Economics & Management Series *EMS-2012-18*

Disruption Management for Commercial Aviation

Wenkai Li
International University of Japan

Mark Wallace
Monash University

November 2012

IUJ Research Institute
International University of Japan

Disruption Management for Commercial Aviation

Wenkai Li[1,2*], Mark Wallace[1],
[1]Faculty of Information Technology, Monash University, Australia
[2]Graduate School of International Management, International University of Japan, Niigata 949-7277, Japan

Abstract

Airlines are constantly facing operational disruptions such as reduced airport capacity because of bad weather or strikes, unexpected aircraft unavailability due to mechanical failures, and delayed or cancelled flights. In view of this, ***ROADEF*** organized a worldwide challenge to explore the problems encountered in real world airlines when disruptions happen and find approaches to tackle them. In this paper, a new continuous time aircraft routing model is developed which can minimize aircraft delay cost accurately and efficiently handle all types of disruptions encountered in ***ROADEF***. Applying a new decomposition algorithm, near optimal solutions for aircraft routing can be obtained. A passenger re-accommodation model is solved subsequently using the results from the aircraft routing model as input. Competitive results are obtained applying the proposed approach to instances provided by ***ROADEF***.

Keywords: Disruption Management, *ROADEF* Challenge 2009, Airline

1. Introduction

Typically, airline schedule planning is effected by solving several problems sequentially. *Schedule design* generates profit maximizing flight schedules (when and where to offer flights) by considering origin–destination (OD) market demands. *Fleet assignment* then assigns an aircraft type to each flight leg so at to minimize operational and spill costs. Then for each aircraft, *aircraft routing* determines the sequence of flight legs to be flown by each individual aircraft. *Crew scheduling* finally assigns crew members (cabin and cockpit crews) to each flight leg so that the crew costs are minimized. Since deregulation in the 1970s, airlines have been operating near their optimal capacities, allowing little slack in flight durations in the hope that the airline schedule will operate as planned. However, this optimistic scenario is rarely achieved

*Corresponding author. E-mail: lwk@iuj.ac.jp

in practice since airline schedules are frequently disrupted by bad weather, aircraft mechanical failures, airport fuel shortage, surface transportation congestion, strikes, fluctuating customer demands and many other intangible factors. In fact, airlines have been suffering from increasing levels of disruption in the last decades, over and above the huge impact of 7/11 and its aftermath. In 2003, fuel shortage hit Sydney airport caused numerous flights delayed or cancelled affecting around 2,500 passengers (BBC news 2003). A recent computer glitch hit US flights with dozens of flights cancelled or delayed in Hartsfield-Jackson Atlanta International Airport alone (BBC news 2009). The Federal Aviation Administration (FAA) reported a 58% increase in delays from 1995 to 1999, and a 68% increase in flight cancellations over the same period (Schaefer 2005). In 2000, about 30% of the flight legs were delayed and about 3.5% of these flight legs were cancelled in one major U.S. airline (Lan et al. 2006). Research showed that a 1% increase in air traffic results in a 5% more delays, and air traffic in the United States and Europe was predicted to double in the next 10 to15 years (Schaefer 2005). These disruptions have imposed huge costs on airports and airlines. It was estimated that the total cost to Hartsfield airport due to cancellations was $250.9 million in 1999 (Schaefer 2005). The impacts of irregularities encountered by a single US major airline exceeded $400 million per year (Bratu et al. 2006).

Disruption management techniques have emerged and are becoming essential for robust airline scheduling and operation. Some airlines are now shifting towards ensuring that planned schedules are robust and allow for efficient recovery (Kohl et al. 2007), but, however robust the schedule, disruption handling is an essential feature of airline operations. The French Operational Research (OR) and Decision Support Society thus organized a worldwide competition in 2008/9 (*ROADEF 2009 Challenge*, referred to as *the Challenge* later in this paper) for airline disruption management. The problem of *the Challenge* was to find the best aircraft routing and passenger re-accommodation solution to recover from a fixed set of disruptions within a specified period of time, with given regular operating constraints. The aim of *the Challenge* was to explore the problems encountered in real world airlines when disruptions happen and find efficient approaches to solve them. *The Challenge* provides two sets of problem instances (each with 10 instances), each set having a different size. Four types of disruptions were considered:

1. airport capacity: restrictions on the number of departures and landings (including closure) for a given period;
2. aircraft unavailability: a time and duration of an aircraft unavailability (i.e. due to an unserviceability or fault);
3. flight cancellations; and
4. flight Delays.

Crew recovery is not included. Operating costs of switching aircraft (e.g. flying a larger plane than required is more expensive) are ignored. Because aircraft of the same type have similar cabin and cockpit features, and can therefore be flown by the same crew, airlines typically avoid reassigning flights to planes of a different type during disruptions Consequently in *the Challenge* aircraft swaps are limited to within an aircraft family. Some researchers (Rosenberger et al. 2003, Bratu et al. 2006) also tried restricting swaps to aircraft of similar capacities, to minimize passenger disruption.

Airlines typically recover from disruptions in stages: aircraft recovery follow by crew recovery and finally passenger recovery (Rosenberger 2003). The disruption management problem in *the challenge* requires integration of the resource handling at these different stages, such as aircraft routing, passenger itinerary handling, slot management and maintenance handling, etc. Moreover, since operations controllers must react to disruptions as soon as they occur, recovery decisions have to be made quickly, usually in a matter of minutes. This paper will develop techniques for fast disruption handling, taking into account aircraft and passenger recovery, and ensuring maintenance requirements are satisfied. It has direct application to all forms of transportation.

2. Literature review

Barnhart et. al. (2004) reviewed current approaches and challenges facing airline scheduling. They pointed out that airline schedule recovery problems are particularly challenging, involving multiple highly-constrained resources and requiring a global view of the system. Teodorovic and Guberinic (1984) studied the aircraft recovery problem. They considered disruption due to aircraft unavailability with the objective of minimizing total passenger delays. A connection network model was developed and a small example with only 8 flights was solved. Teodorovic et al. (1990) later extended their work by considering airport curfews. A greedy algorithm was used in which aircraft rotations were built one by one. The solution quality could be far from optimal . Jarrah et al. (1993) developed a timeline network model to handle two kinds of flight disruptions: cancellation and re-timing. Flights and aircraft were represented using different nodes in their time-line network. An arc from a flight node to an aircraft node represented a flight-to-aircraft assignment. Two minimum cost flow network models were developed for handling delay and cancellation, respectively. Results from small test instances involving three airports were reported. Talluri et al. (1996) built connection network models to precompute alternatives for swapping aircraft among flights when disruption happens. Aircraft maintenance and crew pairings were not considered in their solutions. Lettovsky (1997) developed an integrated MIP model for crew, aircraft and passenger recovery. Due to the extremely large problem size, he applied Bender's decomposition algorithm which involves a master problem and three sub-problems. The master problem, Schedule Recovery, included key constraints such as slot capacity restrictions, flight seating capacity, and itinerary balances. Initial plans for reassigning equipment to cope with delays and cancellations were determined from the master problem. Then three sub-problems, Aircraft Recovery, Crew Recovery and Passenger Flow, were solved to return Bender's feasibility and optimality cuts corresponding to the current equipment assignments. These cuts were added into the master problem at the next iteration. Results from small instances were reported in his work. Bard et al. (2001) proposed a time-band network to deal with disruptions. The model was a minimum cost flow network model with side constraints. The problem was solved by relaxing the integrality constraints first and then deriving integer-valued solutions to create a schedule. They tackled the solution quality issue by initially setting coarse time-band lengths and then reducing them systematically until a satisfactory solution quality was achieved, or the CPU limit was exceeded. The test instances in Bard et al. (2001) consisted of 162 flights, 27 aircraft and 30 stations. The solution time ranged from several seconds to several hundred of seconds. Rosenberger et al. (2003) proposed an aircraft selection heuristic (ASH) to search a subset of other aircraft for potential swaps with a disrupted aircraft. The reduced number of aircraft involved in the swaps reduced the problem size and the computation time. The CPU time to solve an instance involving 96 aircraft and 469 legs was less than 16 seconds. In their paper, crew and passenger recoveries were not included. When generating possible aircraft routes, alternatives of flight delays and ferry flights were not fully captured, and this could influence crew and passenger recovery. Bratu et al. (2006) presented two models that address aircraft and crew recovery with a passenger-centric objective. In the Disrupted Passenger Metric model (DPM) they implemented, delay costs were approximate. They then used a Passenger Delay Metric model (PDM) to calculate delay costs more accurately by generating a list of candidate recovery itineraries. They showed that PDM takes too long to solve, and was thus not suitable for real time disruption management. It was assumed that itineraries would only contain one or two flight legs in their model. To generate a maintenance feasible solution, it was sometimes necessary to solve their model iteratively. Kohl et al. (2007) provided a general introduction to airline disruption management process and some commonly used techniques to build flexibilities into airline schedules. They introduced the architecture of their disruption management system, called "Descartes" which included a dedicated passenger recovery solver, a dedicated aircraft recovery solver and a dedicated crew recovery solver. Two integration methods were discussed: ISR (integrated sequential recovery) constructed an integrated solution based on solutions from the dedicated solvers. The challenge in designing ISR was to generate high quality integrated solutions with respect to all resources. It also sometimes needed an iteration process to find a feasible integrated solution; TIRS (tailored integrated recovery) tried to incorporate all resources into one model. TIRS was based on a time-band model which divides the time line into intervals. A simulated annealing method was implemented into TIRS to solve the model, but no results were reported from their prototype system.

