TRANSPORTATION SCIENCE
Articles in Advance, pp. 1–22
ISSN 0041-1655 (print) | ISSN 1526-5447 (online)

http://dx.doi.org/10.1287/trsc.2013.0463

The Recoverable Robust Tail Assignment Problem

Gary Froyland, Stephen J. Maher
School of Mathematics and Statistics, University of New South Wales, Sydney NSW 2052, Australia
{g.froyland@unsw.edu.au, stephen.maher@unsw.edu.au}

Cheng-Lung Wu
School of Aviation, University of New South Wales, Sydney NSW 2052, Australia, c.l.wu@unsw.edu.au

Schedule disruptions are commonplace in the airline industry with many flight-delaying events occurring each day. Recently there has been a focus on introducing robustness into airline planning stages to reduce the effect of these disruptions. We propose a recoverable robustness technique as an alternative to robust optimisation to reduce the effect of disruptions and the cost of recovery. We formulate the recoverable robust tail assignment problem (RRTAP) as a stochastic program, solved using column generation in the master and subproblems of the Benders' decomposition. We implement a two-phase algorithm for the Benders' decomposition and identify pareto-optimal cuts. The RRTAP includes costs due to flight delays, cancellation, and passenger rerouting, and the recovery stage includes cancellation, delay, and swapping options. To highlight the benefits of simultaneously solving planning and recovery problems in the RRTAP we compare our tail assignment solution against current approaches from the literature. Using airline data we demonstrate that by developing a better tail assignment plan via the RRTAP framework, one can reduce recovery costs in the event of a disruption.

Key words: robust airline optimisation; recovery; Benders' decomposition
History: Received: December 2011; revisions received: June 2012, December 2012; accepted: December 2012.
Published online in *Articles in Advance*.

1. Introduction

The airline planning process is generally performed with the expectation that operations will be executed as planned. Generally this is not the case, and delays and cancellation of flights are commonplace every day. In August 2011 the on-time performance of the airline industry of Australia averaged 83.7% for departures and 81.7% for arrivals, where on time is defined as flights departing or arriving within 15 minutes of the scheduled time (Bureau of Infrastructure, Transport and Regional Economics 2011). For the same period, Europe experienced an on-time performance of 82.48% (Central Office of Delay Analysis 2011) and the United States had 79.34% of all flights on time from reporting carriers (Research and Innovative Technology Administration 2011a). The operations of an airline are susceptible to outside influences such as weather, and in the long run a significant percentage of flights will be disrupted.

It is becoming more accepted by the airline industry that airline operations *will* be affected by some level of disruption and that this should be accounted for in the planning process. The considerable additional cost associated with any delay has spurred an interest in *robust* airline planning. These robust plans focus on incorporating features to improve operational performance in the presence of unforeseen disruptions.

1.1. Proxy Robustness

A very popular approach for developing robust planning models is to identify a particular aspect of the airline planning process that is a proxy for robustness. We call this class of robust planning models *proxy robust*. The performance of these approaches depends on efficacy of the aspect capturing the desired robustness goals and can vary across data sets. Furthermore, additions to the model (e.g., enlarging the set of possible decisions, or introducing additional constraints) may render a particular proxy robust approach less effective. There is, by definition of this class, no feedback between the planning stage and the operations stage that could improve the robust solution potentially leading to an overly conservative planned solution.

The general approach to airline planning involves defining a series of stages to be solved in a sequential order. These stages typically consist of, but are not limited to, the schedule design, fleet assignment, aircraft routing, crew planning, and passenger routing. To improve upon the solution quality and robustness of this sequential process, there has been increasing interest in finding an integrated solution for two or more stages. Cordeau et al. (2001) integrate the crew scheduling and aircraft routing problems through the use of short connections and solve this problem by

Benders' decomposition. Ensuring that the short connections are used by the same aircraft and crew improves the robustness by reducing the possibility of delays spreading throughout the network, commonly called delay propagation. The Benders' decomposition approach to integrated planning is extended by Mercier, Cordeau, and Soumis (2005), who introduce the concept of restricted connections, which are penalised if the two flights are serviced by the same crew and not the same aircraft. Further integration using Benders' decomposition is carried out by Papadakos (2009), with the integration of fleet assignment, maintenance routing, and crew pairing.

To integrate the aircraft routing and crew scheduling problems, an iterative approach is proposed by Weide, Ryan, and Ehrgott (2010) and Dunbar, Froyland, and Wu (2012). This approach is applied to handle the combined delay interactions from the routing and crew connection networks with a focus on propagated delay. Weide, Ryan, and Ehrgott (2010) attempt to increase the amount of slack, excess connection time, between the restricted connections to mitigate delay propagation. Dunbar, Froyland, and Wu (2012) explicitly measure the delay that would be propagated along assigned flight routes, or strings, in the combined routing and crewing network, and minimise this. Both Weide, Ryan, and Ehrgott (2010) and Dunbar, Froyland, and Wu (2012) determine that the length of the connections is a contributing factor to the performance of an airline planning solution.

