Disruptions in the airline industry: math-heuristics for re-assigning aircraft and passengers simultaneously

Raïd Mansi

[1] Univ Lille Nord de France, F-59000 Lille, France
[2] CNRS, FRE 3304, F-59313 Valenciennes, France
[3] UVHC, LAMIH, F-59313 Valenciennes, France
Le Mont Houy, ISVT2
59313 Valenciennes Cedex 9, France
E-mail: raid.mansi@univ-valenciennes.fr

Saïd Hanafi

[1]Univ Lille Nord de France, F-59000 Lille, France
[2] CNRS, FRE 3304, F-59313 Valenciennes, France
[3] UVHC, LAMIH, F-59313 Valenciennes, France
Le Mont Houy, ISVT2
59313 Valenciennes Cedex 9, France
E-mail: said.hanafi@univ-valenciennes.fr

Christophe Wilbaut*

[1] Univ Lille Nord de France, F-59000 Lille, France
[2] CNRS, FRE 3304, F-59313 Valenciennes, France
[3] UVHC, LAMIH, F-59313 Valenciennes, France
Le Mont Houy, ISVT2
59313 Valenciennes Cedex 9, France
Fax: +33327511940
E-mail: christophe.wilbaut@univ-valenciennes.fr
*Corresponding author

François Clautiaux

[1] Univ Lille Nord de France, F-59000 Lille, France
[2] CNRS, UMR 8022, F-59000 Lille, France
[3] USTL, LIFL, F-59000 Lille, France
Université des Sciences et Technologies de Lille, IUT A de Lille 1
59655 Villeneuve d'Ascq Cedex
E-mail: francois.clautiaux@univ-lille1.fr

Abstract: In this paper, we propose an oscillation strategy heuristic combined with mathematical programming for Disruption Management

in the Airline Industry (DMAI). The goal of this problem is to resume normal operations as quickly as possible during the recovery period while minimizing the resulting costs and the potential impacts to passengers. In cases of disruptions, DMAI aims to reassign aircraft and passengers simultaneously rather than according to the natural hierarchy of aircraft, crews and passengers. In this problem, we consider many types of practical disruptions, such as mechanical failures, personnel strikes or inclement weather. Just finding a feasible flight schedule is a hard problem. Our method can be divided into two main stages: in the first stage, we try to generate a feasible solution to the problem; in the second, we improve this solution using an oscillation strategy that alternates between constructive and destructive phases. Our numerical results show the effectiveness of this method, which produced the best results known for some of the most demanding instances of a real-life problem. With these results, we ranked 2nd in an international challenge.

Keywords: Airline Industry; Mathematical Programming; Metaheuristic; Matheuristic; Oscillation Scheme.

Biographical notes: Raïd Mansi has a post-doc position in the University of Minho, Portugal. He got his PhD degree in Computer Science in the University of Valenciennes, in 2009. His main research field is the hybridization of heuristics and mathematical programming to solve variants of knapsack and bilevel problems.

Saïd Hanafi holds a Full Professor position in Computing Science at Institute of Techniques and Sciences, University of Valenciennes. His research lies in the design of effective heuristic and metaheuristic algorithms for solving large-scale combinatorial search problems. His is interested in theoretical as well as algorithmic modeling and application aspects of integer programming and combinatorial optimization and has published over 30 articles on the topic. His current interests revolve around the integration of tools from hybrid methods mixing exact and heuristics for solving hard problems.

Christophe Wilbaut is an Associate Professor in the "Institut des Sciences et Techniques de Valenciennes", Valenciennes University, France. He received a PhD in Computer Sciences at the University of Valenciennes in 2006. His teaching and research interest include algorithms and combinatorial optimization. His current researches include the development of efficient metaheuristics and hybrid methods for solving hard optimization problems.

François Clautiaux is an Associate Professor in the "Université de Lille 1", France. He received a PhD in Computer Sciences at the "Université de Technologie de Compiègne" in 2005. His teaching and research interest include algorithms, graphs, and combinatorial optimization. His current researches deal with hybrid methods based on mathematical programming, metaheuristics and graph theory.

1 Introduction

When a commercial airline determines a flight schedule, the aim is generally to optimize its revenue. However, external events (e.g., mechanical failures, personnel strikes or inclement weather) commonly occur, which disrupts the airline's operations. In such cases, effective solutions must be found to minimize the duration and the impact of the disruption. The search for effective solutions to this problem led to the development of Disruption Management in the Airline Industry (**DMAI**). In general, resources are re-allocated sequentially, according to a natural hierarchy: aircraft, crew and passengers. In our research, we aimed to integrate global decision-making into the DMAI problem in order to re-assign aircraft and passengers simultaneously. To do this, we took passenger inconvenience into account as a function of the total delay compared to the original itinerary, with an indicator expressed in monetary units. Thus, the objective function of our DMAI problem is a weighted sum of the factors associated with additional costs and/or gains due to modifying the flight schedule and the indicator of the passenger inconvenience. Amadeus[1] proposed this subject for the 6th challenge of the French Society of Operations Research and Decision Analysis (ROADEF).

Airline problems can be divided into two main categories: yield management and airline scheduling. Yield management concerns seat allocation and booking policies, and is not within the scope of this article. Airline scheduling covers every aspect of constructing and executing flight timetables. This general category can be divided into several sub-categories that depend on the instant when the problem occurs. These sub-categories are divided into three different levels: the *strategic* level, the *tactical* level and the *operational* level. In this paper, we focus more particularly on the operational level.

There are many papers in the literature about problems that are close to our DMAI problem. In most cases, disruption problems concern only one kind of disturbance. At the operational level, disturbances such as flight delays and cancellations, crew absences and/or closed airports are generally taken into account. Several important tasks are dealt with at this level: flight and passenger rerouting and crew scheduling. Several papers describe models for aircraft reassignment under disruptions (see for instance Teodorovic and Guberinic (1984) or Jarrah et al. (1993)). Teodorovic and Stojkovic (1990) proposed a lexicographical model meant to minimize the number of cancellations first and, at the level of minimum cancellations, minimize total passenger delays. Teodorovic and Stojkovic (1995) also proposed a general process to solve disruption issues by taking into account crew constraints and the maintenance operations performed on aircraft.

Maintenance constraints require a large amout of resources. They represent hard constraints in the disruption management as we show in the following. Some researchers consider more than one type of maintenance checks simultaneously (Sriram and Haghani (2003)).

More recently, Andersson (2006) presented two metaheuristic methods for solving the flight perturbation problem. Specifically, he showed that a tabu search method can produce efficient solutions in a reasonable CPU time. A quadratic 0-1 programming model simultaneously dealing with flight delays and cancellations during irregular operations was also given by Cao and Kanafani (1997).

Even if crew scheduling is not considered in this work, this problem is crucial in airline industry and under disruptions. Lettovsky et al. (2000) proposed an integer programming model for the crew re-scheduling problem, and used a primal-dual subproblem simplex method to solve its LP-relaxations. Yan and Lin (1997) focused on a critical issue for the airline industry: airport closings. The models that they proposed take the possibility of swapping, delaying or cancelling flights into consideration, which means that all flights landing at or taking-off from the closed airports have to be delayed or cancelled.