Existing network models

The network models appearing in the literature for airline scheduling mainly include the time-line model (Jarrah et al. 1993; Hane et al. 1995) and the connection network model (Teodorovic et al. 1984, 1990, Talluri et al. 1996). In a time-line network, the activities of a station (an airport that an airline operates) are modeled using a time line whose length is the planning/scheduling horizon (Hane et al. 1995). Arrivals to/departures from the station add nodes to the time line at the departure (arrival plus connection) time. An arc connecting nodes at different time lines represents a feasible flight. An arc connecting nodes on the time line of a station represents the grounded aircraft.

In a connection network, flight legs are represented by a set of nodes. A directed arc connecting node *i* to node *j* means that flight *j* follows flight *i* immediately, using the same aircraft. A connection between nodes *i* and *j* is feasible if it satisfies requirements such as minimum turnaround time. The details of these requirements will be discussed further in this paper. There are also nodes representing the position of aircraft at the beginning and the end of the planning/scheduling period. An aircraft rotation can be represented by a path in the network. Finding a single feasible path if there is one, even with complicated maintenance requirements, is computationally tractable. This makes it possible to solve the problem using Dantzig-Wolf decomposition, with the expected disadvantage that the number of feasible paths grows exponentially with the number of flights. As an alternative to column generation and branch and price, or as a refinement, preprocessing steps can be employed to reduce the number of useful feasible paths to be considered (Lever 1996).

As pointed out by Barnhart et al. (1998), the advantage of involving fewer arcs in a time-line network is dominated by the richer modeling possibilities provided by connection networks, such as through revenue calculation, and maximum flying/elapsed time before aircraft maintenance. Thus Barnhart et al. (1998) implemented a connection network in the pricing subproblem generated in their column generation algorithm.

Existing network models with continuous/discrete time

Despite the continuous nature of flight delays, approximating flight delays by discretising the time axis and generating copy arcs has been employed in the literature extensively (Bratu et. al. 2006). Jarrah et al. (1993) represented alternative flight departure times using different aircraft nodes along the time axis. The limited number of aircraft nodes restricts the alternatives of delays considered in their work. Clarke (1997) incorporated several delay arcs into his time line (named time space network in his paper) network to consider flight delay alternatives. Similarly, Yan et al. (1997) and Thengvall (2001) also modeled the retiming of a flight by introducing several alternative sliding arcs and requiring only one of the arcs be true in the final solution in their time line network. Lettovsky (1997) considered a set of delay alternatives for each flight leg. The details of how to choose the delay alternatives are not discussed.

Bard et al. (2001) proposed a modified timeline network called a time-band network which divides the recovery period into discrete time intervals. Several parallel arcs emanating from a station-time node represent copies of a flight leg with different delay times, of which only one is allowed to take value of 1. Arc delay costs are calculated by the difference between the scheduled departure time and the earliest arrival time, which underestimates the delay costs (Bard et al. 2001). The model becomes a minimum cost flow network model with side constraints. To incorporate flight leg retiming, Bratu et al. (2006) discretized feasible departure time of a flight leg into one minute and created arc copies. They presented an algorithm to limit the generation of flight copies.

The main disadvantage of generating flight copies is that it increases the model size significantly. In the example illustrated by Bard et al. (2001), the transformed model of a 12-flight 10-hour recovery period and 30-minutes time interval problem involves 66 flight copies. Moreover, to increase the accuracy of the delay approximation, time interval has to be set small which dramatically increases the number of flight copies required.

Model Contributions

An extension of the time line network, the time band network has difficulties in handling some types of disruptions, such as maximum flying/elapsed time before aircraft maintenance. In view of this, an extended connection network is proposed in this paper which can readily handle disruptions. We extend the connection network in two aspects:

1. to model the connection network from a machine scheduling perspective;
2. to cope with different types of disruptions efficiently.

A continuous time aircraft routing model is developed which deals with flight delays explicitly by introducing a continuous time variable. Discretising the time axis and generating copy arcs for flight legs are thus avoided. The delay costs of flight legs can be calculated accurately using the time variables.

3. A new mathematical model

The goal of *the challenge* is to determine the recovery operations in case of disruptions to the planned flight schedule. The period an airline returns to its original planned schedule after disruptions is called recovery period (*RP*). The length of the *RP* ranges from one to several days in different benchmark problem instances. Only the schedule within the predetermined *RP* can be changed: Flights that (have arrived or) have already departed at the beginning of the period cannot be modified. If a passenger has arrived or has already left at the beginning of the *RP*, the part of his/her itinerary that is before the beginning of the *RP* cannot be modified (*ROADEF* 2009).

The main assumptions made in *the challenge* are:

- Aircraft swaps are allowed within the same family of aircraft types only;
- Crew scheduling is ignored;
- The airport surface capacity is not taken into account;
- Surface public transportation is assumed to have infinite capacity and zero operating cost

Airlines are primarily seekers of low operation costs and high service levels. These two sometimes conflicting objectives are incorporated into one objective in *the challenge*: a weighted sum of the actual airline operation costs and the passenger disutility costs. The passenger disutility costs (*DC*, eqn. 1), trying to measure the service level, include dissatisfaction cost for passenger whose flight delayed (*Delayed_P*), itinerary cancelled (*Cancelled_P*) and/or flight downgraded (*Downgraded_P*).

$$DC = \sum_{p \in Delayed\_P} C_p^{delay\_pax} + \sum_{p \in Cancelled\_P} C_p^{cancel\_pax} + \sum_{p \in Downgraded\_P} C_p^{down} \qquad (1)$$

The actual airline operation costs (*OC*) is shown is eqn. 2. *OC* composes the operating cost of new flights (*Created_f*) minus the operating costs of cancelled flights (*Cancelled_f*), plus food and accommodation costs for delayed passengers (*Delayed_P*) and ticket reimbursement and compensation for passengers whose itineraries are cancelled (*Cancelled_P*).

$$OC = \sum_{f \in Created\_f} C_f^{op} - \sum_{f \in Cancelled\_f} C_f^{op} + \sum_{p \in Delayed\_P} C_p^{delay\_legal} + \sum_{p \in Cancelled\_P} C_p^{cancel\_legal} \qquad (2)$$

The challenge enforces the requirement that airline operations return to the original schedule at the end of *RP* by adding a penalty on the discrepancy. This is measured by the extent to which the family/model/configuration of aircraft at each airport differs from the original schedule. In eqn (3), $NbFamily_a$ is the number of aircraft that failed to match with the original setup at airport a. C_{family} is the corresponding penalty. Similarly for $NbModel_a$ and $NbConfig_a$.

$$NC = \sum_{a \in Airports} \left(NbFamily_a C_{family} + NbModel_a C_{model} + NbConfig_a C_{config} \right) \tag{3}$$

The Roadef objective can then be represented as:

$$\min \quad TC = \alpha OC + \beta DC + \gamma NC \tag{4}$$

Where α,β,γ are weights associated with different costs. More discussions on cost components and settings of parameters can be found in Roadef file (*ROADEF* 2009).

3.1 Aircraft routing with disruptions

In this section, we propose a Flight Sequencing Model (FSM) for aircraft re-routing, flight retiming and/or cancellation when disruptions happen. The objectives are either to minimize the total flight delays and cancellations or to minimize the total cost defined by *the Challenge*. Underlying the model is an extended connection network in which each flight is represented as a node and each arc represents a possible connection for passengers between two flights. Extending from machine scheduling (Hui and Gupta, 2001), we model each aircraft as a machine and each flight as a task, in the aircraft routing model. A continuous delay variable is defined for each flight to allow it to be retimed.

