution for the CDR problem when all the connections are kept. This solution has a connection value equal to 5 and a maximum consecutive delay of 50 time units (due to train A). The other trains face the following consecutive delays: 30 time units for train B and 15 time units for train D, while the initial delay of A, B and D is zero. The consecutive delay of train C is zero since the exit delay 30 is entirely an initial delay.

Since the connections constraints are variables for the BCDR problem, in total there are eight combinations of transfer connections to keep or drop. By computing an optimal solution to the CDR problem in terms of maximum consecutive delay for each combination, there are eight points in the plane of the objective functions of the BCDR problem, three of which are non-dominated and shown in Figure 4.

The solution of Figure 3 (b) is clearly non-dominated. The other two Pareto optimal solutions are shown in Figure 3 (c) and (d). Figure 3 (c) shows the solution in which the connection D-A is not enforced. The consecutive delay of train A is reduced to 0, while the delays of the other trains do not change with respect to the solution of Figure 3 (b). The connection value of this solution is equal to 3. Figure 3 (d) reports the solution in which the connections D-A and C-B are dropped. Train D still faces a consecutive delay of 15 time units and its delay cannot be further reduced by removing other transfer connections or rescheduling trains. The connection value of this solution is equal to 2.



(a)



(b)

Figure 4: Pareto front for the illustrative example.

The Pareto front is shown in Figure 4 (a). Each solution is associated to the point with coordinates equal to the corresponding connection value and maximum consecutive delay. A solution is represented by a circle and labeled with the list of enforced connections (see, e.g., {D-A, C-B, A-D}). The grey area represents the region of the solutions space that is dominated by the three Pareto optimal solutions (i.e., {A-D}, {C-B, A-D} and {D-A, C-B, A-D}).

An interesting performance indicator is the *Pareto front area*, computed as the ratio between the area covered by the non-extreme Pareto optimal solutions (i.e., the area of the small rectangle generated by points (3,30) and (2,50) in Figure 4 (b), equal to 20) and the area defined by the bounding box of the two extreme Pareto optimal solutions (i.e., the area of the large rectangle generated by points (2,15) and (5,50) in Figure 4 (b), equal to 105). In the proposed example, the Pareto front area is 0.19. More in general, if only the two extreme solutions (no connections and all connections enforced) are found, then the Pareto front area is 0. Clearly, the value 1 is the maximum achievable for the Pareto front area, reached when a single non-dominated solution is found.

3.4 Algorithms

In this section we describe two algorithms to compute an approximation of the Pareto front of the BCDR problem, named Add and Remove. Both algorithms use the branch and bound algorithm described in [5] as a building block. This branch and bound algorithm solves the CDR problem with a fixed set of enforced connections and is incorporated within a metaheuristic framework based on the Pareto local search algorithm proposed by Paquete and Stützle [18] for the multi-objective quadratic assignment problem.

Both Add and Remove maintain an archive Z of non-dominated solutions which is returned at the end of the search. In what follows we redefine the set C as the set of all transfer connections to keep or drop, and the set $S^C \subseteq C$ as the set of enforced transfer connections. We let $D(S^C)$ be the maximum consecutive delay associated to an optimal solution to the CDR problem with the set of enforced connections S^C. We also let $V(S^C)$ be the total value of the connections satisfied in this solution. The pair $[V(S^C), D(S^C)]$ is the associated point in the plane of the two objective functions for the BCDR problem.

```
Algorithm Add
Set $S^C = \emptyset$ and compute $D(S^C)$
Initialize archive $Z$ with element $S^C = \emptyset$ and attributes $[0, D(S^C)]$ and visited flag $f(S^C) = 0$
    while there is at least an element in the archive with $f(S^C) = 0$ do
        Select an element $S^C$ with $f(S^C) = 0$ from the archive $Z$
        for all connections $j \in C - S^C$ do
            Generate a neighbor $\hat{S}^C = S^C \cup \{j\}$
            if the set $\hat{S}^C$ is not in $Z$ do
                Compute $V(\hat{S}^C)$ and $D(\hat{S}^C)$
                Append $\hat{S}^C$ to $Z$ with $[V(\hat{S}^C), D(\hat{S}^C)]$ and $f(\hat{S}^C) = 0$
            end if
        end for
        Set $f(S^C) = 1$
        Remove from $Z$ all the dominated elements
    end while
```

Figure 5: Pseudocode of the Add algorithm.