Some other studies in the literature deal with several disruptions at the same time. For instance, Bratu and Barnhart (2006) considered aircraft, crews and passengers simultaneously, determining which flight leg departures to postpone and which to cancel to minimize both airline operating costs and estimated passenger delays and disruption costs. Finally, Clausen et al. (2009) has very recently provided a review of the state of the art in airline disruption resource management, dealing with aircraft, crews, passengers and integrated recovery. The reader can refer to Yu (1998) or Yu and Qi (2004) for more references about disruption management.

In our problem there can be many causes of disruption (e.g., cancelled flights, modifications of the number of possible landings/take-offs at some airports), many recovery strategies (e.g., cancelling flights, splitting groups of passengers into sub-groups and rescheduling these sub-groups on different alternative flights, creating new flights) and many practical constraints related to the aircraft (e.g., maximum distances, maintenance), the passengers (e.g., maximum acceptable delay) and the airports (e.g., capacity). Just finding a feasible solution is already a difficult problem, even more so since the recovery period is limited in time. These problem characteristics justify the design of an *ad hoc* approach.

Our experiments demonstrated that the problem is too large to be handled by a single Mixed Integer Program (**MIP**). However, integer programming techniques are able to effectively incorporate the large quantity of the different costs and operational constraints of the DMAI problem. Research on mathematical programming has led to a state of the art where MIP solvers can be effective in a heuristic context, both as primary solvers or as subprocedures. These techniques are now known as "math-heuristics" or "matheuristics" in the literature (see Maniezzo et al. (2009) for example). In this paper we exploit mathematical programming techniques in a metaheuristic framework for a real-world transportation problem.

Our method can be divided into two main stages: in the first, a feasible solution to the problem is generated; in the second, this solution is improved by alternating between constructive and destructive phases in an oscillation strategy. As shown in the following sections, our method combines mathematical programming with heuristics in both stages, which helped us rank second among the 29 teams registered initially in the ROADEF Challenge[2].

This article is organized as follows. In Section 2, we describe the DMAI problem we consider precisely. Section 3 describes the MIP used to obtain an initial feasible flight schedule to the problem. Section 4 is devoted to the main components of the oscillation scheme and shows how the initial feasible solution can be improved using this scheme. Section 5 summarizes our computational results. We offer our conclusions in Section 6.

2 Description of the problem

The goal of our DMAI problem is to resume normal operations of the original flight schedule as quickly as possible after a disruption. Three main actors in DMAI are the fleet of *aircraft*, the *airports* and the *passengers*.

2.1 Aircraft

A fleet of aircraft is a set A of aircraft a operated by an airline. Each aircraft a is defined by a set of characteristics, including a unique identification number, a model (e.g., Boeing 747, A320) and a cabin configuration (i.e., the passenger capacity for each cabin class: first, business, economy). Operational characteristics are common to all aircraft in a given model: turn-round time, transit time and idle time. T_a denotes the maximum value between the turn-round time and the transit time, which corresponds to the minimum idle time between two consecutive flights.

An aircraft may be unavailable for an extended period of time when it needs maintenance. M_a denotes the maintenance date for aircraft a when such a date exists; otherwise, it is set to $+\infty$. Maintenance requires large quantities of resources, both workforce and equipment, and has to be performed at a specific airport, which means that aircraft a must be at the specific airport at time M_a (please note that we do not deal with the management of the resources associated to the maintenance in this work). An aircraft cannot fly more than a given maximum number of hours between two consecutive maintenance operations. The maintenance constraints are *hard* constraints in our DMAI problem, and a solution has to respect these constraints to be feasible. $A^m \subseteq A$ denotes the subset of aircraft that must undergo maintenance.

In the original schedule, aircraft assure a rotation (i.e., an ordered list of sequential flights). $T^-_{i,a}$ (respectively $T^+_{i,a}$) denotes the starting (resp. ending) date of flight i operated by aircraft a (with the original delay), and i^m_a is the last scheduled flight of aircraft a before it undergoes maintenance. In the rest of this paper, flight i denotes the i^{th} flight of an aircraft in its original rotation. Flight $i-$ (resp. flight $i+$) is used to refer to the predecessor (resp. successor) of flight i.

2.2 Airports

Let K be the set of airports k. $D_{k,k'}$ denotes the travelling time from airport k to airport k' (expressed as a constant in the DMAI problem) and $D_{k,k'} = D_{k',k}$. In our problem, airports are characterized by the maximum departure and arrival rates (i.e., the hourly arrival and hourly departure capacities for each given airport). The number of departures (or arrivals) is counted only for discrete 60-minute periods, not for sliding periods. Examples of such periods would be [7:00–8:00[or [8:00–9:00[. $T^-_{k,h}$ (resp. $T^+_{k,h}$) denotes the beginning (resp. the end) of time period h at airport k, and H_k denotes the set of time periods at airport k. In the same way, $C^-_{k,h}$ (resp. $C^+_{k,h}$) denotes the maximum number of hourly departures (resp. arrivals) at airport k during time period h. $k^-(i,a)$ (resp. $k^+(i,a)$) denotes the departure airport (resp. arrival airport) of the flight i executed by aircraft a. For simplicity, some real constraints are not taken into account in our problem (e.g., the maximum number of aircraft on the airport surface).

2.3 Passengers

Let P be the set of group of passengers p. A group of passengers p makes reservations on the flights offered by the airlines. G_p stands for the group's cardinality. A reservation is defined by the number of passengers for whom the reservation was made, the average price paid per passenger, and the description of an itinerary V_p (i.e., an ordered list of consecutive flights, where each flight is represented by a couple $(flight, aircraft)$). All passengers in the group have the same original itinerary, and there is a minimum transit time of 30 minutes between two consecutive flights in an itinerary.

2.4 Disruptions and decision-making

Our DMAI problem deals with the following possible disturbances: (i) flight delays, caused by boarding procedures, longer-than-usual previous flights, or waiting for connecting crew or passengers; (ii) flight cancellations; (iii) the unexpected unavailability of an aircraft for a given period; and/or (iv) reductions of the airport departure and arrival capacities, due to inclement weather conditions, for instance. Possible decisions concerning the flights planned for the original schedule include intentional cancellations and/or delays, aircraft changes (within a specific set of equivalent planes) and the addition of any newly created flights. Possible decisions concerning passengers include cancellations if it is impossible to assure the itinerary within a maximum delay (according to the type of the itinerary), the split of groups of passengers into sub-groups and the reschedule of these sub-groups on different alternative flights. The period of time in which it is possible to step in to recover from the disturbance is called the *recovery period*. T^- (resp. T^+) denotes the beginning (resp. the end) of this period. A set of flights may already be scheduled before this period begins (flights i such that $T^-_{i,a} < T^-$), making it impossible to modify (or cancel) these flights.

The primary objective of DMAI is to minimize airline costs. However, the company may also wish to incorporate into the problem an indicator of the potential passenger impact of the various recovery strategies, in terms of delay or cancellation costs, but also in terms of passenger inconvenience. The costs used in the objective function can be divided into three categories: (i) operating costs, (ii) costs related to passenger inconvenience, and (iii) penalties. The operating costs in the DMAI problem depend on the aircraft model and are expressed in terms of hours of flight time. Some operating costs are also connected to the effect of delays and cancellations on the passengers, according to the delay and the duration of the initial trip. The costs related to passenger inconvenience are expressed in monetary units and depend on the impact of trip cancellation or the total delay compared to the original itinerary. Additional penalties are assessed in cases of non-compliant aircraft locations at the end of the recovery period. These penalties are connected to the constraints that are added to the problem in order to enforce the number of aircraft of each model and configuration at each airport at the end of the recovery period, with the aim of encouraging a swift return to normal operations.