Besides normal flight nodes, several types of special nodes are created within the connection network to handle different types of disruptions:

1). Start nodes. These comprise:

a) the last flight in an aircraft rotation that has departed before *RP*;

b) an artificial start node at the beginning of *RP* for aircraft with empty original rotation;

c) maintenance start nodes for aircraft whose maintenance period spans across or starts at the beginning of *RP*;

d) unavailable start nodes for aircraft whose unavailable time spans across or starts at the beginning of *RP*.

The nodes created in c) or d) supersede nodes created in a) or b) if any conflicts happen.

2). Maintenance nodes.

These nodes are created for aircraft whose maintenance period falls within *RP*.

3). Aircraft unavailable nodes.

If the unavailable time of an aircraft falls within *RP*, an aircraft unavailable node is created at each airport, to allow the flexibility for the aircraft to land at each airport before the unavailable time.

4). End nodes.

There is an end node for each aircraft at the airport where it was originally scheduled to finish at the end of the *RP*.

Notation

We introduce the following notations for our model:

Indices

a: aircraft

c: cabin classes

f, f': flights

i: itineraries

p: airports

s: slots in airports

Sets

A_f: Aircraft that can serve flight f

$AC_{ff'}$: Aircraft that can fly both flights f and f'

$ANC_{ff'}$: Aircraft that fly flight f but cannot fly f'

AFX_f: the original fixed registration of flight f

$book_i$: set of flights constituting itinerary i

C: set of cabin classes (i.e., First, Business, Economy).

END: set of end nodes

FA_a: All flights that aircraft a can serve

FC_f: a flight f' is included into set FC_f if there exists at least one aircraft which can fly both f and f'

FM: set of maintenance nodes

FP_f : feasible predecessors of flight f

F_r: set of flights which are scheduled in the recovery period.

FSN: flights which are defined as start nodes

FS_f: feasible successors of flight f

FUA_a: set of dummy nodes created at airports for unavailable aircraft a.

FXN: set of flights whose registration is fixed

FXR_a: fixed rotation of aircraft a. This includes flights served by a before RP

ITN: set of itineraries

$ML_{ff'}$: pair of flight legs corresponding to a multi-leg flight

P: set of airports

SL_p: set of slots in airport p

UA: set of unavailable aircraft

UAN: set of unavailable aircraft nodes

Decision variables:

AX_{fps}: 1 if flight f arrivals at slot s of airport p, 0 otherwise

AY_{fps}: 1 if flight f arrivals after SE_{ps}, 0 if flight f arrivals before SS_{ps}

CX_f : 1 if flight f is cancelled, 0 otherwise

DX_{fps}: 1 if flight f departs from slot s of airport p, 0 otherwise

DY_{fps}: 1 if flight f departs after the end time of slot s in airport p, 0 if flight f departs before the start time of slot s in airport p

$IFC_{ff'}$: 1 if two consecutive flights f and f' can be connected in an itinerary.

ITD_i: the actual delay time of itinerary i

itc_i: 1 of itinerary i is covered, 0 otherwise

T_f: a continuous variable denoting the departure time of a flight, min

W_{fa}: 1 if flight f served by aircraft a, 0 otherwise.

WF_{fa}: 1 if flight f is the first flight in the rotation of aircraft a, 0 otherwise

$X_{ff'}$: 1 if flight f precedes flight f' in time, 0 otherwise

Parameters:

AP_f: the arrival airport of f

$ACAP_{ps}$: the arrival capacity at slot s of airport p

CAP_{ac}: the seating capacity of class c in aircraft a

$DCAP_{ps}$: the departure capacity at slot s of airport p

DP_f: the departure airport of f

M: a large number whose value varies with constraints

FT_f: the flying time of flight f, min

MSS: the maximum length to delay a flight to wait for another flight

MGT: the maximum allowed difference between the original departure and arrival times of two potentially connected flights

MFD: the maximum allowed delaying time of a flight f away from its original scheduled departure time

MFT_a: the maximum flying time before maintenance of aircraft a

MIT: the minimum connection time between two consecutive flights in an itinerary

MNT_a: the minimum turn-round time between consecutive flights served by aircraft a

MST_a: the minimum transit time between multi-leg flights served by aircraft a

PCT_i: the number of passengers booking itinerary i

PCL_{if}: the cabin class of passengers booking flight f in itinerary i.

SE_{ps}: the end time of slot s in airport p, min

SS_{ps}: the start time of slot s in airport p, min

sta_f: the original scheduled arrival time of f

std_f: the original scheduled departure time of f

3.1.1 Mathematical formulation

Each flight f should either be assigned to an aircraft or be cancelled.

$$\sum_{a \in A_f} W_{fa} = 1 - CX_f \ , \ \forall f \in F_r, \forall f \notin FXN \qquad (5)$$

This is the cover constraint. For nodes with fixed registration, we simply fix their aircraft as:

$$W_{f,AFX_f} = 1, \ \forall f \in FXN$$

Nodes with fixed registration include flights with original delay disruption, start nodes, maintenance nodes and aircraft unavailable nodes. Although it can only take values 0 or 1, W_{fa} is defined as a continuous variable because it will be forced to be integral by other binary variables.

Flight sequence

A non-cancelled flight *f* is either the first flight in the rotation of aircraft *a* or it has a predecessor:

$$\sum_{f' \in FP_f} X_{f'f} + \sum_{a \in A_f} WF_{fa} = 1 - CX_f \ , \ \forall f \in F_r, f \notin FSN, f \notin UAN, f \notin FM \quad (6)$$

If a flight is a start node, it is defined as the first flight of an aircraft's rotation within *RP*:

$$WF_{f,AFX_f} = 1, \ \forall f \in FSN$$

A flight *f* cannot have more than one successor:

$$\sum_{f' \in FS_f} X_{ff'} \leq 1 - CX_f \ , \ \forall f \in F_r \quad (7)$$

FS_f denotes feasible successors of flight *f*. A flight *f′* can be a successor of flight *f* provided the following requirements are all satisfied:

1) There exists at least one aircraft which can fly both *f′* and *f*. i.e., $f' \in FC_f$

2) *f* is not an ending node and *f′* is not a start node

3) the destination airport of flight *f* is the origin airport of flight *f′*

4) the family of aircraft scheduled to fly flights *f* and *f′* are the same

5) $std_{f'} + MSS > sta_f + MNT_{AFX_f}$. When disruption happens, a better option might be to delay *f′* (originally earlier than *f*) until the aircraft flying *f* becomes available. *MSS* is the maximum length *f′* can be delayed to wait for *f*.

6) $std_{f'} - sta_f < MGT$. We can create a connection between *f* and *f′* if their original departure and arrival time difference is less than *MGT*.

Depending on the nature of a problem, other rules (Lever 1996) may be added to restrict the possible connections among flights so as to reduce the number of variables.

Two consecutive flights should be served by the same aircraft:

$$W_{f'a} \le W_{fa} + 1 - (X_{ff'} + X_{f'f}),\ \forall f \in F_r,\ \forall f' \in FS_f,\ \forall a \in AC_{ff'} \quad (8)$$

$$(X_{ff'} + X_{f'f}) + \sum_{a \in ANC_{f'f}} W_{f'a} \le 1,\ \forall f \in F_r,\ \forall f' \in FS_f \quad (9)$$

Constraints (8) and (9) avoid defining a tri-index variable $X_{ff'a}$ and thus reduce the number of binary variables significantly. A simple example is used to illustrate the definition of constraints (8) and (9) in Appendix I.

Multi-leg flights are consecutive and should be assigned to the same aircraft:

$$X_{f'f} = 1 - CX_f\ ,\ \forall (f', f) \in ML_{f'f} \quad (10)$$

Flight departure times

A flight f may be delayed to certain extent during *RP* for better resource allocation. However, it cannot depart earlier than its original departure time. i.e.,

$$T_f \ge std_f,\ \ \forall f \in F_r$$

T_f should also be less than its maximum delaying time:

$$T_f \le std_f + MFD,\ \ \forall f \in F_r$$

Turn-time constraints for consecutive flights:

$$T_{f'} \ge T_f + FT_f + MNT_{AFX_f} - M(1 - X_{ff'}) - M * CX_f,\ \ \forall f \in F_r,\ \forall f' \in FS_f \quad (11)$$

If flight f' follows f and not cancelled, then flight f' should depart after the arrival time of flight f, T_f+FT_f, plus the minimum connection time, MNT_{AFX_f}. The minimum connection time is the turn-round time of the aircraft flying flight f, i.e., the minimum time needed to prepare (passenger boarding, cabin cleaning crew changing etc.) for the subsequent flight. If f' and f corresponds to a multi-leg flight, MNT_{AFX_f} should be replaced by the transit time of the aircraft, MST_{AFX_f} , which is usually shorter than turn-round time. If f or f' is a maintenance node, aircraft unavailable node, f' is an end node or f is an artificial start node created for aircraft with empty original rotation, *f* and/or f' are not real flights but represent a particular connection to an airport, a time point, an ending position or a start position, MNT_{AFX_f} is zero.