Each solution in the archive is characterized by the set S^C with attributes $[V(S^C), D(S^C)]$ and a visited flag $f(S^C)$ initially set to 0. This flag is used during the search to keep track of the already visited solutions (with $f(S^C) = 1$). Initially, a starting solution is inserted in the archive depending on the chosen algorithm ($S^C = \emptyset$ for the Add algorithm and $S^C = C$ for the Remove algorithm). A neighbor $\hat{S}^C$ is the set obtained by adding to S^C a single connection in $C - S^C$ (algorithm Add) or removing a single connection from S^C (algorithm Remove). The sketch of Algorithm Add is shown in Figure 5. The sketch of Remove is not shown since its structure is quite similar.

Figure 6 shows the behavior of both algorithms by using the illustrative example of Section 3.3. For the sake of clarity in figure we report under $V(S^C)$ the value of the connections in S^C (connections enforced) rather than those satisfied in the associated optimal solution. For the Remove algorithm, the starting solution is the one with all connection constraints enforced, which achieves the highest values for connections and maximum consecutive delay. There are three neighbors of this solution that are reachable by removing a single connection constraint. The associated values $[V(S^C), D(S^C)]$ for the three neighbors are then computed, as shown in Figure 6 (a). Two neighbors are dominated by the initial solution and therefore removed from the archive Z, and the search is repeated from the only non-dominated neighbor with S^C = {C-B, A-D}. The procedure is iterated until all the solutions in Z are visited. Figure 6 (b) reports on the behavior of algorithm Add for the same example. Note that Add algorithm generates a different set of visited solutions compared to Remove algorithm.

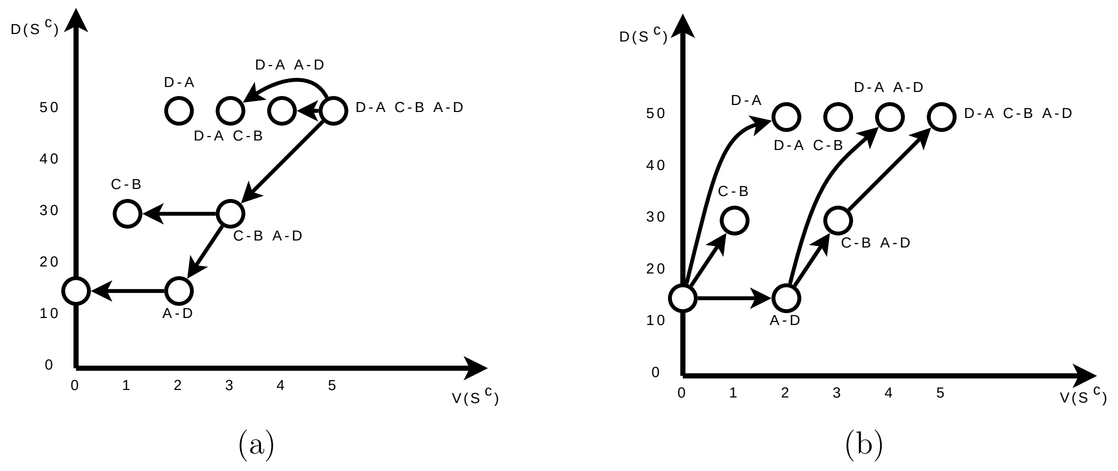


Figure 6: Exploration of the solution space by Remove (a) and Add (b).