In this paper, we propose a hybrid method that uses MIP and metaheuristics with an oscillation strategy for DMAI. This method is described in the following two sections.

3 Generating an initial feasible flight schedule

We propose a 2-stage DMAI method that will allow normal airport operations to be restored as quickly as possible after a set of disruptions. In the first stage of our method, once the disruptions have appeared, we try to obtain a feasible solution that will minimize the total delay (and cancellation) while maximizing the total number of passengers transferred. Without any maintenance constraints, a valid solution could be obtained by cancelling all flights. Obviously, this solution would not be advantageous from the perspective of objective value. In addition, this cannot be done since some planes would not reach their maintenance airport on time. When both the number of aircraft scheduled for maintenance and the number of disrupted airports are large, obtaining a feasible solution is a hard problem. To obtain a feasible solution, first we use mixed integer programming. If the MIP does not produce a feasible solution, a repair heuristic must be used to make the solution feasible. The second stage employs an oscillation strategy to improve the initial feasible solution.

Due to its difficulty, it is not possible to solve the DMAI problem optimally within a reasonable time (e.g., 10 minutes in the challenge context). Preliminary experiments showed us that it is not even possible to load a linear relaxation of the model in a commercial software, such as CPLEX of Ilog, with the memory limitations imposed in the challenge (2 GB of RAM). Starting with the formulation of another DMAI relaxation, our experiments showed us that the constraints concerning aircraft maintenance and airport capacity are the most difficult to deal with. For this reason, we propose an iterative scheme that first focuses on the maintenance problems, then eliminates the airport capacity problems and finally maximizes the total number of passengers transferred.

3.1 Dealing with the maintenance constraints

To deal with the maintenance constraints on the aircraft, we modeled a relaxation of our problem. This model takes into account the other problem constraints related to the airports and the passengers.

The aircraft constraints aim to insure flight continuity and the respect of the maintenance dates. To insure this continuity, we use a flow model. When disruptions occur, we modify the initial aircraft route by removing certain flight *circuits*. A circuit is a sequence of consecutive flights for a given aircraft, in which the departure airport for the first flight is the same as the arrival airport for the last flight. In order to keep the flow constraints valid, we introduce the notion of *fictive* flight, which corresponds to a cancelled circuit. During a fictive flight, the aircraft remains at its current airport. We also allow the change of the route of an aircraft that is scheduled for maintenance by creating a direct flight (i.e., a "jump") to the maintenance airport, which gets the aircraft to the appropriate airport on time. As a result of these modifications, the new rotation of a given aircraft a is composed of a series of three types of consecutive flights:

- existing flights (U_a), which are already scheduled before the disruption (for all aircrafts),
- fictive flights (V_a) (for aircrafts flying circuits on their initial schedule), and

- created flights (W_a) (only for aircrafts scheduled for maintenance).

Of necessity, since the alternative aircraft rotations are obtained by removing or replacing a flight sequence in the original schedule by a single flight, we have $max(|V_a|, |W_a|) \leq |U_a|$. We also introduce $F_a = U_a \cup V_a \cup W_a$.

Our model also takes into account the other problem constraints. The airport constraints must respect the maximum departure and arrival rates for each interval, and the passenger constraints must insure that the transit times between flights in the same itinerary are respected. These passenger constraints do not have any impact on solution feasibility since it is possible to cancel a large number of itineraries without making the solution infeasible. However, cancelling an itinerary for a group of passengers has an important impact on solution cost.
In our mathematical model, we use the following sets of variables:

- Variables associated with the aircraft constraints - $u_{i,a}$ a binary variable fixed at 1 if aircraft a executes an existing flight i; otherwise, 0; $v_{i,a}$ a binary variable fixed at 1 if aircraft a executes a fictive flight i; otherwise, 0; $w_{i,a}$ a binary variable fixed at 1 if aircraft a executes a created flight i; otherwise, 0; z_a a binary variable fixed at 1 if aircraft a arrives at its scheduled maintenance on time; otherwise, 0; $r_{i,a}$ the delay associated with flight i potentially executed by aircraft a (continuous variable); $t_{i,a}$ the starting time associated with the created flight i potentially executed by aircraft a.

- Variables associated with the airports - $x^{-}_{i,a,k,h}$ a binary variable fixed at 1 if the departure of aircraft a, flight i, is planned in time interval h at airport k; otherwise, 0; $x^{+}_{i,a,k,h}$ a binary variable fixed at 1 if the arrival of aircraft a, flight i, is planned in time interval h at airport k; otherwise, 0.

- Variables associated with the passengers - y_p a binary variable fixed at 1 if and only if the group of passengers p arrives at its final destination; otherwise, 0.

In Figures 1 to 3, we illustrate the different flights for a given aircraft a. In these figures, the value k_j (j=1,...,4) represents an airport, and the value T_h represents the h^{th} period at this airport. For simplicity, in this example, we assume that all the airports have the same time periods. We also assume that aircraft a is scheduled for maintenance at airport k_1 between time periods T_5 and T_6.

Figure 1 Existing flights in the network

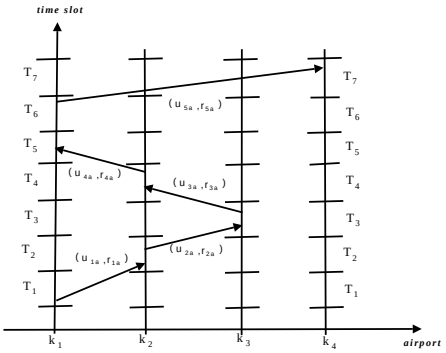


Figure 2 Fictive flights in the network

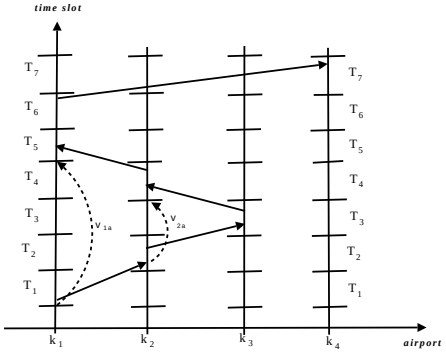
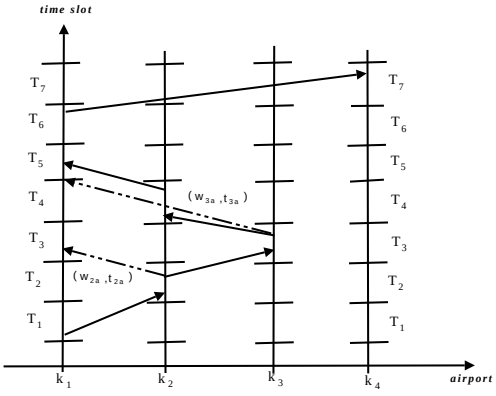


Figure 1 presents the 5 existing flights in U_a. These flights are identified by the variables u and r. Figure 2 presents the 2 fictive flights possible in the network (set V_a). These flights are identified by the variable v (please note that the fictive flight corresponding to the circuit just before flight i of aircraft a is denoted by $v_{\hat{i},a}$ in the rest of the paper). Figure 3 presents the 2 flights in set W_a. These flights are identified by the variables w and t and have the objective of insuring an aircraft's on-time arrival for its scheduled maintenance.