Integration of two or more airline planning processes has been shown to improve robustness, however there have been many alternate methods proposed for individual planning stages. In the case of the aircraft routing problem, Lan, Clarke, and Barnhart (2006) propose the use of flight retiming to reduce delay propagation and minimise the effect of disruptions on passengers. In practice, a general method of introducing robustness into the aircraft routing problem is via *key performance indicators*; examples of such methods are explained in Wu (2010). Borndörfer et al. (2010) present an alternative to a traditional *key performance indicator* method for the aircraft routing problem, more specifically the tail assignment problem, which aims to find the aircraft routing with the lowest potential propagated delay. In Borndörfer et al. (2010) the probability of the length of delay is explicitly modelled, and the expected delay is included in the objective function.

Within the airline planning problems there are a number of different characteristics identified that can be exploited to improve operational performance. In the development of a robust airline schedule, Ageeva (2000) presents the idea of incorporating aircraft swaps to avoid the propagation of delay. Swapping occurs at points when two aircraft are planned to be on the ground at the same airport at the same time, with the swap allowing the higher valued route to continue on time. Eggenberg (2009) presents both robust and recoverable aircraft routing problems by optimising favourable characteristics of the model, such as aircraft swaps (Ageeva 2000) and increased idle and passenger connection times, to improve recoverability with the use of uncertainty feature optimisation. Kang (2004) proposes a unique method for robust planning involving decomposing the airline schedule into a number of different layers, partitioning the flights into layers by their expected yield. A simple recovery algorithm is devised from this approach, where the highest layer is rerouted first and the lowest layer has a higher delay and cancellation priority. These approaches are attempting to provide options for the airline in the event of disruption, reducing the potential recovery costs.

Some U.S. airline networks are designed with a hub and spoke structure, where the majority of the activity occurs at the hubs. Rosenberger, Johnson, and Nemhauser (2004) exploit this particular network structure by introducing the concept of hub isolation and short cycles. Limiting the number of aircraft that service each hub in the network introduces possible recovery actions to isolate a disruption to a particular hub, protecting all other flights. The ideas of Rosenberger, Johnson, and Nemhauser (2004) are extended by Smith and Johnson (2006) to include the concept of "station purity." This concept involves limiting the number of fleets that can service each station, attempting to provide more recovery options, such as aircraft swapping opportunities. The integration of this work with crew planning is demonstrated in Gao, Johnson, and Smith (2009), providing station purity restrictions on each crew base to help ensure that each crew can return to base in a disruption.

1.2. Feedback Robustness

Conversely, other approaches incorporate this feedback via second-stage (recovery) decisions, and we call this class *feedback robust*. Feedback robust approaches are a potentially superior method for reducing the weighted recovery costs because it is possible to improve the planned solution with outcomes from simulated recovery scenarios. Yen and Birge (2006) present an example of such an approach solving a robust crew scheduling problem using stochastic programming. In this model each subproblem describes a disruption scenario, which evaluates the effect of the disruption on the propagated delay caused by crew pairings.

Outside the airline literature, Liebchen et al. (2009) have developed a concept called *recoverable robustness* with an application to railway transportation. This technique focuses on finding a planning solution

that is recoverable in a limited number of steps, or with *limited effort*. Liebchen et al. (2009) present the recoverable robust timetabling problem as a demonstration of this technique. The concept of limited effort is related to the selection of a simple recovery algorithm and attempting to minimise the number of changes made to the planned timetable in the recovery stages. In Liebchen et al. (2009), the authors contrast recoverable robustness with robust planning, indicating that strict robustness can often be overly conservative, requiring a planned solution to perform under *all* disruptions. A key feature of recoverable robustness is recognising that in a disruption, recovery is required and the planned solution will need to be changed in operations.

In this paper we present a model for the tail assignment problem in a recoverable robust framework, attempting to develop a planning solution that is recoverable with limited effort. For the recoverable robust tail assignment problem (RRTAP) we define limited effort as the *lowest recovery cost* with the least number of required changes. Because this is a feedback robust approach, the quality of the planned solution is dependent on the feedback provided from the recovery scenarios. In this respect, we extend upon the recovery policies considered by Liebchen et al. (2009) to include a full set of airline recovery options, such as flight cancellations and delays and allowing for aircraft rerouting. The inclusion of these recovery policies provides an accurate simulation of an airline operations control centre and produces quality feedback for the planning problem. In our novel approach for the RRTAP we formulate a full set of recovery options while combining the planning and recovery problems in a difficult mixed-integer program.

1.3. Integrating Planning and Recovery

The RRTAP attempts to improve the planned solution based on information from recovery scenarios as a form of feedback robustness. In contrast, the proxy robust models incorporate robustness by focusing on perceived beneficial characteristics to the planning problem. There is no guarantee that the selected characteristic will improve robustness, thus can only be evaluated through experiments. By explicitly solving the recovery problem, the RRTAP is not limited to specific actions to find the best recovery solution to a given planned tail assignment. As a result, the RRTAP is a more sophisticated attempt to improve recoverability by allowing feedback between the planning stage and an accurately simulated recovery subproblem. The feedback provided by the evaluation of the recovery problem minimises the expected recovery costs over a large range of scenarios. This approach is similar to the stochastic programming approach applied to the crew pairing problem in Yen and Birge (2006), however only a limited set of recovery options are employed in the second stage. One of our contributions is the implementation of a more complex recovery subproblem in the RRTAP that uses a *full* set of recovery options, aircraft rerouting, and delaying or cancelling flights. To the best of the authors' knowledge this two-stage approach—incorporating a full set of allowable recovery decisions—has not been applied to airline optimisation. The resulting planned tail assignment solution will assign flights to aircraft that will reduce the expected cost of flight cancellations and delays and aircraft rerouting.