We next illustrate a modified version of the Add algorithm based on a larger neighborhood of the current solution. Starting from the current set S^C, the modified Add algorithm extends the neighborhood of S^C as follows. If two or more neighbors, say $S^1 = S^C \cup \{j_1\}, \ldots, S^k = S^C \cup \{j_k\}$, yield the same value $D(S^1) = \ldots = D(S^k)$ of maximum consecutive delay of the current solution, an additional neighbor $\hat{S}^C = S^C \cup \{j_1, \ldots, j_k\}$ is generated. In our computational experiments, it frequently occurs that $D(\hat{S}^C) = D(S^1) = \ldots = D(S^k)$, which implies that $\hat{S}^C$ dominates the previous neighbors. We also observe that a similar modification of the Remove algorithm does not yield improvements in the quality of the Pareto front and increases the computation time of the algorithm. Therefore in the next section we only report on the modified version of Add and on the basic version of Remove.

4 Computational results

In this section, we present a real-world test case of the BCDR problem and investigate the performance of the proposed algorithms. The code is implemented in C++ and runs on an Intel Core Duo 2 GHz workstation, under a Windows platform.

4.1 Test case description

The test instances are based on the railway network around the main station of Utrecht, in the Netherlands. This network, shown in Figure 7, includes approximately the first 10 km of five main lines departing from Utrecht, and it is delimited by the following stations: Utrecht Overvecht on the line to Amersfoort; Driebergen-Zeist on the line to Arnhem and Nijmegen; Culemborg on the line towards Den Bosch, Eindhoven and Maastricht; Vleuten on the line to Rotterdam and The Hague; Maarssen on the line towards Amsterdam and Schiphol airport.

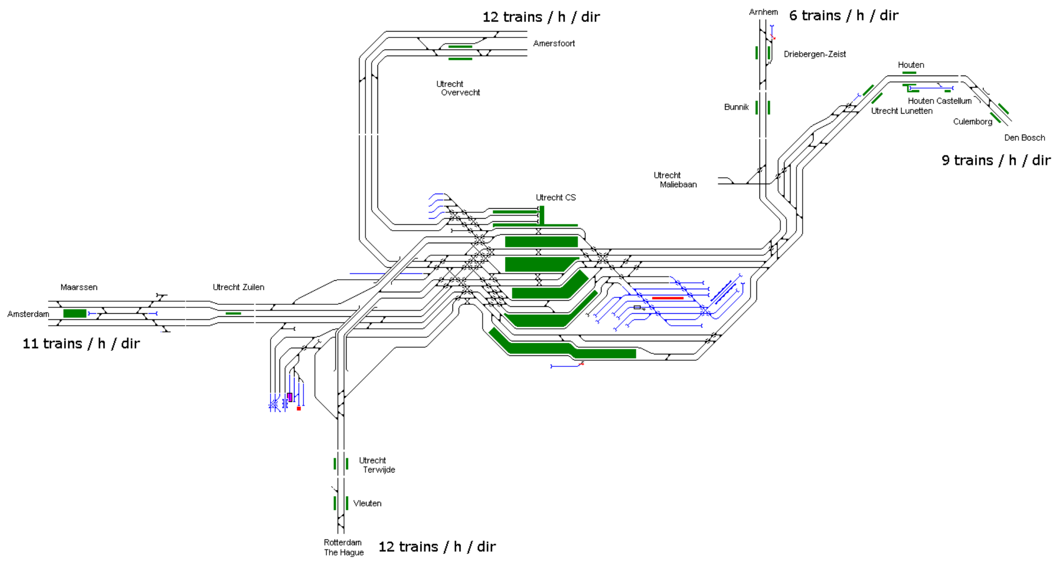


Figure 7: Dispatching area around Utrecht Central Station.