Figure 3 Created flights in the network



To illustrate the notation introduced above, we propose the following example: consider flight $i = 4$ then variable $v_{\hat{i},a}$ corresponds to variable v_{2a}. Since it is not necessary to create a flight for $i = 1$, the first index in the set W_a is 2. The variable v_{3a} does not exist in V_a because the corresponding fictive flight cannot be considered since there are no circuits. These two rules are valid for the sets W_a and V_a. The flight i_a^m in this example is equal to 4.

The problem (P) associated to the first stage can be formulated as follows:

$$\text{(P)} \quad \text{maximize} \quad \sum_{a \in A^m} z_a \tag{1}$$

$$\text{subject to } u_{i-,a} + v_{i,a} \geq u_{i,a} + v_{i,a} + w_{i,a} \quad \forall a \in A, \forall i \in F_a \setminus \{i_a^m\} \tag{2}$$

$$u_{i_a^m,a} + v_{i_a^m,a} + \sum_{j=1}^{i_a^m} w_{j,a} = z_a \quad \forall a \in A^m \tag{3}$$

$$r_{j,a} + T_{j,a}^+ + T_a - L(1-u_{j,a}) \leq r_{i,a} + T_{i,a}^- + L(1-u_{i,a}) \quad \forall a \in A, \forall j < i \tag{4}$$

$$r_{i-,a} + T_{i-,a}^+ + T_a - L(1-u_{i-,a}) \leq t_{i,a} + L(1-w_{i,a}) \quad \forall a \in A^m, \forall i < i_a^m \tag{5}$$

$$r_{i,a} \leq M_a - T_{i,a}^+ \quad \forall a \in A^m, \forall i \leq i_a^m \tag{6}$$

$$t_{i,a} + D_{k^-(i,a),k^+(i_a^m,a)} \leq M_a \quad \forall a \in A^m, \forall i \leq i_a^m \tag{7}$$

$$\sum_{h=1}^{H_{k^-(i,a)}} T_{k^-(i,a),h}^- \times x_{i,a,k^-(i,a),h}^- \leq r_{i,a} + T_{i,a}^- \quad \forall a \in A, i \in U_a \tag{8}$$

$$r_{i,a} + T_{i,a}^- \leq \sum_{h=1}^{H_{k^-(i,a)}} T_{k^-(i,a),h}^+ \times x_{i,a,k^-(i,a),h}^- \quad \forall a \in A, i \in U_a \tag{9}$$

$$u_{i,a} = \sum_{h=1}^{H_{k^-(i,a)}} x_{i,a,k^-(i,a),h}^- \quad \forall a \in A, i \in U_a \tag{10}$$

$$\sum_{h=1}^{H_{k^+(i,a)}} T_{k^+(i,a),h}^- \times x_{i,a,k^+(i,a),h}^+ \leq r_{i,a} + T_{i,a}^+ \quad \forall a \in A, i \in U_a \tag{11}$$

$$r_{i,a} + T_{i,a}^+ \leq \sum_{h=1}^{H_{k^+(i,a)}} T_{k^+(i,a),h}^+ \times x_{i,a,k^-(i,a),h}^+ \quad \forall a \in A, i \in U_a \tag{12}$$

$$u_{i,a} = \sum_{h=1}^{H_{k^+(i,a)}} x_{i,a,k^+(i,a),h}^+ \quad \forall a \in A, i \in U_a \tag{13}$$

$$\sum_{h=1}^{H_{k^-(i,a)}} T_{k^-(i,a),h}^- \times x_{i,a,k^-(i,a),h}^- \leq t_{i,a} \quad \forall a \in A, i \in W_a \tag{14}$$

$$t_{i,a} \leq \sum_{h=1}^{H_{k^-(i,a)}} T_{k^-(i,a),h}^+ \times x_{i,a,k^-(i,a),h}^- \quad \forall a \in A, i \in W_a \tag{15}$$

$$w_{i,a} = \sum_{h=1}^{H_{k^-(i,a)}} x_{i,a,k^-(i,a),h}^- \quad \forall a \in A, i \in W_a \tag{16}$$

$$\sum_{h=1}^{H_{k^+(i,a)}} T_{k^+(i,a),h}^- \times x_{i,a,k^+(i,a),h}^+ \leq t_{i,a} + D_{k^-(i,a),k^+(i_a^m,a)} \quad \forall a \in A, i \in W_a \tag{17}$$

$$t_{i,a} + D_{k^-(i,a),k^+(i_a^m,a)} \leq \sum_{h=1}^{H_{k^+(i,a)}} T_{k^+(i,a),h}^+ \times x_{i,a,k^-(i,a),h}^+ \quad \forall a \in A, i \in W_a \tag{18}$$

$$w_{i,a} = \sum_{h=1}^{H_{k^+(i,a)}} x_{i,a,k^+(i,a),h}^+ \quad \forall a \in A, i \in W_a \tag{19}$$

$$\sum_{i,a:k^-(i,a)=k} x_{i,a,k,h}^- \leq C_{k,h}^- \quad \forall k \in K, h \in H_k \tag{20}$$

$$\sum_{i,a:k^+(i,a)=k} x^+_{i,a,k,h} \leq C^+_{k,h} \quad \forall k \in K, h \in H_k \tag{21}$$

$$|V_p|\, y_p \leq \sum_{(i,a)\in V_p} u_{i,a} \quad \forall p \in P \tag{22}$$

$$r_{i,a} + T^+_{i,a} + 30 \leq r_{j,b} + T^-_{j,b} + L(1-y_p) \quad \forall p \in P, (i,a) = V_p(q), (j,b) = V_p(q+1), \forall q \in \{1,\ldots,|V_p|-1\} \tag{23}$$

$$r_{i,a} = 0 \quad \forall a \in A, i \in U_a : T^-_{i,a} < T^- \tag{24}$$

$$u_{i,a} = 1 \quad \forall a \in A, i \in U_a : T^-_{i,a} < T^- \tag{25}$$

$$v_{i,a} = 0 \quad \forall a \in A, i \in U_a \cap \mathcal{F} \tag{26}$$

$$x^-_{i,a,k,h} = 0 \quad \forall a, i, k, h : \left[T^-_{k^-(i,a),h}, T^+_{k^-(i,a),h}\right] \subset \left[I^-_a, I^+_a\right] \tag{27}$$

$$x^+_{i,a,k,h} = 0 \quad \forall a, i, k, h : \left[T^-_{k^-(i,a),h}, T^+_{k^-(i,a),h}\right] \subset \left[I^-_a, I^+_a\right] \tag{28}$$

$$D_{k^-_{(i,a)}k^+_{(i,a)}}\, w_{i,a} \leq D^a_{max} \quad \forall a \in A^m, i \in W_a \tag{29}$$

$$u_{i,a}, v_{j,a}, w_{l,a} \in \{0,1\} \quad \forall a \in A, i \in U_a, j \in V_a, l \in W_a \tag{30}$$