There are a number of different techniques that are available to the airline operations control centre to mitigate the effects of disruptions. These are generally in the form of flight delays, cancellations, and aircraft swapping (Kohl et al. 2007). Each of these recovery actions contribute a high cost, constructed with either real or artificial costs, to the objective function of the minimisation problem. In addition to minimising the use of these recovery techniques there is a focus on returning operations back to plan either within a specified time period or as quickly as possible. A very good, recent review of the recovery literature can be found in Clausen et al. (2010).

In this paper, §2 describes the concept of recoverable robustness and how we apply it to the tail assignment problem. This section includes a brief review of recovery literature and a description of the policies that have been applied in our model. We also present the mathematical model for the recoverable robust tail assignment problem in §2. A description of the techniques used to solve the RRTAP is provided in §3. In §4 we present our numerical results based on airline data and in §5 we report our conclusions.

2. The Recoverable Robust Tail Assignment Problem (RRTAP)

We define the tail assignment problem as the task of assigning routes to individual aircraft, ensuring that all flights are serviced in a network while maintaining operational constraints. The RRTAP solves the planning and recovery tail assignment problems simultaneously in a stochastic programming framework to improve the recoverability of the planned solution. To provide a detailed representation of the recovery problem in the RRTAP, a full set of recovery options and estimations of actual costs will be employed. Also, the planning tail assignment problem is generally modelled as a feasibility problem, and results in a large number of feasible solutions. Therefore, the use of the RRTAP attempts to improve the expected recovery costs of the planned solution without any additional planning costs.

Table 1 Notation Used in the Model

S	is the set of all scenarios s
R	is the set of all aircraft r
P^r	is the set of all strings p for aircraft r, the planning variables
P^{sr}	is the set of all strings p for aircraft r in scenario s, the recovery variables
N	is the set of all flights j
$N^{s\text{-pre}}$	is the set of all flights j that depart before the first disrupted flight in scenario s
$N^{s\text{-post}}$	is the set of all flights j that depart after and including the first disrupted flight in scenario s
B	is the set of all airports b an aircraft flight route can begin and terminate
C	is the set of all feasible connections in the network, $C = \{(i, j) \mid i, j \in N \cup B\}$
M_b	is the minimum number of aircraft required to start the following days flights from base b
y_p^r	$= 1$ if aircraft r uses string p, 0 otherwise
c_p^r	= the cost of flying aircraft r on string p
a_{jp}	$= 1$ if flight j is contained in string p, 0 otherwise
o_{bp}	$= 1$ if string p terminates at airport b, 0 otherwise
y_p^{sr}	$= 1$ if aircraft r uses string p in scenario s, 0 otherwise
c_p^{sr}	= the cost of flying aircraft r on string p in scenario s, this includes the cost of any delayed flight on that string
a_{jp}^s	$= 1$ if flight j is contained in string p in scenario s, 0 otherwise
o_{bp}^s	$= 1$ if string p terminates at airport b in scenario s, 0 otherwise
d_j	= the cost of cancelling flight j
z_j^s	$= 1$ if the flight j is cancelled in scenario s, 0 otherwise
$\epsilon_{jr}^{s+}, \epsilon_{jr}^{s-}$	$\epsilon_{jr}^{s+} = 1, \epsilon_{jr}^{s-} = 0$ if flight j is assigned to aircraft r for planning but not for recovery in scenario s $\epsilon_{jr}^{s+} = 0, \epsilon_{jr}^{s-} = 1$ if flight j is assigned to aircraft r for recovery in scenario s but not for planning $\epsilon_{jr}^{s+} = 0, \epsilon_{jr}^{s-} = 0$ if flight j is not assigned to aircraft r for both planning and recovery in scenario s
g^s	weight applied to ϵ_{jr}^{s-} in the objective function, the swap cost
w^s	weight for each scenario s in the objective function

The tail assignment model for the RRTAP has been developed using a flight string formulation introduced by Barnhart et al. (1998). The notation presented in Table 1 is used to describe the planning and recovery stages in the RRTAP. A superscript s denotes the components of the model that relate to the recovery stages and the disruption scenario s that they belong to, where $s \in S$.