Utrecht Central Station contains 20 platform tracks, most of which might host two trains at a time, used for instance when coupling or splitting train units. Most of the platform tracks are used by through traffic (i.e., trains traversing the station without changing the direction of travel after their stop at a platform), even if some trains change direction and there are three dead-end platforms.

For the computational experiments, we use the 2008 timetable that is cyclic with a cycle length of one hour. The trains are mostly for passenger services, operated by NS (Nederlandse Spoorwegen), except for a few freight trains. The timetable schedules up to 80 trains in a peak hour. At Utrecht Central Station the main lines connecting the North and South of the Netherlands cross the major lines to the West and the East, making the station a major communication hub. The total amount of travelers at Utrecht Central Station is around 150.000 per day, with a large share of transfer passengers.

Transfer connections have been considered in the test experiments, as well as connections due to coupling and splitting of rolling stock for intercity and local services coming from/going to Rotterdam, the Hague or Amersfoort, and re-use of rolling stock for commuter services towards Utrecht Overvecht and Culemborg. Connection value for the different types of connections is set as follows:

- Passenger transfer connection for intercity services, important or less frequent service. The minimum required time is 60 seconds and the value is set to 2.
- Passenger transfer connection for intercity services, less important or frequent service. The minimum required time is 60 seconds and the value is set to 1.
- Rolling stock connections due to coupling or decoupling units. The minimum required time is 120 seconds and the value is set to 5.
- Rolling stock connection due to turn-around units. The minimum required time is 300 seconds and the value is set to 5.

The chosen values take into account the relevance of connected services. Other possible methods for defining connection values would take into account the number of passengers that transfer at each connection, similarly to [8].

We consider timetable perturbations generating initial delays according to the Weibull distribution, which is particularly effective to approximate delays occurring in practice [26]. The definition of the parameters of the Weibull distribution for the different train categories is based on an analysis of more than 33000 train events (arrivals, departures, dwell processes and passing times) recorded at Utrecht Central Station in April 2008 by Dutch infrastructure manager ProRail. However, in this paper we consider larger parameters for intercity and international trains, with respect to the values occurring in practice, in order to generate heavily perturbed scenarios. The parameters are shown in Table 1. These parameters have been used to generate 25 perturbation instances. The resulting perturbations range from an entrance delay equal to 0 to a maximum of 1313 seconds, with an average of 181.6 seconds. About 25% of all trains are delayed at their entrance in the area by more than 5 minutes.

Table 1: Parameters of the Weibull distribution.

Train Category	Scale	Shape	Shift
Intercity	788	2.2	-315
Stoptrein	227	2.4	-198
Sprinter	235	3.0	-186
International	1120	1.4	-205
Freight	1099	2.6	-885

We considered two scenarios with different sets of connections. For both scenarios, each instance contains 79 trains and 451 resources (either block sections or platforms). The resulting alternative graph has the following size: $|N| = 1847$ nodes, $|F| = 2156$ arcs, $|A| = 4773$ pairs of alternative arcs. As for the set $|C|$, the first scenario considers twelve transfer connections for passenger services and seven additional connections due to rolling stock circulation and coupling/decoupling constraints. Rolling stock connections are not relaxable in this scenario. Therefore, in the first scenario $|C| = 12$ and $|F| = 2163$. The second scenario is introduced in order to assess the performance of the different algorithms when increasing the number of connections. In this scenario, we added other five rolling stock connections to the first scenario and considered relaxable all the connections, thus leading to a larger set of $|C| = 24$ relaxable connections. We observe that the first scenario is closer to the practical situation, in which only transfer connections are flexible, while the second scenario considers more challenging instances.

4.2 Effects of keeping or removing all connections

In this section we focus on the effects on delay propagation of keeping/dropping connections. Table 2 compares the best solutions achieved for the 25 instances of Section 4.1 for the two scenarios with 12 and 24 relaxable connections and for the two extreme cases in which all relaxable connections are enforced or dropped. 90 instances have been solved at optimality by the branch and bound algorithm described in [5], while for the remaining 10 instances we keep the best solution found after 20 hours of computation.