$$r_{i,a}, t_{l,a} \geq 0 \quad \forall a \in A, i \in U_a, l \in W_a \tag{31}$$

$$x^-_{i,a,k_1,h_1}, x^+_{i,a,k_2,h_2} \in \{0,1\} \quad \forall a \in A, i \in U_a \cup W_a, k_1 = k^-(i,a), h_1 \in H_{k_1}, k_2 = k^+(i,a), h_2 \in H_{k_2} \tag{32}$$

$$y_p \in \{0,1\} \quad \forall p \in P \tag{33}$$

$$z_a \in \{0,1\} \quad \forall a \in A^m \tag{34}$$

The objective (1) of problem (P) is to maximize the number of aircraft that satisfy the maintenance constraint. The constraints (2) - (7) are related to the aircraft. Constraint (2) is a flow constraint that specifies that the aircraft cannot leave an airport if it was not already at this airport or if it did not land at this airport. Please note that all the variables $v_{i,a}$ do not inevitably exist in constraint (2) since their existence depends on the circuit t in the initial route of the aircraft a. It is the same for the variables $w_{i,a}$, which only exist if the associated aircraft is scheduled for maintenance and if the corresponding initial flight was programmed before the maintenance. This applies to all the model constraints involving these variables. Constraint (3) is considered only for the aircraft that are scheduled for maintenance. It insures that an aircraft respecting the maintenance constraint executes the last flight preceding the scheduled maintenance, be it an existing flight, a fictive flight (noted $\hat{i}^m_a$), or a created flight. Please note that if a direct feasible solution for the initial problem is sought, the second member in this constraint can be replaced by the value 1 to insure that the maintenance constraints are respected. Constraint (4) indicates that each flight sequence (j,i) for an aircraft must respect the minimal turn-round/transit time for the aircraft, provided that flight j is initially programmed before flight i. Please note that if one of the two flights is cancelled, then constraint (4) is redundant because the constant L is sufficiently large to relax this constraint (L can be set to the difference in minutes between T^+ and T^-). Constraint (5) plays the same role as constraint (4) but between the flights envisaged and those potentially created. Constraint (6) sets an upper bound on the delay that can be assigned to a flight of an aircraft that is scheduled for maintenance. The landing date of all the flights

before the scheduled maintenance must be under this upper bound. Constraint (7) plays exactly the same role, setting landing dates for the flights that are potentially created.

The constraints (8) - (21) are related to the airports. Constraints (8) and (9) check that each takeoff is in a given time period at the initial airport. The unicity of the selected time period is ensured by constraint (10), which insures that a cancelled flight does not appear in any takeoff time period. Constraints (11), (12) and (13) have same interpretations as constraints (8), (9) and (10) to insure the landing of a flight in a given time period. Constraints (14) - (19) have the same interpretations as constraints (8) - (13), just considering created flights instead of existing flights. Constraints (20) and (21), respectively, insure the respect of the takeoff capacity and the landing capacity associated to each time period at each airport.

Constraints (22) and (23) are related to the passengers. Constraint (22) indicates that an itinerary is cancelled if at least one of its flight is cancelled. Constraint (23) is similar to constraint (4), controlling the passenger's transit time. The validity of this constraint relies on the fact that V_p is an ordered list of consecutive flights.

The constraints above are not sufficient to take all aspects of the problem into account. Thus, the flights that have already arrived or departed before the beginning of the recovery period cannot be affected (constraints (24) & (25)). Some flights may be cancelled by the disturbance (constraint (26), where the set $\mathcal{F}$ indicates all of the flights cancelled before the recovery period). In addition, according to constraints (27) & (28), some aircraft may be unavailable during a given period (apart from the maintenance period), with $[I_a^-, I_a^+]$ being the possibly empty period denoting the unavailability of the aircraft a. Moreover, according to constraint (29), a given aircraft a cannot cross a distance greater than a certain threshold, D_{max}^a.

Finally constraints (30) - (34) state the bounds of the variables in the model.

3.2 Relaxing airport capacities

An optimal solution z^* for problem (P), where all the variables are equal to 1, is a feasible solution to the problem from a maintenance perspective. However, if at least one variable z_a^* is equal to 0, then there is at least one aircraft a that cannot insure its scheduled maintenance. In this case, we apply the repair heuristic described in Section 3.4. After setting the variables z obtained by solving problem (P), we must return to a problem closer to the original problem that take the passengers into account.

Analyzing problem (P) shows us that constraints (20) and (21) related to the airport capacity are the key constraints because they make the variables x^+ and x^- not only indicative but decisional. For this reason, we next consider a partial relaxation of these constraints using the tolerance variables:

- $s_{k,h}^-$, an integer variable that represents the number of landings authorized beyond the arrival capacity during the time period h at airport k, and
- $s_{k,h}^+$, an integer variable that represents the number of takeoffs authorized beyond the departure capacity during the time period h at airport k.

Model P(Q), associated to a parameter Q that penalizes the violation of the capacity constraints, is then obtained from problem (P) by replacing objective (1) with objective (35) and constraints (3), (20) & (21) with constraints (36), (37) & (38)

$$\text{P}(Q) \text{ maximize } \alpha \sum_{p=1}^{|P|} G_p y_p - \beta \sum_{a \in A} \sum_{i \in F_a} r_{i,a} - \sum_{k \in K} \sum_{h \in H_k} Q(s^-_{k,h} + s^+_{k,h}) \tag{35}$$

$$\text{subject to } (2), (4)-(19), (22)-(33),$$

$$u_{i^m_a,a} + v_{i^m_a,a} + \sum_{j=1}^{i^m_a} w_{j,a} = z^*_a \quad \forall a \in A^m \tag{36}$$

$$\sum_{i,a:k^-(i,a)=k} x^-_{i,a,k,h} \leq C^-_{k,h} + s^-_{k,h} \quad \forall k \in K, h \in H_k \tag{37}$$

$$\sum_{i,a:k^+(i,a)=k} x^+_{i,a,k,h} \leq C^+_{k,h} + s^+_{k,h} \quad \forall k \in K, h \in H_k \tag{38}$$

In model P(Q), the objective is to maximize the number of passengers that arrive at their final destination while minimizing the total delay. The coefficients α and β in (35) allow the two costs to be weighted and standardized. Please note that, in practice, the delay has a slight weight compared to the transportation of the passengers to their final destination. However, adding the delay to the objective function allows us to avoid unjustified delays that can occur when a flight's arrival time can be set within an interval (e.g., [11:30,11:50]). In this case we prefer to fix the landing to minimize the delay. When $Q = 0$, the capacity constraints are relaxed, whereas when Q moves towards $+\infty$, the capacity constraints are not relaxed. Our solution is based on an iterative principle in which the model P(Q) is solved at each iteration for an increasing value of Q. The process stops when Q moves closer to $+\infty$ (in practice, when a certain CPU time is reached) or when a feasible solution is obtained. This method allows the solver to provide a starting solution for each iteration, which makes it possible to accelerate the solution process or at least to improve the current solution quickly, if the problem cannot be solved exactly.