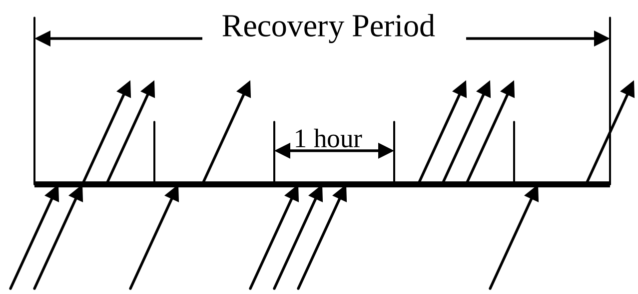

Figure 1 departures and arrivals at slots within *RP*

Slot capacities

As a consequence of rapid growth in air passenger traffic, major airports are struggling to cope with growing demand. While airport capacity can be increased by expanding the infrastructure, this is a major capital investment and it takes a long time. Consequently optimal usage of existing airport capacity becomes essential. One of the major components of airport capacity is its departure and arrival capacity, often termed "slot capacity", which correspond to the maximum number of aircraft take-offs/landings within a time interval. These maxima vary by day and by time within a day. In *the challenge*, *RP* is divided into one-hour intervals or slots.

Constraints (12) to (17) are used to define the number of departures and arrivals at airport slots.

$$T_f \geq SS_{ps} - M(1 - DX_{fps}), \quad \forall f \in F_r,\ p = DP_f, \forall s \in SL_P \qquad (12a)$$

$$T_f \leq SE_{ps} + M(1 - DX_{fps}), \quad \forall f \in F_r,\ p = DP_f, \forall s \in SL_P \qquad (12b)$$

When T_f is less than SS_{ps}, constraint (12a) forces DX_{fps} to be zero because otherwise (12a) is violated. Constraint (12b) is trivially satisfied when DX_{fps} is zero and T_f less than SS_{ps}. When T_f is bigger than SE_{ps}, constraint (12b) forces DX_{fps} to be zero because otherwise (12b) is violated. Constraint (12a) is trivially satisfied when DX_{fps} is zero and T_f bigger than SE_{ps}. Thus, DX_{fps} can take value of 1 only within slot *s*. Note that constraints (12a) and (12b) are defined only on slots whose capacities are greater than zero. Similarly, constraint (13a) and (13b) ensure that AX_{fps} can take value of 1 only within slot *s*.

$$T_f + FT_f \geq SS_{ps} - M(1 - AX_{fps}), \quad \forall f \in F_r,\ p = AP_f, \forall s \in SL_P \qquad (13a)$$

$$T_f + FT_f \le SE_{ps} + M(1 - AX_{fps}), \quad \forall f \in F_r,\ p = AP_f, \forall s \in SL_P \tag{13b}$$

Note that maintenance nodes, aircraft unavailable nodes, and end nodes do not occupy actual airport depart/arrival capacity and hence should not be included in constraints (12) to (13). For slots with zero departure or arrival capacity, no flight can depart from or arrive at it. To tighten the search space, constraints (14a) and (14b) are used to prevent flight *f* from departing from these slots:

$$T_f \le SS_{ps} + M * DY_{fps} + M * CX_f, \quad \forall f \in F_r,\ p = DP_f, \forall s \in SL_P \tag{14a}$$

$$T_f \ge SE_{ps} - M(1 - DY_{fps}) - M * CX_f, \quad \forall f \in F_r,\ p = DP_f, \forall s \in SL_P \tag{14b}$$

(14a) forces DY_{fps} be 1 and (14b) forces DY_{fps} be 0 which leads to a conflict if flight *f* departs from within the slot with zero departure capacity. Constraints (14a) and (14b) are trivial if flight *f* is cancelled. Similarly, constraints (15a) and (15b) are used to prevent flight *f* from arriving at slots with zero arrival capacity:

$$T_f + FT_f \le SS_{ps} + M * AY_{fps} + M * CX_f, \quad \forall f \in F_r,\ p = AP_f, \forall s \in SL_P \tag{15a}$$

$$T_f + FT_f \ge SE_{ps} - M(1 - AY_{fps}) - M * CX_f, \quad \forall f \in F_r,\ p = AP_f, \forall s \in SL_P \tag{15b}$$

Because we allow each flight to be delayed continuously, it is possible for *f* to depart from/arrive at any slot after its original departure/arrival time. To reduce the number of slots involved, we define *MFD* to be the maximum delaying time of flight *f*. Only a slot whose SS_{ps} is less than the original departure/arrival time of *f* plus *MFD* is considered for flight *f* in constraints (12) to (13). We further define the number of flights which could possibly depart from/arrive at a slot as the demand of the slot.

The flights in progress at the start of the *RP* cannot be changed: they arrive at their scheduled times. As specified in *the challenge*, on arrival, each flight takes one landing slot (*ROADEF* 2009). Thus we reduce the available capacity in the corresponding slot at which these flights arrive. These particular flights are then excluded from constraints (12) to (13) because their slot occupancies are known and are deduced from the slot capacity a priori.

Slot demand is the maximum possible number of departures/arrivals during that slot. For slots whose demand is greater than their capacity, the following capacity constraints, (16a) and (16b), are imposed:

$$\sum_{f \in F_r} DX_{fps} \le DCAP_{ps}, \quad \forall p \in P, \forall s \in SL_p \tag{16a}$$

$$\sum_{f \in F_r} AX_{fps} \leq ACAP_{ps}, \quad \forall p \in P, \forall s \in SL_p \tag{16b}$$

The left hand side of (16a)/(16b) counts the number of departures/arrivals at slot *s* of airport *p*. Constraints (17a)/(17b) require normal flights within *RP* to depart from/arrive at one slot unless cancelled. These two constraints are not imposed on start nodes, maintenance nodes, aircraft unavailable nodes or end nodes.

$$\sum_{s \in SL_p} DX_{fps} = 1 - CX_f, \quad \forall f \in F_r,\ p = DP_f \tag{17a}$$

$$\sum_{s \in SL_p} AX_{fps} = 1 - CX_f, \quad \forall f \in F_r,\ p = AP_f \tag{17b}$$

Constraints (5) to (17) complete the aircraft routing model. Constraints (18) to (19) are added below to handle different types of disruption.

Handling aircraft unavailability.

During certain periods of time, an aircraft may out of service for unspecified reasons. No flight may be assigned to that aircraft within an unavailable period. As the previous flights to which the aircraft has been assigned are not specified a priori, the airport where it becomes unavailable is also unknown. Accordingly, a dummy node is created at each airport for each unavailable aircraft period in our connection network. Constraints (18a) to (18c) are used to ensure the unavailable aircraft stays at the destination airport of its last flight before the unavailable period and no flight is assigned to it during the unavailable period.

$$\sum_{f \in FUA_a} \sum_{f' \in FP_f} X_{f'f} = 1, \ \forall a \in UA \tag{18a}$$

Ensuring the unavailable node of an aircraft has a predecessor in constraint (18a) guarantees that the aircraft unavailable period is respected. Constraint (18b) ensures that the unavailable aircraft has at most one successor after its unavailable period to reflect the fact that the unavailable aircraft may continue to serve any flight after its unavailable period.

$$\sum_{f \in FUA_a} \sum_{f' \in FS_f} X_{ff'} \leq 1, \ \forall a \in UA \tag{18b}$$

Constraint (18c) ensures that the predecessor and the successor of the unavailable aircraft connect to the same dummy node.

$$X_{f'f} + \sum_{\substack{m \in FUA_a \\ m \neq f}} \sum_{n \in FS_m} X_{mn} \leq 1 \,, \; \forall a \in UA, \forall f \in FUA_a, \forall f' \in FP_f \tag{18c}$$

Constraint (18c) says that if a predecessor is connecting to a dummy node *f*, i.e., $X_{f'f}$ =1, then no successor is allowed to connect to any other dummy nodes except *f* (i.e., X_{mn} =0, $\forall m \in FUA_a \cap m \neq f$).