Each value in Table 2 is the average over the 25 instances. We report on the maximum consecutive delay, the total value of the relaxable connections satisfied (also when not enforced) and the computation time. Under the first scenario, the value of rolling stock connections is set to zero since these connections cannot be dropped. We observe that, being the timetable quite robust, most of the connections are satisfied. In fact, the time reserve inserted in the timetable is quite large, in particular for the rolling stock connections. For both scenarios, the maximum consecutive delay when all connections are enforced is more than doubled with respect to the other case in which few connections are not satisfied. In other words, keeping all the connections increases significantly the propagation of delays.

Table 2: Gap between the extreme BCDR solutions.

No Connection Enforced		All Connections Enforced	
Max Cons Delay (s)	Connections Value	Max Cons Delay (s)	Connections Value
First Scenario			
208.7	12.8	530.1	16
Second Scenario			
208.7	69.6	583.6	76

4.3 Performance of the algorithms

This section focuses on the bi-objective problem and on the Pareto front generated by the two algorithms described in Section 3.4. We report on the performance of the modified version of Add and on the basic version of Remove. For the first scenario we also compare the performance of the two algorithms with the Pareto front computed by enumerating all possible combinations of enforced connections. For all instances, the CDR problem is solved by the branch and bound algorithm described in [5] truncated after 10 seconds of computation.

Table 3 reports on our results for the 25 instances of Section 4.1 and for the two scenarios with 12 and 24 relaxable connections. Each value in the table is the average over the 25 instances. For each algorithm, we report on the following quantities: the average number of instances of the CDR problem that have to be solved for an instance of the BCDR problem; the average time required to compute the Pareto front; the number of non-dominated solutions; the Pareto front area; the percentage of times the branch and bound code is truncated before proving the optimality of the solution found.

Table 3: Average results of the Pareto front algorithms.

Algorithm	Scheduler Calls	Total Time (s)	# of P. F. Solutions	P. F. Area (%)	Time Limits BB (%)
First Scenario					
Remove	36	283	3.32	26	18
Add	20	166	3.32	26	23
Exhaustive	4096	33504	3.32	26	20
Second Scenario					
Remove	91	705	4.04	28	18
Add	36	309	4.00	27	25

We observe that both algorithms are very effective in generating the Pareto front. For the first scenario, both Add and Remove find the same Pareto front of the exhaustive search within less than 1% of computation time. For the second scenario, the number of non-dominated solutions found by Add algorithm is only 1% less compared to Remove. On the other hand, algorithm Add is quite more efficient than algorithm Remove. In fact, the computation time of Add is only 58% [respectively 44%] than the computation time of Remove for the first [the second] scenario. On the whole, no algorithm outperforms the other.

4.4 Critical transfer connections

This section focuses on the detection of the most critical connections with respect to delay propagation. We limit our analysis to the first scenario, more relevant in practice than the other scenario, and assume that only Pareto optimal solutions should be chosen for conflict resolution purposes. We define the criticality of a connection as the percentage of instances in which a connection is dropped in at least one non-dominated solution. Differently from other studies on timetable robustness (see, e.g., [13]), we investigate the combined effect of dropping different subsets of connections and rescheduling trains on the basis of a detailed optimization model.

Table 4 shows the criticality of the twelve transfer connections of the first scenario. Every connection occurs at Utrecht Central Station. Each row of the table refers to a pair of connected train services. For example, the connection D800-B2000 connects the intercity service D800 coming from Amsterdam to the intercity service B2000 going to Nijmegen. We than show origin and destination of the feeder and waiting trains respectively, connection value and criticality. The latter information might be useful for taking fast decisions in real-time. For example, if a feeder train of a critical connection is heavily delayed, it is likely that the connection should be dropped in order to reduce delay propagation, since the probability of having that connection satisfied in a non-dominated solution is very small. Similarly, information on criticality can be useful for planning purposes, since it highlights those connections that require larger time reserves to increase the timetable resilience to disturbances. For example, in Table 4 the connections A2000-A800, B800-D2000 and C3500-C3000 are the most critical.