3.3 Maximizing the number of transferred passengers

After tightening airport capacity constraints as much as possible, all variables of the problem, except variables related to flight cancellations, can be set with the goal of maximizing the number of passengers transferred. Solving problem (P) provides us with the initial date and the final date of a given existing or created flight. From these dates, the associated time slots can be defined. Then all the variables x^+ and x^- associated with the other time slots can be set to 0 in the constraints (10), (13), (16) & (19). Finally, we solve one last linear program (P') (below) in which the objective focuses only on the arrival bound for the groups of passengers.

$$\text{(P')} \quad \text{maximize } \sum_{p=1}^{|P|} G_p y_p \tag{39}$$

$$\text{subject to } u_{i-,a} + v_{i,a} \geq u_{i,a} + v_{i,a} + w_{i,a} \quad \forall a \in A, \forall i \in F_a \setminus \{i^m_a\} \tag{2}$$

$$u_{i_a^m,a} + v_{i_a^m,a} + \sum_{j=1}^{i_a^m} w_{j,a} = z_a^* \quad \forall a \in A^m \tag{36}$$

$$u_{i,a} = \sum_{h=1}^{H_{k^-(i,a)}} x^-_{i,a,k^-(i,a),h} \quad \forall a \in A, i \in U_a \tag{10}$$

$$u_{i,a} = \sum_{h=1}^{H_{k^+(i,a)}} x^+_{i,a,k^+(i,a),h} \quad \forall a \in A, i \in U_a \tag{13}$$

$$w_{i,a} = \sum_{h=1}^{H_{k^-(i,a)}} x^-_{i,a,k^-(i,a),h} \quad \forall a \in A, i \in W_a \tag{16}$$

$$w_{i,a} = \sum_{h=1}^{H_{k^+(i,a)}} x^+_{i,a,k^+(i,a),h} \quad \forall a \in A, i \in W_a \tag{19}$$

$$\sum_{i,a:k^-(i,a)=k} x^-_{i,a,k,h} \leq C^-_{k,h} \quad \forall k \in K, h \in H_k \tag{20}$$

$$\sum_{i,a:k^+(i,a)=k} x^+_{i,a,k,h} \leq C^+_{k,h} \quad \forall k \in K, h \in H_k \tag{21}$$

$$|V_p|\, y_p \leq \sum_{(i,a)\in V_p} u_{i,a} \quad \forall p \in P \tag{22}$$

$$y_p \in \{0,1\} \quad \forall p \in P \tag{33}$$

Nonetheless, the solution obtained may not be feasible from the maintenance perspective. To make the solution feasible, we apply a "repair" heuristic to the solution obtained from this stage of the method.

3.4 Using a heuristic to repair infeasible flight schedule

Even when the model described in the previous sections is applied, maintenance issues may lead to an infeasible solution, meaning that at least one aircraft is not able to reach the airport assigned for its scheduled maintenance on time without cancelling flights. For each aircraft that cannot arrive on time, we use a dynamic programming-based (**DP**) algorithm to find a suitable route to allow it to reach the assigned airport for its scheduled maintenance, with the goal of cancelling as few flights as possible in the process. In some cases, due to airport capacity issues or longer-than-authorized flights for a given aircraft, it is not even possible to schedule a direct flight to the airport. Consequently, a brand new route may be needed, which profoundly modifies the structure of the input solution. The DP algorithm uses a spatio-temporal graph, in which the vertices are situation pairs (*airport, time slot*) and each arc corresponds to the possibility of reaching one situation from another. Using this graph, we compute the most profitable path between the initial situation and the final situation, which is equal to the pair (*maintenance airport, deadline*). Practically speaking, it would be much too time-consuming to create a vertex for each possible minute; instead, we use one-hour time periods.
Since it does not modify the flights that aircraft with scheduled maintenance

execute, this algorithm is a heuristic. Consequently, the quality of the solution may depend on the order in which the aircraft are considered.

In practice, we always obtained a feasible solution at the end of the first stage of our method. The next stage applies an oscillation strategy to improve our initial feasible solution.

4 Improving the solution by using an oscillation strategy

Strategic oscillation is closely connected to tabu search (Glover and Laguna (1997)) and aims to obtain an efficient interaction between intensification and diversification. It is based on the notion of *critical level* (e.g. boundary of the feasible region) that corresponds in many cases to a point where the method normally stops or turn around. The method allows the process to cross this boundary for a specified depth. Strategic oscillation defines an alternating rhythm designed to cross critical levels in different directions. The process of repeatedly approaching and crossing the critical level from different directions creates an oscillatory behavior. The critical levels depend on the problem setting, for example, a particular form of graph structure, a subset of the feasible domain, a target value, balance states, local optima and so on. Strategic oscillation has been applied efficiently to solve knapsack problems by many authors (see for instance Hanafi and Fréville (1998)). Our oscillation strategy is summarized in Algorithm 1.

Algorithm 1 Oscillation-based algorithm

Require: an initial feasible solution.
  `// Step 0: Initialization`
  Choose an initial feasible or non-feasible solution;
  Initialize the parameters of the oscillation strategy;
  while *No stopping criterion is satisfied* **do**
    `// Step 1: Constructive Phase`
    Step 1.1: Generate feasible routes for aircraft;
    Step 1.2: Generate feasible itineraries for passengers;
    Step 1.3: Assign routes to aircraft and itineraries to passengers simultaneously;
    `// Step 2: Destructive Phase`
    Step 2.1: Delete routes for some aircraft and cancel the corresponding passengers itineraries;
    Step 2.2: Assign cancelled passengers to existing flights;
    Step 2.3. Update the parameters;
  end while

During the constructive phase (Step 1), we generate a set of feasible flight sequences for the fleet of aircraft (Step 1.1) according to the constraints associated with the aircraft and the airports. Then, we generate a set of possible itineraries for the groups of passengers (Step 1.2) with at least one cancelled flight. Generating the itineraries for a given group depends on which airport is the group's current location, the airport's available start time and the group's latest possible arrival date. While generating the possible passenger itineraries, we have to take the newly

created flights associated with the aircraft into account. During the destructive phase (Step 2), routes are deleted and passenger itineraries are cancelled (Step 2.1) to make room for future successful constructive phases and the passengers on the deleted routes are re-assigned to existing flights (Step 2.2).

Steps 1.1 and 1.2, as well as some of Step 2.1, are managed by constructive and destructive heuristics. Steps 1.3 and 2.2, as well as some of Step 2.1, are managed by a mixed integer programming model to assign aircraft to routes and groups of passengers to the various flights. In practice, the algorithm always remains in the feasible space. In our case a critical level corresponds to a local optimum where it is not possible to improve the objective function value after adding (or cancelling) flights and itineraries. In that case we can allow the constructive (or destructive) phase to continue for a given depth, by choosing the less degrading move.

4.1 Creating and deleting routes for aircraft

In the oscillation strategy, we not only try to assign the passengers to a set of existing flights, we also create new aircraft routes to permit additional itineraries to be created. Routes are generated in the available aircraft intervals. These intervals are located between two consecutive flights or after the last flight. In the former, the route has to be a circuit. In the latter, the route's arrival airport may be different from its initial airport (positioning constraints may be added). The route-generation algorithm is based on a truncated best-first search.