Handling aircraft maintenance

$$\sum_{f' \in FP_f} X_{f'f} = 1 \,, \; \forall f \in FM \tag{19a}$$

$$\sum_{f' \in FS_f} X_{ff'} \leq 1 \,, \; \forall f \in FM \tag{19b}$$

Ensuring the maintenance node of an aircraft has a predecessor in constraint (19a) guarantees that the aircraft maintenance requirement is satisfied. An aircraft may continue to serve flights after its maintenance period. This is done by constraint (19b) ensuring that the maintenance node has at most one successor after its maintenance period. Constraint (19c) ensures the maximum flying time of an aircraft before maintenance is not exceeded.

$$\sum_{f' \in FA_{AFX_f}} \mathrm{W}_{f',AFX_f} FT_{f'} + \sum_{f' \in FXR_{AFX_f}} FT_{f'} \leq MFT_{AFX_f} \,, \; \forall f \in FM \tag{19c}$$

In constraint (19c), *f* is a maintenance node representing an aircraft which requires maintenance within *RP*. AFX_f is the fixed aircraft registration of *f*. MFT_{AFX_f} is the maximum flying time of aircraft AFX_f before maintenance. FA_{AFX_f} represents all flights that aircraft AFX_f can serve. The first/second term in the left hand side of (19c) counts the total flying time of flights served by AFX_f within/before *RP* respectively.

Handling itineraries

In *the challenge*, if an itinerary is composed of several legs, the itinerary class is assumed to be the highest of all the booking cabin classes on the different legs of the itinerary (normally they are all the same class). The itinerary type is defined as the type of its longest leg (intercontinental > continental > domestic)(*ROADEF* 2009).

Two consecutive flights on a passenger's itinerary must be separated by a minimum connection time. If any flights are delayed or cancelled after disruptions, their departure/arrival times may violate the minimum connection time requirements between flights in an itinerary. We identify and cancel these itineraries a priori. For a normal itinerary which is not affected by disruptions, its status is determined by the status of the flights which make up this itinerary. Constraints (20) to (23) are used to determine the status of a normal itinerary. A similar approach and constraints have been proposed by previous researchers (Rosenberger et al. 2003, Bratu et al. 2006).

Constraint (20) cancels the whole itinerary if any flight f in the itinerary i is cancelled. We will recover an itinerary with cancelled flights in the passenger re-accommodation model.

$$itc_i \leq 1 - CX_f, \quad \forall i \in ITN, \forall f \in book_i \tag{20}$$

If the connection time between two consecutive flights on an itinerary is less than the minimum connection time, constraint (21) sets their connection status ($IFC_{ff'}$) to be 0.

$$T_f - (T_{f'} + FT_{f'}) \geq MIT - M(1 - IFC_{f'f}), \quad \forall i \in ITN, \forall (f', f) \in book_i \tag{21}$$

In (21), MIT is the minimum connection time between two consecutive flights in an itinerary. In *the challenge*, MIT is assumed to be 30 minutes. Flight f' is the predecessor of f in itinerary i. If any two consecutive flights f and f' in an itinerary cannot be connected, the whole itinerary is then cancelled through constraint (22):

$$itc_i \leq 1 - IFC_{ff'}, \quad \forall i \in ITN, \forall (f, f') \in book_i, \tag{22}$$

Constraint (23) calculates the actual delay time of an itinerary. Note that f should be the last flight in itinerary i in constraint (23).

$$ITD_i \geq T_f - std_f - M(1 - itc_i), \quad \forall i \in ITN, f \in book_i \tag{23}$$

With the status of itinerary i determined, we can add a constraint to ensure that the cabin capacity of each aircraft is not exceeded.

$$\sum_{a \in A_f} W_{fa} CAP_{ac} \geq \sum_{\substack{i \in ITN \\ \cap f \in book_i \\ \cap c = PCL_{if}}} itc_i PCT_i, \ \forall f \in F_r, \ \forall c \in C \tag{24}$$

Constraint (24) says that, the total number of passengers booking itineraries which include flight *f* as one of their bookings, with the passenger class of flight *f* being *c*, should be less than the seating capacity of class *c* in the aircraft *a* which serves flight *f*.

3.2 Passenger re-accommodation

When disruption occurs, large airlines usually solve the problem in a sequential fashion: disruptions relevant to aircraft are first solved with some flights are re-assigned to other aircraft. This is followed by crew recovery, and finally the impact on passengers is evaluated (Jens Clausen, 2005). A second phase then reoptimises passenger itineraries based on the flight schedules determined in phase one. In the disruption management literature, passengers are usually given a low priority (Kohl et al. 2007).

For passenger disruption, a multi-commodity network flow model is used, with each passenger itinerary represented as a separate commodity, flowing through arcs representing each cabin class in each flight. The objective is then to maximize the value of the itineraries flowing through the flight network, within the given flight capacity and passenger demand.

- The cost of passenger delays. The cost depends on the delay at the final destination of the passenger. This is not the traditional way to measure delay in airlines, but we find this is a more relevant measure than the delay of the aircraft compared to schedule. The delay cost calculation also takes into consideration the commercial value of the passenger—for example based on the booked fare class and frequent flyer information.
- It is a subjective issue how to derive a formula for the cost of passenger delays, but it is well established that there is a long-term cost associated with delaying passengers.
- The cost of passenger off loads. There may be several real costs as well as loss of goodwill associated with offloading a booked passenger.
- The cost of meals and hotel accommodation for severely disrupted passengers. In many cases the airline is required to or volunteers to provide passengers with meals and accommodation in case of disruptions.
- The cost of passenger upgrades and downgrades. These costs are partly real costs for upgraded catering and downgrade compensation, but there is also loss of goodwill costs associated with downgrades.

4. Solution method

The models were implemented in Xpress Mosel 3.0 on a PC with a Intel(R) Core(TM)2 Duo processor and 2GB of RAM running Windows XP Professional SP3 (32bits). *The Challenge* provides two sets of problem instances: set A and set B, each with 10 instances of different sizes. Set A instances involve up to 1000 flights, 80 aircraft, 35 airports and 4000 itineraries. Set B instances involve up to 3000 flights, 250 aircraft, 44 airports and 11,000 itineraries. All test instances are provided by Amadeus (*ROADEF* 2009). For set A instances, optimal solutions to some instances can be obtained quickly. Two different objectives were used to test the performance of our models. The first objective was the total flight delays plus the total flight cancellation penalty as shown in (obj1) below:

$$ACR\text{: Min } TC = \sum_{f \in F_r} (T_f - std_f) + \sum_{\substack{f \in F_r, f \notin FSN, \\ f \notin UAN, f \notin FM \\ f \notin END}} C_f^{op} FT_f CX_f \qquad \text{(obj1)}$$

As there is no direct data for flight cancellation penalty, the flight operation cost is used instead. That is, the higher the flight operation cost, the higher the flight cancellation penalty. As C_f^{op} is only used for discouraging the cancellation of flights in the aircraft routing model, its exact value is not important.

The second objective is an approximation of the objective of *the Challenge*.

$$\begin{aligned} AIP\text{: Min } TC = & \sum_{f \in Created\_f} C_f^{op} FT_f (1 - CX_f) + \\ & \sum_{i \in ITN} (C_i^{cancel\_legal} + C_i^{cancel\_pax})(1 - itc_i) PCT_i + \\ & \sum_{i \in ITN} C_i^{delay\_pax} PCT_i ITD_i + \\ & \sum_{f \in END} NbModel * CX_f \end{aligned} \qquad \text{(obj2)}$$

$C_i^{cancel\_legal}$ is the cost of the cancellation of a passenger in itinerary *i*. It is the ticket price plus financial compensation. $C_i^{delay\_pax}$ and $C_i^{cancel\_pax}$ are the penalty for delaying/canceling a passenger in an itinerary *i*. They depends on the itinerary type and the nature of the itinerary (inbound or outbound), respectively. The first model, *ACR*, involves (obj1) and constraints (5) to (19). The second model, *AIP*, requires the status and delays of itineraries. Thus *AIP* involves (obj1) and constraints (5) to (24).

Most of the set A instances in *the challenge* can be solved in reasonably short time while set B instances are far too large to be solved as a single MILP model. The solvers we tried (academic and commercial) even failed to load the problem into the memory. A decomposition algorithm thus becomes imperative.

4.1 Decomposition by relaxing sub-problems as LP

In this paper we propose to decompose the problem into sets of flights. We first explore the modeling consequences of such a decomposition. Equations (1)-(4) above deal with costs, which are discussed later. Constraint (5) requires an aircraft to be assigned to each flight, so each subproblem must model not only a fixed set of flights but also a fixed set of aircraft which can be assigned to those flights. For global consistency, the subproblems must partition both the flights and aircraft (i.e. no flight or aircraft can belong to more than one subproblem). By fixing a priori the set of aircraft associated with each subproblem we restrict possible assignments. Constraints (6)-(10) constrain the predecessor and successor of each flight - thus each subproblem must include all the flights flown by an aircraft. Since all the flights in a subproblem must anyway be flown by aircraft associated with the subproblem the predecessor and successor of each flight must anyway belong to the same subproblem. Constraint (11) is on flight departure times, and each instance of the constraint belongs naturally to the subproblem of its flight.