The flight schedule is described as a set of flights j contained in the set N, with each flight representing a node in a connection network. Two flights that can be performed by the one aircraft are defined as connected flights. A pair of connected flights are identified by (i) the destination of the incoming flight being the same as the origin of the outgoing flight; and (ii) a minimum connection time, called the minimum turn time, between the arrival and departure of the connected flights. The set of all feasible connections, which represent the arcs of the connection network, is described by $C = \{(i, j) \mid i, j \in N\}$. A flight string, or flight route, p is defined as a sequence of connected flights to be operated by one aircraft r. The decision variables y_p^r and y_p^{sr} equal 1 when flight string p is operated by aircraft r for the planning and recovery scenarios, respectively. The cost of using flight route p for aircraft r is given by c_p^r and c_p^{sr} for the planning and recovery scenarios. The cost of a flight route for the planning variables is dependent on the length of the connections contained in the flight route. In the recovery scenarios, the cost of a flight route is defined by the delay on flights contained in the string that is incurred during the recovery process. In the model constraints, the parameters a_{jp} and a_{jp}^s are the coefficients of the decision variables y_p^r and y_p^{sr}, respectively, that capture whether flight j is included in string p. In addition to describing a set of connected flights, the flight string also indicates end-of-day locations. All end-of-day locations b, described as aircraft bases or overnight airports, used in the model are contained in the set B. The parameters o_{bp} and o_{bp}^s equal 1 if flight string p terminates at base b for the planning and recovery scenarios, respectively. The RRTAP is a single-day problem, so to maintain feasibility for the following days' schedule we enforce a minimum number of aircraft to terminate at each end-of-day location b through the parameter M_b. The sequence of flights and the end-of-day location described in the flight string represents a column in the constraint matrix.

The tail assignment problem is the task of assigning flight routes or strings to individual aircraft. Treating each aircraft individually in this problem requires an explicit definition of all aircraft r contained in the set R. The set R contains all aircraft used in the model, and this is the same set used for the planning and all recovery stages. We generate individual strings for each aircraft r, and additionally we generate strings for the planning and each recovery stage contained in the sets P^r and P^{sr}, respectively. As a result we treat the sets P^r and P^{sr} as disjoint.

To solve the recovery tail assignment problem we implement the recovery techniques of flight delays and cancellations but also allowing aircraft rerouting. Flight delays are represented by flight copies, with each copy representing a different departure time of the same flight. All departure times for the flight copies are later than the planned departure. Flight cancellations are included in the model through the additional variables z_j^s, that equal 1 if flight j in scenario s is cancelled, contributing a cost of d_j to the objective. The addition of the cancellation variables allows the decision of an aircraft operating or cancelling a flight in the recovery problem.

There are two main objectives for this problem: minimising the cost of recovery and minimising the deviation from the planned solution. Because we are attempting to simulate the recovery process while finding the planned solution, it is important to enforce non-anticipativity. All of the flights j in this model are contained in the set N, and to model non-anticipativity we define two partitions of this set, $N^{s\text{-pre}}$ and $N^{s\text{-post}}$, $N = N^{s\text{-pre}} \cup N^{s\text{-post}}$. The sets $N^{s\text{-pre}}$ and $N^{s\text{-post}}$ include all of the flights that depart before and after the first disrupted flight in scenario s, respectively. To reduce the number of deviations in the recovery solution we penalise any difference in the flight routes assigned to an individual aircraft through the use of the variables ϵ_{jr}^{s+} and ϵ_{jr}^{s-}. In the objective function the penalty g^s is applied for every flight j that is added to the planned route for aircraft r in the recovered solution for scenario s, indicated by $\epsilon_{jr}^{s+} = 0$ and $\epsilon_{jr}^{s-} = 1$. It is possible to add and remove flights from an aircraft's planned route, however removing a flight is penalised through either cancellation or adding it to another route. The objective of minimal deviation is similar to that presented by Thengvall, Bard, and Yu (2000). In the RRTAP the recovery model includes constraints in the column generation master problem to track the amount of deviation from the planned solution. This differs from the model presented in Thengvall, Bard, and Yu (2000), where the deviation from the planned solution is restricted through arc constraints in the subproblem. The objective of Thengvall, Bard, and Yu (2000) is to maintain the use of the planned connections in recovery, whereas our model attempts to maintain the same flights for each aircraft. The second objective of the RRTAP attempts to find a set of recovered solutions with *limited effort*; in our case this is defined as the lowest cost recovery solution. As mentioned in §2, we will be using actual costs for the delay and cancellation of flights in the recovery scenarios.

The recoverable robust tail assignment problem is formed by simultaneously solving the planning and recovery problems within the one model. We describe the RRTAP as follows:

(RRTAP)

$$\min \left\{ \sum_{r\in R}\sum_{p\in P^r} c_p^r y_p^r + \sum_{s\in S} w^s \left\{ \sum_{r\in R}\sum_{p\in P^{sr}} c_p^{sr} y_p^{sr} + \sum_{j\in N} d_j z_j^s + \sum_{r\in R}\sum_{j\in N} g^s \epsilon_{jr}^{s-} \right\} \right\}, \tag{1}$$

$$\text{s.t.} \sum_{r\in R}\sum_{p\in P^r} a_{jp} y_p^r = 1 \quad \forall j \in N, \tag{2}$$

$$\sum_{p\in P^r} y_p^r \le 1 \quad \forall r \in R, \tag{3}$$

$$\sum_{r\in R}\sum_{p\in P^r} o_{bp} y_p^r \ge M_b \quad \forall b \in B, \tag{4}$$

$$\sum_{r\in R}\sum_{p\in P^{sr}} a_{jp}^s y_p^{sr} + z_j^s = 1 \quad \forall s \in S, \forall j \in N, \tag{5}$$

$$\sum_{p\in P^{sr}} y_p^{sr} \le 1 \quad \forall s \in S, \forall r \in R, \tag{6}$$

$$\sum_{r\in R}\sum_{p\in P^{sr}} o_{bp}^s y_p^{sr} \ge M_b \quad \forall s \in S, \forall b \in B, \tag{7}$$

$$\sum_{p\in P^r} a_{jp} y_p^r - \sum_{p\in P^{sr}} a_{jp}^s y_p^{sr} = 0 \quad \forall s \in S, \forall r \in R, \forall j \in N^{s\text{-pre}}, \tag{8}$$

$$\sum_{p\in P^r} a_{jp} y_p^r - \sum_{p\in P^{sr}} a_{jp}^s y_p^{sr} = \epsilon_{jr}^{s+} - \epsilon_{jr}^{s-} \quad \forall s \in S, \forall r \in R, \forall j \in N^{s\text{-post}}, \tag{9}$$

$$y_p^r \in \{0,1\} \quad \forall r \in R, \forall p \in P^r, \tag{10}$$

$$y_p^{sr} \in \{0,1\} \quad \forall s \in S, \forall r \in R, \forall p \in P^{sr}, \tag{11}$$

$$z_j^s \in \{0,1\} \quad \forall s \in S, \forall j \in N, \tag{12}$$

$$\epsilon_{jr}^{s+} \ge 0, \quad \epsilon_{jr}^{s-} \ge 0 \quad \forall s \in S, \forall r \in R, \forall j \in N. \tag{13}$$

The planning tail assignment problem is described by constraints (2)–(4) and (10). Similarly the recovery tail assignment problem is described by constraints (5)–(7) and (11)–(12) for each scenario s. The constraints (8)–(9) are used to track the deviation between the planning and recovery tail assignment problem variables.

The objective function (1) minimises the cost of the planning tail assignment and the expected cost of the recovery solutions from all scenarios, weighted by w^s, with a penalty for each flight change g^s. Flight coverage in the planning model is enforced through constraints (2) and in the recovery model through constraints (5) with an additional variable z_j^s to allow for flight cancellations. The restriction on the number of available aircraft is described in constraints (3) and (6) for the planning and recovery scenarios. Also, constraints (4) and (7) ensure the required number of

aircraft are positioned at each airport at the end of the day to begin the next day's flying in the planing and recovery scenarios. The set of constraints (8) are the non-anticipativity constraints ensuring that each aircraft r is assigned to the same flights in both the planning and recovery stages up to the first disrupted flight for each scenario s. Because all scenarios are known ahead of time we require these constraints to reflect decisions that would be made by the airline operations control centre in the event of a disruption. After, and including, the first disrupted flight, the constraints (9) are used to count any deviation in the flight strings assigned to each aircraft for planning and recovery variables in absolute terms. In the objective function we have included only the variable ϵ^{s-}_{jr}, which represents whether flight j is added to the recovered flight route of aircraft r. Because the RRTAP is a minimisation problem, the optimal solution requires ϵ^{s+}_{jr} and ϵ^{s-}_{jr} to be tightly constrained at the lower bound, defined by constraints (9) or (13). Now the lower bound of (13) is dominated by (9) when the left-hand side is greater than zero, which is at most one. Therefore, the values of ϵ^{s+}_{jr} and ϵ^{s-}_{jr} will be at most one in the optimal solution of the RRTAP.

3. Solution Methodology

The RRTAP is a large-scale optimisation problem that solves the planning and recovery tail assignment problems simultaneously. The resulting naïve formulation of the RRTAP is a very large and intractable problem, requiring decomposition and enhancement techniques to improve the solution runtime. A key feature of our solution methodology is to integrate the techniques of Benders' decomposition and column generation, shown to be very effective in solving integrated airline planning problems (Cordeau et al. 2001; Mercier, Cordeau, and Soumis 2005; Papadakos 2009).

Because the RRTAP is similar in structure to a stochastic program, when decomposed by Benders' decomposition, each of the recovery scenarios form scenario subproblems. This technique moves the difficult constraints, Equations (8)–(9), to the subproblem, and by fixing the planning variables from the master problem, the individual recovery problems are solved more efficiently.

The planning master problem can take any form because the only connection between it and the subproblems is the assignment of flights to aircraft, which is the main objective of the tail assignment problem. This is also true for the recovery subproblems, where a variety of simple or complicated recovery policies can be used. By having the recovery problem separated out into the Benders' subproblems it is even possible to use heuristic recovery techniques in an attempt to improve the runtime of the RRTAP.