It takes as input the current location of a given aircraft, a departure airport and an arrival airport, and seeks a set of routes for the aircraft, starting from its current location to its destination. Please note that the final destination is not necessarily fixed. The partial routes are stored in a heap, which is initialized with all possible routes that involve one flight from the selected departure airport to any destination. At each step, the most profitable partial route is selected, and the algorithm tries to attach a flight to the end of this route. Each new route obtained, if it solves the problem, is recorded in the solution set; otherwise, it is pushed into the partial-solution heap and the algorithm continues. The algorithm stops when a stopping criterion is reached (e.g., number of solutions, number of explored nodes). The choice of the exploration strategy is a major issue in this step because, if the remaining airport capacities and time windows are too tight, a poor choice may lead to no solutions at all at the end of the algorithm. We tested several strategies: less visited airports first, those closest to the aircraft destination (if they exist) first, shortest flight, less overloaded airports first, earliest possible flight and passenger preferences first. Practically speaking, choosing a direct flight for at least one set of passengers leads to the best results.

The main idea behind the route deletion step (Step 2.1 in Algorithm 1) is that the aircraft are not necessarily exploited optimally for some flights, which may mean that there are many empty seats in some aircraft. Deleting these routes allows:

- the remaining departure and arrival capacity of the airports to be increased, thus allowing new flights to be created; and
- the number of passengers on the other flights to be reassigned.

The deletion process insures the feasibility of the solution by removing only the circuits executed by the aircraft. Moreover, the aircraft scheduled for maintenance are not taken into consideration. The ratio of empty seats in a circuit is used as a selection criterion. Other criteria can be used, such as the total number of passengers, the number of seats in the aircraft and/or the type of flights in the circuit. In practice, we consider the cancellation of routes if it can be paired with an effective re-assignment of the passengers that will improve the quality of the current solution. If it is not the case we can also accept the cancellation of some routes to diversify the search.

4.2 Creating and cancelling passenger itineraries

The variables of the model used in our oscillation strategy (Steps 1.3 and 2.2 in Algorithm 1) are the possible aircraft routes and the possible passenger itineraries. The itinerary-generation heuristic is the same as the route-generation algorithm, except that the number of possible partial solutions is far smaller. This heuristic takes as input the current location of a group of passengers, this group of passengers, a departure airport and an arrival airport and seeks a set of itineraries for the passengers in this group, starting from their current location to their destination. The algorithm used for route creation is still valid for this heuristic, except that in the present case the arrival airport is constrained.

The effectiveness of the method depends on the pertinence of the choice criteria used to sort the partial solutions. We tested several strategies: less visited airports first, those closest to the final destination first and minimum idle time. Selecting the partial solutions that are closest to the final destination was generally the best strategy and produced the best results.

Cancelling the passenger's itinerary is directly connected to the deletion of the aircraft's routes. Thus we do not cancel the itineraries unless certain flights are deleted and the corresponding passengers cannot be re-assigned.

4.3 Assigning passengers and aircraft

The assignment step (Steps 1.3 and 2.2 in Algorithm 1) finds a solution that combines new route creation and/or effective passenger re-assignment. From a set of routes associated with the planes and a set of possible itineraries associated with these passengers, we solve a multidimensional knapsack problem with binary and integer variables. Binary variables correspond to the choice of the routes for the planes, and integer variables correspond to the number of passengers in each group that are re-assigned to the new itineraries.

The model considers the constraints associated with the airports and the capacity of the cabins of the planes. The objective function of the model is to

$$\text{maximize } z = \sum_{j \in I} \sum_{p \in P} c^1_{j,p} y_{j,p} - \sum_{i \in R} \sum_{a \in A} c^2_{i,a} x_{i,a},$$

where R is the set of possible routes for the planes; I is the set of possible itineraries for passengers; P is the set of group of passengers ; $c^1_{j,p}$ is the cost associated to group $p \in P$ of passengers relatively to itinerary $j \in I$ computed as the cancellation cost of a passenger of group p less the delay cost; $c^2_{i,a}$ is the cost associated with the creation of flights for plane a in a route $i \in R$, $x_{i,a}$ is a binary

variable fixed at value 1 if plane a takes route $i \in R$; and $y_{j,p}$ is an integer variable fixed to the number of passengers in group p who take itinerary $j \in I$.

The constraints of the problem indicate that each aircraft cannot take more than one route among the routes generated in the same interval of availability. In addition the arrival (resp. departure) capacities have to be respected, as the aircraft capacity constraints. Finally the number of affected passengers in a given group for all the itineraries should not exceed the cardinality of this group, and some parts of the aircraft's route are fixed. Like all knapsack problems, this problem is NP-hard, and is solved within a given time limit in the context of the challenge.

5 Computational results

In this section, we summarize the computational results obtained with our algorithm during the ROADEF Challenge. The computer used contained an AMD Turion 64x2 processor and 2 GB of RAM. We used the Ilog CPLEX solver (11.1 version) to solve our MIPs. The execution time was limited to 10 minutes for the evaluation. There were three sets of instances: A, B and X. Set A contained 10 instances and was used for the qualification phase of the Challenge. Sets B and X contained respectively 10 and 8 instances, and the results from these instances were used to rank the 11 finalists. All the instances can be found on the Internet[3].

Table 1 Results for data set A.

Inst.	airports	aircraft	itin.	BQ	OQ	OF
A01	35	85	1943	23 789.05	28 155.6	-11 059.15
A02	35	85	1943	83 515	126 684.7	90 429.4
A03	35	85	1943	131 694.6	131 694.6	79 647.95
A04	35	85	1943	108 081.9	523 678.8	36 379.55
A05	35	85	3959	3 717 376.35	8 092 977.55	1 332 420.6
A06	35	85	1872	44 305.05	71 355.25	61 752.9
A07	35	85	1872	202 247.75	245 696.7	157 931.3
A08	35	85	1872	329 521.85	329 521.85	199 834.35
A09	35	85	1872	187 144.4	109 6303.85	128 526.4
A10	35	85	3773	7 210 166.9	17 166 321.75	3 570 361.9

Table 1 reports the results obtained for the data set A. In the left-hand columns, it gives the name of the instance, the number of airports, the number of aircraft and the number of itineraries. The organizers have not yet published the final results obtained by the finalists for this data set. Thus, in the right-hand columns, we provide the results for the ten instances used in the qualification phase: the best value reported during the qualification phase (column BQ), the value obtained by our "qualification" algorithm (column OQ) and the value obtained by our "final" algorithm (column OF).

Before presenting our analysis of the results reported in Table 1, we need to describe the main differences between our "qualification" algorithm and our "final" algorithm. For the qualification phase, our method was based on three main

Table 2 Official results for data set B.

Instance	airports	aircraft	itin.	Feasible	Quality Init.	Quality Fin.
B01	44	255	11214	MIP	0.80	0.89
B02	44	255	11214	Repair	0.47	0.71
B03	44	255	11214	MIP	0.83	0.90
B04	44	255	11214	MIP	0.82	0.90
B05	44	255	11214	MIP	0.86	0.96
B06	44	255	11565	MIP	0.76	0.91
B07	44	255	11565	Repair	0.26	0.73
B08	44	255	11565	MIP	0.76	0.89
B09	44	255	11565	MIP	0.77	0.90
B10	44	255	11565	MIP	0.82	1.00

components: a mathematical model to generate an initial feasible solution to the problem (the maintenance problem did not occur in data set A), a heuristic to assign passengers to existing flights, and a heuristic to create new flights in the schedule. It did not include the oscillation strategy described in Section 4. For the final phase of the Challenge, our method was based on the same three components, but included the oscillation strategy.