Table 4: Criticality of the twelve transfer connections.

Feeder-Waiting	Origin	Destination	Value	Criticality (%)
A2000-A800	Nijmegen	Amsterdam	1	36
B800-D2000	Amsterdam	Nijmegen	1	32
C3500-C3000	Eindhoven	Amsterdam	2	32
C2000-C800	Nijmegen	Amsterdam	1	28
D800-B2000	Amsterdam	Nijmegen	1	28
A3000-A3500	Nijmegen	Schiphol	2	20
A3500-A3000	Eindhoven	Amsterdam	2	20
C800-C2000	Maastricht	The Hague	1	16
C3000-C3500	Nijmegen	Schiphol	2	12
A800-A2000	Maastricht	The Hague	1	4
B2000-D800	The Hague	Maastricht	1	4
D2000-B800	The Hague	Maastricht	1	0

4.5 Decision support system

In this section we discuss the implementation of a user interface to support the dispatching decisional process. We then investigate non-dominated solutions that satisfy additional properties, making them particularly interesting from a theoretical point of view.

Figure 8 shows a possible representation of the Pareto front for a perturbed situation of the first scenario. Here the dispatcher has an aggregated view of feasible solutions from the viewpoint of the two main performance indicators. On the horizontal axis the total value of satisfied connections is reported, while the maximum consecutive delay is reported on the vertical axis. There are five non-dominated points: (10, 247), (12, 250), (13, 328), (14, 471), (16, 534). For each non-dominated solution, the waiting trains that should leave the station without waiting for the arrival of the feeder train are also displayed (i.e., which connections must be dropped for implementing that solution). For example, for the leftmost non-dominated solution (10, 247) trains A3000, C2000, B2000 and C3500 should not wait for their respective feeder trains A3500, C800, D800 and C3000.

This kind of tool would be useful to support the discussion between the infrastructure dispatcher and the managers of the involved train operating companies, who can express preferences on the connections to keep or drop, and therefore on the solution to implement.

We next highlight two well-balanced non-dominated solutions that exhibit some special characteristics according to bargaining theory. In a bargaining problem two players (e.g., a manager A of a train operating company and a dispatcher B) negotiate a common strategy among a set of possible *agreements* (i.e., the non-dominated solutions of the BCDR problem). A *bargaining problem* (P, d) is defined by a set P of the possible agreements and a *disagreement point d*. Each agreement $p \in P$ is characterized by the payoffs for players A and B, denoted by $u_A(p)$ and $u_B(p)$. If the negotiation fails, A and

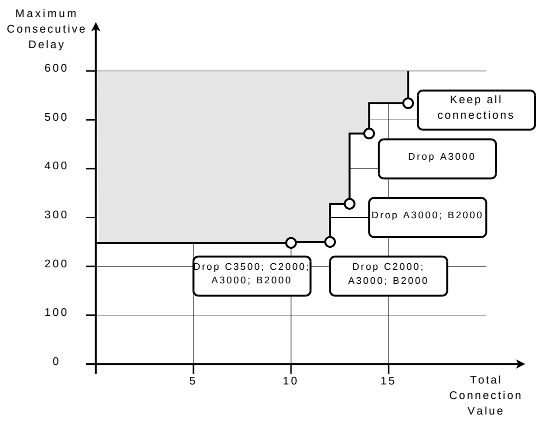


Figure 8: BCDR solutions on the Pareto front.

B are forced to agree on the disagreement point $d = (d_A, d_B)$, which can be seen as the worst possible solution. The example in Figure 8 can be viewed as a bargaining problem with five possible agreements characterized by the payoffs (10, 247), (12, 250), (13, 328), (14, 471), (16, 534). We note that while the first element in each pair is a payoff, the second is actually a cost to be minimized. A common definition of the disagreement point d consists of setting for each coordinate the worst value among the possible agreements [1], i.e., $d = (10, 534)$ in the example.