In the implementation of Benders' decomposition, further techniques can be used to accelerate the convergence of this method. Such techniques include the Magnanti–Wong method (Magnanti and Wong 1981), the independent Magnanti–Wong method (Papadakos 2008), and local branching (Rei et al. 2009); here we use the Magnanti–Wong method (Magnanti and Wong 1981). To attain an integral solution we require the use of branch and price, and in our implementation we have contributed new, and modified existing, branching rules for use with the tail assignment problem. By identifying specific structures in our problem, we are able to exploit this through branching rules to enhance the branch-and-price process.

In the next sections we will describe how we applied the techniques of Benders' decomposition and column generation to the RRTAP and the enhancement techniques used.

3.1. Benders' Decomposition

The decomposition for this problem is clear given the distinct separation of variables, y^{sr}_p, z^s_j, ϵ^{s+}_{jr}, and ϵ^{s-}_{jr}, between each of the recovery scenarios $s \in S$. The primal Benders' subproblems (PBSP-s) created for each scenario s include these variables and the constraints (5)–(9). Only the planning variables, y^r_p, $\forall r \in R$, $\forall p \in P^r$, and the constraints (2)–(4) are included in the Benders' decomposition master problem (BMP). At each iteration of the Benders' decomposition algorithm a Benders' cut will be found from the optimal dual solutions for every scenario s and introduced to the BMP. The feasibility Benders' cuts ensure that PBSP-s is feasible for the solution of the optimal planning variables, and the optimality cuts provide a lower bound for the objective function of PBSP-s in the BMP. Each subproblem PBSP-s finds an optimal recovery strategy for a given set of optimal planning variables $\bar{\mathbf{y}}$ and a particular disruption scenario s. The primal Benders' decomposition subproblem for scenario s is described by

(PBSP-s)

$$\mu^s(\bar{\mathbf{y}}) = \min \left\{ \sum_{r\in R}\sum_{p\in P^{sr}} c^{sr}_p y^{sr}_p + \sum_{j\in N} d_j z^s_j + \sum_{r\in R}\sum_{j\in N} g^s \epsilon^{s-}_{jr} \right\}, \tag{14}$$

$$\text{s.t.} \sum_{r\in R}\sum_{p\in P^{sr}} a^s_{jp} y^{sr}_p + z^s_j = 1 \quad \forall j \in N, \tag{15}$$

$$\sum_{p\in P^{sr}} y^{sr}_p \le 1 \quad \forall r \in R, \tag{16}$$

$$\sum_{r\in R}\sum_{p\in P^{sr}} o^s_{bp} y^{sr}_p \ge M_b \quad \forall b \in B, \tag{17}$$

$$\sum_{p\in P^{sr}} a^s_{jp} y^{sr}_p = \sum_{p\in P^r} a_{jp}\bar{y}^r_p \quad \forall r\in R,\ \forall j\in N^{s\text{-pre}}, \tag{18}$$

$$\sum_{p\in P^{sr}} a^s_{jp} y^{sr}_p + \epsilon^{s+}_{jr} - \epsilon^{s-}_{jr} = \sum_{p\in P^r} a_{jp}\bar{y}^r_p \quad \forall r\in R,\ \forall j\in N^{s\text{-post}}, \tag{19}$$

$$y^{sr}_p \geq 0 \quad \forall r\in R,\ \forall p\in P^{sr}, \tag{20}$$

$$z^s_j \geq 0 \quad \forall j\in N, \tag{21}$$

$$\epsilon^{s+}_{jr} \geq 0, \quad \epsilon^{s-}_{jr} \geq 0 \quad \forall r\in R,\ \forall j\in N. \tag{22}$$

The optimal solution for BMP is given by the set of variables $\bar{\mathbf{y}}$, which are fixed and passed to PBSP-s for each scenario s. To ensure that PBSP-s is always feasible, an initial set of strings $p'\in P^{sr}$ are generated by replicating the routes from the optimal master problem variables $\bar{\mathbf{y}}$ for all flights j contained in $N^{s\text{-pre}}$, $a_{jp'}\bar{y}^r_{p'} = a^s_{jp'} y^{sr}_{p'}$, $\forall j\in N^{s\text{-pre}}$, $\forall r\in R$, $\forall p'\in P^r$. This satisfies the cover constraints (15) because we are able to set $z^s_j = 1$, $\forall j\in N^{s\text{-post}}$ and the non-anticipativity constraints (18). We define the dual variables as $\boldsymbol{\beta}^s = \{\beta^s_j \mid \forall j\in N\}$, $\boldsymbol{\gamma}^s = \{\gamma^{sr} \mid \forall r\in R\}$, $\boldsymbol{\lambda}^s = \{\lambda^s_b \mid \forall b\in B\}$, and $\boldsymbol{\delta}^s = \{\delta^{sr}_j \mid \forall r\in R,\ \forall j\in N\}$ for the constraints (15)–(17) and (18)–(19), respectively. For each scenario s, after solving PBSP-s the Benders' cuts are generated from the dual solutions of (15)–(19). The resulting Benders' optimality cut generated from a single iteration of the Benders' decomposition algorithm is defined as

$$\varphi^s \geq \sum_{j\in N}\beta^s_j + \sum_{r\in R}\gamma^{sr} + \sum_{b\in B}\lambda^s_b M_b + \sum_{r\in R}\sum_{j\in N}\sum_{p\in P^r}\delta^{sr}_j a_{jp} y^r_p. \tag{23}$$

For a fixed $\bar{\mathbf{y}}$, the right-hand side of the Benders' optimality cut (23) is the objective function value for the dual of PBSP-s. The dual solutions of (15)–(19) express an extreme point of the dual problem of PBSP-s. The initial columns generated for each subproblem ensure that PBSP-s is always feasible and hence only optimality cuts, of the form given by Equation (23), are required to be added to the BMP.