Constraints (12)-(17) restrict the number of flights arriving and departing from an airport in each time slot. Assuming flights from different subproblems could use the same time slot, this constraint must span multiple subproblems. The approach adopted in this paper is to divide the slot capacity between the subproblems a priori so that constraints (12)-(17) employ the slot capacity allocated to the relevant subproblem, and can then fit neatly into the decomposition. The challenge of the decomposition approach is how to distribute the limited slot capacity to different subproblems. If one could assign the total capacity of all slots of all airports to subproblems optimally, then assuming the optimal aircraft and flights belong to the same subproblem, the optimal solution to the whole problem is obtained.

Constraints (18)-(19) are aircraft constraints, which naturally fit our decomposition. Constraints (20-(24) constrain passenger itineraries, and they will be discussed later.

There are several potential ways of decomposing the flights, including:

1). Decompose by aircraft type

2). Decompose by subproblem size,

A good decomposition method should significantly shorten the solution time without sacrificing too much of solution quality. In this paper, the above methods are combined: Add all the flights assigned to one aircraft type into a single subproblem, until a given maximum subproblem size is exceeded. In that case, add the rest of the flights assigned to the aircraft type into the next subproblem. Fig. 2 illustrates the decomposition approach.

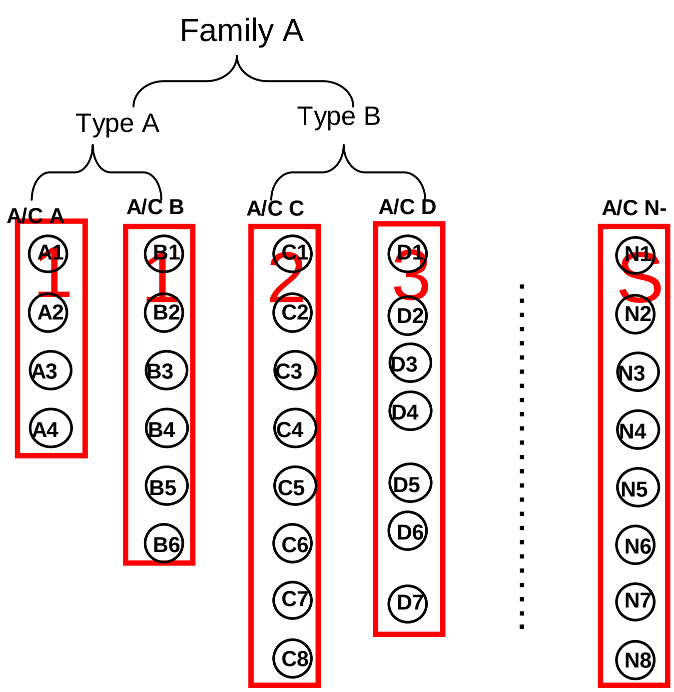

Fig. 2 the decomposition approach

We now return to the cost expression (equations (1)-(4)) and the itinerary constraints (20)-(24).

We propose to handle these through a second form of problem decomposition. This decomposition first handles flights and aircraft (itself using the master/subproblem decomposition introduced above), and then as a second subproblem passenger itineraries are mapped onto the solution of the first subproblem.

To ensure the solution to the first subproblem (assigning aircraft to flights) is suitable for the second subproblem (assigning passengers to flights) the cost function associated with the first subproblem must be carefully designed.

In the following subsections the cost function associated with the first problem is designed on the assumption that passengers will be assigned to their originally scheduled flights. Thus the aircraft assigned to a flight is constrained to have sufficient capacity to meet the passenger demand (and if it is too small there is a cost associated with the passenger "spillage").

Solving the aircraft to flight assignment sub-problem

The allocation of slot capacity between the subproblems is handled as a *master problem*, and an iterative process is used to refine the capacities allocated to the subproblems.

That is, in the master problem model, the only global constraint is the airport slot capacity constraint (16).

The master problem and subproblem models can be illustrated as:

$$\text{Master Problem:} \quad \min \ \text{total cost}$$

$$\text{s.t.} \quad \sum_{i=1\sim n} \text{Usage of slot s} \leq \text{Total Capacity of slot s}$$

and

$$\text{Sub-problems } 1\sim n\text{:} \begin{cases} \text{slot s usage of SubP}_1 \leq \text{ slot s capacity of SubP}_1 \\ \qquad\qquad \ldots\ldots\ldots\ldots\ldots\ldots \\ \text{slot s usage of SubP}_n \leq \text{ slot s capacity of SubP}_n \end{cases}$$

Naïve master/sub-problem algorithm

A naïve implementation of the above master/sub-problem decomposition iterates between the master and subproblem using shadow prices returned from the sub-problems to guide the next slot decomposition to be returned from the master problem. In order to extract shadow prices the sub-problems are handled using linear programming.

Specifically, this naïve approach solves each sub-problem as a relaxed LP – without enforcing integrality on the number of aircraft arriving/departing in each slot - and elicits slots with negative shadow prices. A slot with negative shadow price indicates that increasing its slot capacity can reduce the total cost. Thus, allocating more slot capacities to these slots could be profitable. The algorithm is illustrated in Fig. 3.

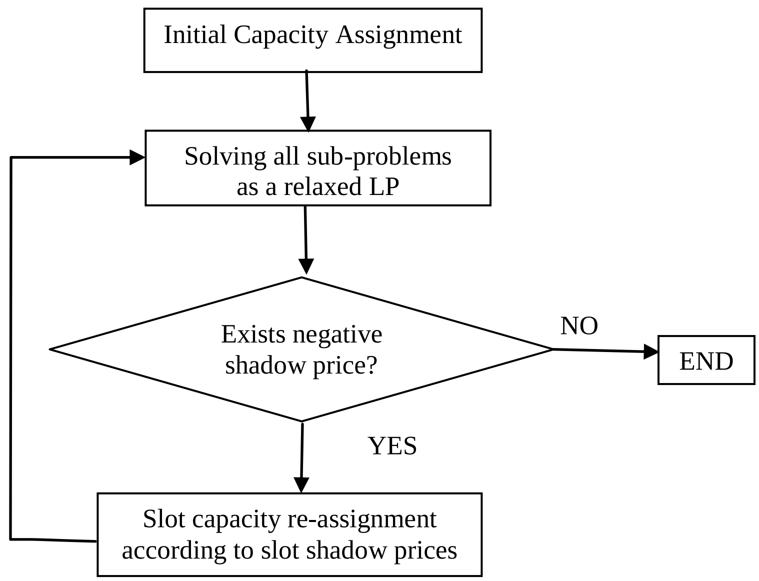

Fig 3. The Relaxed LP algorithm

To evaluate the na ïve approach we compare it to an approach that, for small problem instances, can return a proven optimal solution. For this evaluation we use the set A instances from *the Challenge* which are all relatively small instances that can indeed be solved to optimality by a single MILP. The following table presents the results.

Instances	Solving as a single MILP Gap, %	Solving as a single MILP Obj. Val.	Decomposition algorithm Obj. Val.
A01	0.00%	839	3464
A02	0.00%	734.89	985
A03	0.00%	1329	101751
A04	0.40%	776477	6.30E+06
A05	1.70%	4.04E+07	7.77E+07
A06	0.00%	839	5455
A07	0.00%	734.89	985
A08	0.00%	1329	101932
A09	0.10%	775692	3.50E+06
A10	0.40%	4.04E+07	7.80E+07

Table 1. Results for na ïve master/sub-problem algorithm

It can be seen that the performance of the na ïve algorithm sometimes falls well short of optimality. The total cost of flight delays in A01 is much higher than the optimal solution. Inappropriate allocations of slot capacities to sub-problems lead to flights cancellations in some instances (e.g. A08).

Analysis of Relaxed LP algorithm

To analysis algorithm 1, we solve a set A instance (A03) to optimality and analyse the solution. For this instance, the optimal capacity is achievable by the following swaps from the initial assignment. The following table identifies the initial distribution of the total slot capacity that will in the optimal solution be required by sub-problem 1.

Port	slot id		S1	S2	S3	S4
BOD	2	initial assignment	1	0	1	0
		Optimal assignment	2	0	0	0
LYS	6	initial assignment	0	1	0	1
		Optimal assignment	1	1	0	0
SXB	4	initial assignment	0	1	0	0
		Optimal assignment	1	0	0	0
MPL	8	initial assignment	1	0	1	0
		Optimal assignment	2	0	0	0
NCE	2	initial assignment	0	0	1	0
		Optimal assignment	1	0	0	0
NCE	7	initial assignment	0	1	0	0
		Optimal assignment	1	0	0	0

Table 2. The optimal swap of A03.