It is really interesting to observe the progression of our method between the two Challenge phases. In particular, we improved the values for all the instances, and we improved the reference value (based only on the qualification phase) for 8 instances. These improvements demonstrate the positive impact of our oscillation strategy. In addition, the use of mathematical programming to assign the passengers and to choose the "good" routes for the aircraft also had a real impact on the final results. Our method provided a solution with a negative cost for the instance A01. This negative cost was made possible through the cancellation process, which involved using the initial schedule. This means that in some situations the flight schedule of the airline can really be improved by regrouping passengers on the same flights.

Finally, let us note that our algorithm was able to find better solutions for some instances of this data set by changing some parameters. However, our objective was to propose a method that is as robust as possible to avoid unpleasant surprise on the unknown instances.

Tables 2 and 3 summarize the results for data set B and X that were published on the ROADEF Challenge website. In addition to the information about the instances (columns Instance, airports, aircraft & itin.), Tables 2 and 3 show the quality of our lower bound among all the lower bounds produced by the finalists (columns Quality Init. & Quality Fin.). This quality is defined as $((zw(I) - z(I))/(zw(I) - zb(I)))$, where $zb(I)$ (resp. $zw(I)$) denotes the best (resp. worst) objective function value found on instance I, and $z(I)$ denotes the objective function value obtained by our method. Please note that for the final Challenge ranking, if a method did not provide a solution or if the solution provided was infeasible for instance I, its score was set to two times $zw(I)$. We provide in these tables the quality of our lower bound after the initial phase (Quality Init.) and at the end of the method (Quality Fin.). We also specify (column Feasible) if our first feasible solution was obtained with the MIP models or the repair heuristic.

Table 3 Official results for data set X.

Instance	airports	aircraft	itin.	Feasible	Quality Init.	Quality Fin.
XA01	35	85	1943	MIP	0.89	1.00
XA02	35	85	3959	MIP	0.95	1.00
XA03	35	85	1872	MIP	0.80	1.00
XA04	35	85	3773	MIP	0.20	1.00
XB01	44	255	11214	MIP	0.92	0.96
XB02	44	255	11214	MIP	0.94	0.99
XB03	44	255	11565	MIP	0.89	0.96
XB04	44	255	11565	MIP	0.91	1.00

Tables 2 and 3 show that our method obtained interesting results, in particular for the data set X, which contains instances that were unknown by the finalists (so no specific fine-tuning was possible). Instances XA01 to XA04 were generated from the instances in data set A. These results confirm the results obtained for set A. For data set B, the weaker results obtained for instances B02 and B07 are closely connected to the significant disruption caused by the maintenance constraints. Indeed, our algorithm really needs to move away from the initial schedule to restore feasibility, which is confirmed by the poor quality of our initial lower bound. Clearly, the oscillation strategy saves our method. Nonetheless, for the most disrupted instances (B05 and B10) from the perspective of airport capacity, our method obtains much more interesting results.

Some further experiments confirmed that one of the weakness of our method is the generation of a large number of new aircraft routes. As mentioned above, we tested several strategies to try to enhance this procedure: less visited airports first, those closest to the aircraft destination (if they exist) first, shortest flight, less overloaded airports first, earliest possible flight and passenger preferences first. None of these strategies really dominated the others for the three sets of instances. The computational experiments confirmed that, on average, choosing a direct flight for at least one group of passengers leads to the best results. In addition, it seems that our route deletion process could be extended and improved to take into account more than the circuits. In fact, if the cancellation of other flights were allowed, more aircraft routes could probably be created. Finally, please note that the final ranking of the ROADEF Challenge can be found on the Internet.

6 Conclusions

Several studies have been devoted to Disruption Management in the Airline Industry, which is a current subject with very significant monetary stakes. To solve this problem in the context of the ROADEF Challenge, we chose a hybrid method with an oscillation strategy, which combines heuristics and mathematical programming. After generating an initial feasible solution with mixed integer programming programs and a heuristic, we alternate between constructive and destructive phases to improve this solution by adding and dropping some parts of the aircraft routes. Assigning passengers to aircraft and aircraft to routes is managed by a mixed integer programming model. The possible routes are

generated using heuristic methods. The results obtained for this Challenge are really encouraging.
Our method could still be improved by integrating adaptative memory into the search process and by scaling the parameters. Exploiting dual information to generate the routes or to authorize visiting infeasible solutions are other possible elements to explore.

Acknowledgements

The present research work has been supported by International Campus on Safety and Intermodality in Transportation the Nord-Pas-de-Calais Region, the European Community, the Regional Delegation for Research and Technology, the Ministry of Higher Education and Research, and the National Center for Scientific Research. The authors gratefully acknowledge the support of these institutions.
The authors also would like to acknowledge the anonymous reviewers for their constructive comments, and the Editor of the journal.

References

Andersson, T.: 2006, 'Solving the flight perturbation problem with meta heuristics'. *Journal of Heuristics* **12**, 37–53.

Bratu, S. and C. Barnhart: 2006, 'Flight operations recovery: New approaches considering passenger recovery'. *Journal of Scheduling* **9**, 279–289.

Cao, J. and A. Kanafani: 1997, 'Real–Time Decision Support for Integration of Airline Flight Cancellations and Delays Part I: Mathematical Formulation'. *Transportation Planning and Technology* **20**, 183–199.

Clausen, J., A. Larsen, J. Larsen, and N. J. Rezanova: 2009, 'Disruption management in the airline industry–concepts, models and methods'. *Computers & Operations Research* **doi: 10.1016/j.cor.2009.03.027**.

Glover, F. and M. Laguna: 1997, *Tabu Search*. Kluwer Academic Publishers, Boston.

Hanafi, S. and A. Fréville: 1998, 'An efficient tabu search approach for the 0–1 multidimensional knapsack problem'. *European Journal of Operational Research* **106**, 659–675.

Jarrah, A., G. Yu, N. Krishnamurthy, and A. Rakshit: 1993, 'A decision support frame–work for airline flight cancelations and delays'. *Transportation Science* **27**, 266–280.

Lettovsky, L., E. L. Johnson, and G. L. Nemhauser: 2000, 'Airline crew recovering'. *Transportation Science* **34**, 337–348.

Maniezzo, V., T. Stützle, and S. Voß(eds.): 2009, *Matheuristics: Hybridizing Metaheuristics and Mathematical Programming*, Annals of Information Systems , Vol. 10. Springer.

Sriram, C. and A. Haghani: 2003, 'An optimization model for aircraft maintenance scheduling and re-assignement'. *Transportation Research Part A: Policy and Practice* **37**, 29–48.

Teodorovic, D. and S. Guberinic: 1984, 'Optimal dispatching strategy on an airline network after a schedule'. *European Journal of Operational Research* **15**, 178–182.

Teodorovic, D. and G. Stojkovic: 1990, 'Model for Operational Daily Airline Scheduling'. *Transportation Planning and Technology* **14**, 273–284.

Teodorovic, D. and G. Stojkovic: 1995, 'Model to reduce airline schedule disturbances'. *Journal of Transportation Engineering* **121**, 324–331.

Yan, S. and C. G. Lin: 1997, 'Airline scheduling for the temporary closure of airports'. *Transportation Science* **31**, 72–82.

Yu, G.: 1998, *Operations Research in the Airline Industry*. Kluwer Academic Publishers, Dordrecht.

Yu, G. and X. Qi: 2004, *Disruption management: framework, models and applications*. World Scientific Publishing Company.