A *solution* of a bargaining problem is a subset of agreements (possibly, a single agreement) $\varphi(P, d) \subseteq P$. The *Nash bargaining solution* $\varphi_N(P, d)$ is given by [17]:

$$\varphi_N(P, d) = \arg \max_{(u_A, u_B) \in P} \left[(u_A - d_A)(u_B - d_B)\right]. \tag{2}$$

Nash [17] proves that, for a negotiate over a continuous convex bargaining set, $\varphi_N(P, d)$ is the unique solution which satisfies four axioms of (1) (Weak) Efficiency, (2) Symmetry, (3) Scale Covariance and (4) Independence of Irrelevant Alternatives. Mariotti [14] demonstrates that $\varphi_N(P, d)$ satisfies analogous axioms for a discrete bargaining set. In the example in Figure 8, $\varphi_N(P, d) = (13, 328)$.

The Kalai-Smorodinsky [12] solution $\varphi_{KS}(P, d)$ requires the definition of an additional point, called the *utopia point* $b = (b_A, b_B)$, whose coordinates are the best achievable for each objective function, i.e., $b = (16, 247)$ for the example. When the agreement set P is discrete, Nagahisa and Tanaka [16] prove that, for a negotiate over a discrete bargaining set, $\varphi_K S(P, d)$ satisfies five axioms of (1) Continuity, (2) Independence, (3) Symmetry, (4) Invariance and (5) Monotonicity. Moreover, $\varphi_K S(P, d)$ minimizes the distance from the segment connecting the utopia point with the disagreement point [16]. In the example in Figure 8 $\varphi_{KS}(P, d) = (13, 328)$.

5 Conclusions and further research

This paper presents a bi-objective conflict detection and resolution problem in which the decision maker aims at minimization of train delays and maximization of the total value of satisfied connections. To support the decision process, which typically involves infrastructure dispatchers and train operating companies, we propose the computation of the Pareto front of non-dominated solutions among which to select a compromise solution. To this aim, two heuristic algorithms have been developed. Computational experiments, carried out on real-world data provided by Dutch railways, show that both algorithms are very effective in approximating the Pareto front within a limited computational time and that keeping or dropping even a limited set of connections may impact dramatically on delay propagation. This result demonstrates that finding a compromise solution between delay minimization and connections satisfaction deserves a high potential for advanced performance management.

Future research should further expand the needs of different stakeholders (infrastructure company, train or bus operating companies, passengers, and so on) by taking into account a larger variety of dispatching possibilities (e.g., train rerouting and cancellation of train services) and performance indicators (e.g., energy consumption, balanced use of infrastructure, minimization of passenger delays, schedule robustness). Addressing such needs requires taking into account a larger number of decision alternatives within the strict time limits imposed by real-time applications. To this aim, more efficient algorithms are necessary, able to quickly generate satisfactory solutions. Interactive algorithms might also be an effective approach to quickly drive the search towards good solutions.

References

[1] Agnetis, A., De Pascale, G., Pranzo, M. (2009). Computing the Nash solution for scheduling bargaining problems. *International Journal of Operational Research* **6** (1), 54–69.

[2] Albrecht, T., Oettich, S. (2002). A new integrated approach to dynamic schedule synchronisation and energy-saving train control. In: Allan, J., Brebbia, C.A., Hill, R.J., Sciutto, G., Sone, S. (Eds.), Computers in Railways VIII, 847-856, WIT Press, Southampton, UK.

[3] Caprara, A., Kroon, L.G., Monaci, M., Peeters, L.W.P., Toth, P. (2006). Passenger railway optimization. In: Barnhart, C., Laporte, G. (Eds.), Handbooks in Operations Research and Management Science, **14**, 129-187.