In Table 2, we can see that, at slot 2 of airport BOD, sub-problems S1 and S3 are initially assigned a slot capacity of 1. However, the optimal assignment moves the S3 slot capacity to sub-problem S1. However, from the relaxed LP, the slots with negative shadow price are:

Port	slot id
LIG	11~15
LYS	6~13
SXB	3~13

Table 3. slots with negative shadow price

Slot id 2 of BOD does *not* have a negative shadow price in S1, so the master problem is not driven to increase S1's slot capacity. Moreover LIG should not be swapped; and MPL, NCE should be swapped but also fail to have a negative shadow price in S1. We conclude that LP relaxation of the sub-problems is too loose to provide enough information to drive the master problem towards an optimal distribution of slot capaciaties.

Enhanced master/sub-problem algorithm

To ensure that the required slot capacity needed for an optimal solution is returned from the sub-problem, spare slot capacity is made available for each slot in the form of an additional variable *insdummy*, and this spare capacity is minimized by adding a weighted cost *A*insdummy* to the cost function (similar to a Lagrangean relaxation). Assuming the weight is small enough this allows an optimal solution to the sub-problem to be returned assuming there was enough slot capacity. For slot capacity which is not needed, the solution is prevented from gratuitously keeping it by adding a slack variable *desdummy* whose value is maximized by adding the weighted expression *-A*desdummy* to the cost function. In sum, the slot capacity constraint of each sub-problem is modified as:

$$\sum_{f} \mathrm{Dept}_{f,p,s} \leq \mathrm{TCAP}_{p,s} + \mathrm{insdummy}_{p,s} - \mathrm{desdummy}_{p,s}, \forall p,s$$

$$obj.: \min \ A^* \sum_{p,s} \mathrm{insdummy}_{p,s} - A^* \sum_{p,s} \mathrm{desdummy}_{p,s} + \text{other costs}$$

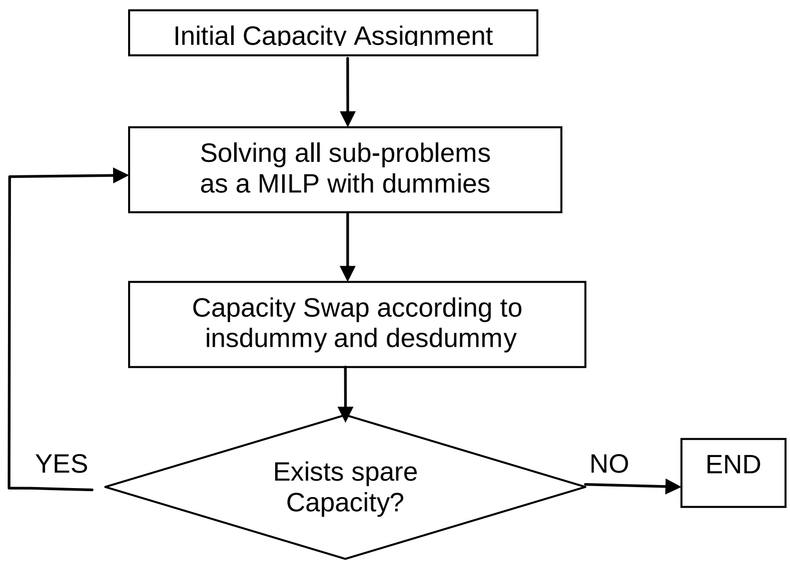

Fig 4. Algorithm 2

The Master problem, instead of changing the weights as in the Lagrangian approach, simply moves slot capacity from the sub-problems which use less than their maximum capacity to sub-problems that use more. The behaviour of this algorithm is quite robust to different weights A, but for the results reported in the following table we used a weight of A=2.

As before we assess the quality of the enhanced algorithm by evaluating it on small (ROADEF set A) problem instances, which can be solved to optimality by a single MILP.

Instances	Solving as a single MILP Gap, %	Solving as a single MILP Obj. Val.	Decomposition algorithm Obj. Val.
A01	0.00%	839	839
A02	0.00%	734.89	735
A03	0.00%	1329	1374
A04	0.40%	776477	1.55e+006
A05	1.70%	4.04E+07	4.81E+07
A06	0.00%	839	839
A07	0.00%	734.89	735
A08	0.00%	1329	1374
A09	0.10%	775692	1.73991e+006
A10	0.40%	4.04E+07	4.81E+07

Table 4. Results for enhanced master/sub-problem algorithm:

This table shows that the results for many instances are close to the optimal value returned from the single MILP algorithm.

4.2 results with Passenger re-accommodation

Although the consequences of operations on passenger are taking into account during the disruption, passenger disruptions rarely drive decision-making (Bratu 2006). We show that, however, passenger re-accommodation has huge impact on the total cost of an airline.

Results of solving as a single MILP with/without passenger recovery and using (obj2):

Instances	solving as a single MILP without passenger recovery	solving as a single MILP with passenger recovery	Gap*, %
A01	1,329,245.4	48,405.7	96.4
A02	994,889.8	195,298.9	80.4
A03	1,748,617.8	107,197.3	93.9
A04	1,039,954.7	123,508.0	88.1
A05	24,670,013.7	12,761,434.2	48.3
A06	1,163,219.8	57,027.6	95.1
A07	737,000.6	231,464.4	68.6
A08	2,000,623.8	262,346.9	86.9
A09	2,009,636.9	274,548.9	86.3
A10	32,063,289.7	26,479,106.5	17.4

Gap*: (without PaxRev – withPaxRev)/without PaxRev*100

Table 5. Comparison between with and without passenger recovery

In the above table, all results have passed *the challenge* Solutionchecker and the total costs are obtained from *the challenge* Costchecker. We can see that the total cost is reduced significantly after the passenger re-accommodation.

Results of solving as a single MILP comparing the total cost with the best Roadef challenge team:

Instances	solving as a single MILP with passenger recovery	The Roadef Challenge Rank No. 1 team	Gap, %
A01	48,405.7	29,891.75	38.2
A02	195,298.9	116,431.70	40.4
A03	107,197.3	202,358.10	-88.8
A04	123,508.0	139,747.10	-13.1
A05	12,761,434.2	3,717,376.35	70.9
A06	57,027.6	44,305.05	22.3
A07	231,464.4	202,247.75	12.6
A08	262,346.9	659,572	-151.4
A09	274,548.9	215,482.35	21.5
A10	26,479,106.5	7,210,166.90	72.8

Table 6. Comparing with the best team

We can see that, for some instances, we can find much better solution than the best team of *the challenge*. It shows that the results in this paper are competitive with the best results.

Results of the decomposition algorithm comparing the total cost with the best Roadef challenge team:

Instances	decomposition algorithm	The Roadef Challenge Rank No. 1 team	Gap, %
A01	81,880.25	29,891.75	63.5
A02	203,199.75	116,431.70	42.7
A03	152,920.45	202,358.10	-32.3
A04	476,459.20	139,747.10	70.7
A05	14,900,586.00	3,717,376.35	75.1
A06	55,016.95	44,305.05	19.5
A07	301,573.85	202,247.75	32.9
A08	293,613.95	659,572	-124.6
A09	1,258,329.90	215,482.35	82.9
A10	30,125,628.20	7,210,166.90	76.1

Table 7. Results of the decomposition algorithm

We can see that, for some instances, we can find much better solution than the best team of *the challenge*. It shows that the results in this paper are competitive with the best results.

Results of the decomposition algorithm for set B and set X instances are listed in Table 8. Note that no solution can be obtained by solving set B instances as a single MILP.

Instances	Total cost, $	Solution time, seconds
B01	3463159.2	848.3
B02	6394073.6	863.4
B03	3508343.0	862.9
B04	5280301.4	859.7
B05	44508280.9	1353.6
B06	8677772.9	930.5
B07	13444446.4	1024.9
B08	11024512.7	889.0
B09	10987895.1	890.3
B10	88613615.6	1313.3
XA01	339186.7	26.7
XA02	16771953.6	702.1
XA03	561159.3	588.9
XA04	30048893.4	657.9
XB01	4842427.6	924.9
XB02	48550553.3	1141.5
XB03	9917899.1	894.8
XB04	90579570.9	1022.5

Table 8. Results of the decomposition algorithm for set B and set X instances

5. Conclusion and future work

In this paper, a new continuous time aircraft routing model is developed which can minimize aircraft delay cost accurately and efficiently handle all types of disruptions encountered in ***ROADEF***. Applying a new decomposition algorithm, near optimal solutions for aircraft routing can be obtained. For future work, we propose to: 1). Improve the current passenger re-accommodation algorithm; 2). Integrate passenger re-accommodation with aircraft routing; 3). Improve the decomposition algorithm for set B instances; 4). Apply LNS (Large Neighbourhood Search) for better decomposition.