To apply the Benders' cuts from the PBSP-s to the BMP an additional decision variable φ^s must be added to the master problem objective function. The value of φ^s in the solution of the BMP provides the current lower bound of the objective function for the PBSP-s in the master problem, constrained through the added cuts. In the solution process of the Benders' decomposition scheme we introduce a new set Ω^s, which is the set of all Benders' cuts ω for an individual scenario s. Each Benders' cut ω is defined by the dual variables $\boldsymbol{\beta}^{\omega s}$, $\boldsymbol{\gamma}^{\omega s}$, $\boldsymbol{\lambda}^{\omega s}$, and $\boldsymbol{\delta}^{\omega s}$ from the PBSP-s for disruption scenario s. The Benders' decomposition master problem (BMP) is described as follows:

(BMP)

$$\Phi = \min \left\{ \sum_{r\in R}\sum_{p\in P^r} c^r_p y^r_p + \sum_{s\in S} w^s \varphi^s \right\}, \tag{24}$$

$$\text{s.t.} \sum_{r\in R}\sum_{p\in P^r} a_{jp} y^r_p = 1 \quad \forall j\in N, \tag{25}$$

$$\sum_{p\in P^r} y^r_p \leq 1 \quad \forall r\in R, \tag{26}$$

$$\sum_{r\in R}\sum_{p\in P^r} o_{bp} y^r_p \geq M_b \quad \forall b\in B, \tag{27}$$

$$\varphi^s - \sum_{r\in R}\sum_{j\in N}\sum_{p\in P^r}\delta^{\omega sr}_j a_{jp} y^r_p \geq \sum_{j\in N}\beta^{\omega s}_j + \sum_{r\in R}\gamma^{\omega sr} + \sum_{b\in B}\lambda^{\omega s}_b M_b \quad \forall s\in S,\ \forall \omega\in\Omega^s, \tag{28}$$

$$y^r_p \in \mathbb{Z}^+ \quad \forall r\in R,\ \forall p\in P^r, \tag{29}$$

$$\varphi^s \geq 0 \quad \forall s\in S. \tag{30}$$

The Benders' decomposition solution process is performed by solving BMP and then with the optimal planning solution $\bar{\mathbf{y}}$ checking PBSP-s for each scenario s for any improvement cuts. There is a trade-off in the computational performance and the optimality gap of the solution. The lower bound of PBSP-s for each scenario s is provided by the value of φ^s in the solution to BMP and the upper bound is the objective function value of PBSP-s for a given $\bar{\mathbf{y}}$, $\mu^s(\bar{\mathbf{y}})$, at each iteration. Our chosen criteria for adding additional cuts is the gap between the upper and lower bounds, $\mu^s(\bar{\mathbf{y}})$ and φ^s, respectively, of the Benders' decomposition problem relative to the Benders' master problem objective value Φ. This optimal subproblem criteria is similar to the condition proposed in Papadakos (2009) as the stopping criteria. For each iteration of the Benders' decomposition, a cut will be added if the following condition is violated:

$$\frac{\mu^s(\bar{\mathbf{y}}) - \varphi^s}{\Phi} < \varepsilon \quad \forall s\in S, \tag{31}$$

where ε is the tolerance used in our model, set at $\varepsilon = 10^{-4}$. The Benders' master problem is solved when no improvement can be made with further cuts from the subproblems; this is equivalent to condition (31) being satisfied for all $s\in S$.

3.1.1. The Magnanti–Wong Method. At each iteration in the solution process the generated cuts provide an incremental improvement to the master problem. The efficiency of the solution process is dependent on the quality of the cuts that are added from the subproblems. In the PBSP-s it is common for a degenerate primal solution to be found that indicates that multiple optimal dual solutions exist. In this situation, the cut that will produce the best improvement in the BMP can be found using the Magnanti–Wong method (Magnanti and Wong 1981). The objective of this method is to find a cut that *dominates* all other possible cuts in the current iteration

of the subproblem; we call such a cut Pareto optimal. Given two optimal dual solutions $(\boldsymbol{\beta}_1^s, \boldsymbol{\gamma}_1^s, \boldsymbol{\lambda}_1^s, \boldsymbol{\delta}_1^s) \neq (\boldsymbol{\beta}_2^s, \boldsymbol{\gamma}_2^s, \boldsymbol{\lambda}_2^s, \boldsymbol{\delta}_2^s)$, the cut generated from solution 1 dominates solution 2 if and only if