[4] D'Ariano, A. (2008). *Improving real-time train dispatching: Models, algorithms and applications.* TRAIL Thesis Series, T2008/6, The Netherlands.

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

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

[7] Demeulemeester, E., Herroelen, W., Elmaghraby, S. (1996). Optimal procedures for the discrete time/cost trade-off problem in project networks. *European Journal of Operational Research* **88** (1), 5068.

[8] Ginkel, A., Schöbel, A. (2007). To wait or not to wait? The bicriteria delay management problem in public transportation. *Transportation Science* **41** (4), 527-538.

[9] Goverde, R.M.P. (1998). Synchronization Control of Scheduled Train Services to Minimize Passenger Waiting Times. In: Bovy, P.H.L. (Ed.), Proceedings of the 4th TRAIL Congress, Delft University Press, Delft, The Netherlands.

[10] Hansen, I.A. and Pachl, J. (2008). *Railway Timetable and Traffic: Analysis, Modelling and Simulation.* Eurailpress, Hamburg, Germany.

[11] Heilporn, G., De Giovanni, L., Labbé, M. (2008). Optimization models for the single delay management problem in public transportation. *European Journal of Operational Research* **189** (3), 762–774.

[12] Kalai, E., Smorodinsky, M. (1975). Other Solutions to Nash's Bargaining Problem. *Econometrica*, **43** (3), 513–518.

[13] Liebchen, C., Schachtebeck, M., Schöbel, A., Stiller, S., Prigge, A. (2009). Computing delay resistant railway timetables. *Computers and Operations Research* to appear.

[14] Mariotti, M. (1998). Nash bargaining theory when the number of alternatives can be finite. *Social Choice and Welfare*, **15** (3), 413–421.

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

[16] Nagahisa, R.I., Tanaka, M. (2002). An axiomatization of the Kalai-Smorodinsky solution when the feasible sets can be finite. *Social Choice and Welfare*, **19** (4), 751–761.

[17] Nash, J.F. (1950). The Bargaining Problem. *Econometrica*, **18**, 155–162.

[18] Paquete L., Stützle T. (2006). A study of stochastic local search algorithms for the biobjective QAP with correlated flow matrices, *European Journal of Operational Research* **169** (3), 943–959.

[19] Peeters, L.W.P. (2003). *Cyclic Railway Timetable Optimization.* TRAIL Thesis Series T2003/5, The Netherlands.

[20] Schöbel, A. (2001). A model for the delay management problem based on mixed-integer programming. Proceedings of the 1st Workshop on Algorithmic Methods and Models for Optimization of Railways, Crete, Greece. Electronic Notes in Theoretical Computer Science **50** (1), 1-10.

[21] Scholl, S. (2005). *Customer-Oriented Line Planning.* PhD thesis, University of Kaiserslautern, Kaiserslautern, Germany.

[22] Vansteenwegen, P., Van Oudheusden, D. (2007). Decreasing the passenger waiting time for an intercity rail network. *Transportation Research Part B* **41** (4), 478-492.

[23] Vromans, M.J.C.M. (2005). *Reliability of Railway Systems.* TRAIL Thesis Series T2005/7, The Netherlands.

[24] Wegele, S., Schnieder, E. (2005). Dispatching of train operations using genetic algorithms. In: Hansen, I.A., Dekking, F.M., Goverde, R.M.P., Heidergott, B., Meester, L.E., (Eds.), Proceedings of the 1st International Seminar on Railway Operations Modelling and Analysis, Delft, The Netherlands.

[25] Wong, R.C.W., Yuen, T.W.Y., Fung, K.W., Leung, J.M.Y. (2008). Optimizing timetable synchronization for rail mass transit. *Transportation Science* **42** (1), 57-69.

[26] Yuan, J. (2006). *Stochastic Modelling of Train Delays and Delay Propagation in Stations*. TRAIL Thesis Series, T2006/6, The Netherlands.