References

[1] *ROADEF 2009* Challenge, http://challenge.roadef.org/2009/index.en.htm

[2] M. Palpant, M. Boudia, C.-A. Robelin, S. Gabteni, F. Laburthe, ROADEF 2009 Challenge: Disruption Management for Commercial Aviation, 2008. http://challenge.roadef.org/2009/challenge_en.pdf

[3] Fuel shortage hits Sydney airport, Sept. 26, 2003, http://news.bbc.co.uk/2/hi/asia-pacific/3141568.stm

[4] Computer glitch hits US flights, Nov. 19 2009, http://news.bbc.co.uk/2/hi/americas/8368742.stm

[5] Shan Lan, John-Paul Clarke, Cynthia Barnhart Planning for Robust Airline Operations: Optimizing Aircraft Routings and Flight Departure Times to Minimize Passenger Disruptions Transportation Science 40(1), pp. 15–28, 2006

[6] Andrew J. Schaefer, Ellis L. Johnson, Anton J. Kleywegt, George L. Nemhauser Airline Crew Scheduling Under Uncertainty Transportation Science 2005, 39(3): 340-348

[7] Bratu, Stephane and Barnhart, Cynthia, Flight Operations Recovery: New Approaches Considering Passenger Recovery, Journal of Scheduling, Vol. 9:3, pp.279-298, 2006.

[8] Niklas Kohl, Allan Larsen, Jesper Larsen,_Alex Ross, Sergey Tiourin Airline disruption management—Perspectives, experiences and outlook. Journal of Air Transport Management 2007; 13:149–162

[9] Jay M. Rosenberger, Ellis L. Johnson, and George L. Nemhauser. Rerouting aircraft for airline recovery. *Transportation Science*, 37(4):408–421, 2003

[10] C. Barnhart, A. Cohn Airline Schedule Planning: Accomplishments and Opportunities. Manufacturing & Service Operations management 2004; 6(1):3-22

[11] Dusan Teodorovi ćc and Slobodan Guberini ćc. Optimal Dispatching Strategy on an Airline Network after a Schedule Pertubation. *European Journal of Operational Research*, 15:178–182, 1984.

[12] Dusan Teodorovi ćc and Goran Stojkovi ćc. Model for Operational Daily Airline Scheduling. *Transportation Planning and Technology*, 14:273–285, 1990.

[13] A. I. Z. Jarrah, G. Yu, N. Krishnamurthy, and A. Rakshit. A Decision Support Framwork for Airline Flight Cancellations and Delays. *Transportation Science*, 27:266–280, 1993.

[14] Kalyan T. Talluri. Swapping Applications in a Daily Airline Fleet Assignment. *Transportation Science*, 30:237–248, 1996.

[15] Lettovsky, L., 1997. Airline operations recovery: an optimization approach. Ph.D. Thesis, Georgia Institute of Technology, Atlanta.

[16] Jonathan F. Bard, Gang Yu, and Michael F. Argüello. Optimizing aircraft routings in response to groudings and delays. *IIE Transactions*, 33:931 – 947, 2001.

[17] Christopher A. Hane, Cynthia Barnhart, Ellis L. Johnson, Roy E. Marsten, George L. Nemhauser, Gabriele Sigismondi The fleet assignment problem: solving a large-scale integer program, Mathematical programming, 1995, 70 211-232

[18] Jonathan Lever, Resource Reallocation: a Pre-processing role for Constraint Programming, Proc., Practical Application of Constraint Technology (PACT'96), London, April, 1996, 169-178

[19] C. Barnhart, N. Boland, L.W. Clarke, E.L. Johnson, and G.L. Nemhauser. Flight string models for aircraft fleeting and routing. *Transportation Science*, 32:208 –220, 1998.

[20] Michael D. D. Clarke. The airline schedule recovery problem. Working paper, October 1997.

[21] Shangyao Yan and Chung-Gee Lin. Airline Scheduling for the Temporary Closure of Airports. *Transportation Science*, 31(1):72–82, 1997.

[22] Benjamin G. Thengvall, Gang Yu, and Jonathan F. Bard. Multiple fleet aircraft schedule recovery following hub closures. *Transportation Research Part A*, 35:289-308, 2001.

[23] Hui CW, Gupta A. A bi-index continuous-time mixed-integer linear programming model for single-stage batch scheduling with parallel units. Industrial and Engineering Chemistry Research, 2001,40(25):5960−5967.

[24] Amadeus S.A.S., Operations Research Division, http://www.amadeus.com/amadeus/amadeus.html

Appendix I.

We use a simple example to illustrate the definition of sets and the logic of constraints (8) and (9).

	a1	*a2*	*a3*	*a4*	*a5*	*a6*	*a7*	*a8*	*a9*
f1	○	○	○	○	○	○			
f2				○	○	○	○	○	○
f3								○	○
f4									○

Table A1. An illustrative example

There are 4 flights and 9 aircraft in Table A1. A circle in Table A1 indicates the flight at its row can be flown by the aircraft at its column. We then have A_{f1} ={*a1*,*a2*,…,*a6*}, A_{f2} ={*a4*,*a5*,…,*a9*}. FC_{f1}={*f2*} because there exists *a4* to *a6* which fly both *f1* and *f2*. FC_{f2}={*f1*,*f3*,*f4*}. $AC_{f1,f2}$={*a4*,*a5*,*a6*}. $ANC_{f1,f2}$={*a1*,*a2*,*a3*} because *a1* to *a3* can fly *f1* but not *f2*. $ANC_{f2,f1}$={*a7*,*a8*,*a9*}, FA_{a1}={*f1*}, FA_{a4}={*f1*,*f2*}.

Observe that constraints (8) and (9) are trivial for non-consecutive flights ($X_{ff'}+X_{f'f}$=0) and they reduce to

$$W_{f'a} \le W_{fa}, \quad \forall f \in F_r,\ \forall f' \in FS_f,\ \forall a \in AC_{ff'} \qquad (8')$$

$$\sum_{a \in ANC_{f'f}} W_{f'a} \le 0, \quad \forall f \in F_r,\ \forall f' \in FS_f \qquad (9')$$

for consecutive flights *f* and *f′* ($X_{ff'}+X_{f'f}$=1).

Assume *f1* and *f2* are consecutive flights in the illustrative example. We further assume that *f2* can follow *f1* according to the rules defined in section 3.1.1, i.e., FS_{f1}={*f2*}. Setting *f*=*f1*, *f′*=*f2* to (9′) lead to:

C1). $W_{f2,a} = 0, \forall a \in \{a7, a8, a9\}$.

This follows from (9′) and the fact that W_{fa} is positive. *C1*) disallow *f2* be assigned to aircraft which cannot fly *f1*. Thus, *f2* can only be assigned to aircraft which can fly both *f1* and *f2*, i.e. $\forall a \in \{a4, a5, a6\}$.

If *f1* can also follow *f2*, i.e., FS_{f1}={*f2*}, FS_{f2}={*f1*} (we ignore *f3*, *f4* for ease of exposition), applying *f1* and *f2* to (8′) respectively lead to $W_{f1,a} = W_{f2,a}, \forall a \in \{a4, a5, a6\}$. i.e., *f1* and *f2* are assigned to the same aircraft *a* which belongs to $AC_{f1,f2}$.

Now assume that *f2* can follow *f1*, but *f1* cannot follow *f2*, i.e., FS_{f1}={*f2*}, FS_{f2}=∅, FP_{f1}=∅, FP_{f2}={*f1*}. We have $X_{f1,f2}$=1 and $X_{f2,f1}$=0 because *f1* and *f2* are consecutive flights. This follows:

C2). CX_{f1}=0. This comes by applying *f1* to (7) and $X_{f1,f2}$=1.

C3). CX_{f2}=0. This comes by applying *f2* to (6) and $X_{f1,f2}$=1.

i.e., two flights are consecutive implies that both of them are not cancelled. From *C1*), *f2* can only be assigned to any aircraft *a* which belongs to $AC_{f1,f2}$. Let us assume *f2* is served by aircraft *a4* ($W_{f2,a4}$=1). It is easy to see that $W_{f1,a4}$=1 from (*8'*). Similar results can be derived for situation when *f1* can follow *f2* but *f2* cannot follow *f1*.