$$\sum_{j\in N}\beta_{1j}^s+\sum_{r\in R}\gamma_1^{sr}+\sum_{b\in B}\lambda_{1b}^s M_b+\sum_{r\in R}\sum_{j\in N}\sum_{p\in P^r}\delta_{1j}^{sr}a_{jp}y_p^r \geq \sum_{j\in N}\beta_{2j}^s+\sum_{r\in R}\gamma_2^{sr}+\sum_{b\in B}\lambda_{2b}^s M_b+\sum_{r\in R}\sum_{j\in N}\sum_{p\in P^r}\delta_{2j}^{sr}a_{jp}y_p^r \quad (32)$$

for all $y=\{y_p^r, r\in R, p\in P^r\}$ with a strict inequality for at least one point. To find the Pareto optimal cut, the Magnanti–Wong method introduces an auxiliary optimisation problem to find the cut that is closest to a chosen core point $\mathbf{y}^0$. The core point is a point that is chosen to be within the relative interior of the LP relaxation of (25)–(30), $\mathbf{y}^0 \in \text{ri}(\mathbf{y}^{LP})$; the method by which this point is obtained is explained later in this section. Because the core point is selected to be within $\text{ri}(\mathbf{y}^{LP})$, satisfying condition (32) for $y=\mathbf{y}^0$ ensures that the condition is satisfied for all y. We define the dual Magnanti–Wong auxiliary problem (DMWAP-s) for scenario s as

(DMWAP-s)

$$\max \left\{\sum_{j\in N}\beta_j^s+\sum_{r\in R}\gamma^{sr}+\sum_{b\in B}\lambda_b^s M_b+\sum_{r\in R}\sum_{j\in N}\sum_{p\in P^r}\delta_j^{sr}a_{jp}y_p^{0r}\right\}, \quad (33)$$

$$\text{s.t.} \sum_{j\in N}\beta_j^s+\sum_{r\in R}\gamma^{sr}+\sum_{b\in B}\lambda_b^s M_b+\sum_{r\in R}\sum_{j\in N}\sum_{p\in P^r}\delta_j^{sr}a_{jp}\bar{y}_p^r=\mu^s(\bar{\mathbf{y}}), \quad (34)$$

$$(\boldsymbol{\beta}^s, \boldsymbol{\gamma}^s, \boldsymbol{\lambda}^s, \boldsymbol{\delta}^s)\in\Delta^s, \quad (35)$$

where Δ^s is the dual solution feasibility space from the PBSP-s and its objective function value is given by $\mu^s(\bar{\mathbf{y}})$.

The DMWAP-s is identical to the dual problem of PBSP-s with the addition of constraint (34) and a change in the objective function. The primal form of DMWAP-s, which we will refer to as PMWAP-s, can be derived from the PBSP-s by (i) including a primal variable corresponding to the additional dual constraint (34); and (ii) setting the right-hand side of the comparison constraints (18)–(19) to the value of the core point $\mathbf{y}^0$. As such, the implementation of the PMWAP-s does not require much more additional development and the computational time is comparable to the PBSP-s. In our Benders' decomposition algorithm we solve the PMWAP-s to find the Pareto optimal cuts.

To implement the Magnanti–Wong method it is a requirement to find a representative core point within the relative interior of the LP relaxation of the BMP. In the case of a degenerate PBSP-s, the DMWAP-s, or the primal form (PMWAP-s), is guaranteed to find the dual solution that is Pareto optimal for the chosen core point as defined by the dominance condition (32). Given that the Benders' master problem is solved using column generation, the set of all variables has not been completely enumerated, hence the LP relative interior is not fully known. This makes the task of finding a core point within the relative interior difficult, and consequently the core point selection can only be made as an approximation without the guarantee that $\mathbf{y}^0 \in \text{ri}(\mathbf{y}^{LP})$. Mercier, Cordeau, and Soumis (2005) state that using a core point $\mathbf{y}^0 \notin \text{ri}(\mathbf{y}^{LP})$ does not preclude PMWAP-s from finding a valid Benders' cut. However, the further $\mathbf{y}^0$ is from $\text{ri}(\mathbf{y}^{LP})$, the weaker the Benders' cuts that are generated by this method. An important consideration for the Magnanti–Wong method is whether the chosen core point $\mathbf{y}^0$ is closer to the relative interior $\text{ri}(\mathbf{y}^{LP})$ than the solution to BMP, $\bar{\mathbf{y}}$. The solution to PMWAP-s will always generate a Benders' cut closest to the chosen core point $\mathbf{y}^0$, which will satisfy the dominance condition (32). If the core point is further from the relative interior than the current solution to BMP, then the Benders' cut generated by the solution to PMWAP-s can be further from $\text{ri}(\mathbf{y}^{LP})$ than the cut generated from PBSP-s. This demonstrates the importance of finding a good representative core point to ensure that the Magnanti–Wong method finds a Benders' cut to improve the BMP solution.

Papadakos (2008) has developed enhancements to the Magnanti–Wong method, which we will discuss in §3.3, along with different schemes used to find an appropriate core point. One such scheme for a binary problem is to set the core point to $\mathbf{y}^0=\mathbf{1}$ or $\mathbf{y}^0=\mathbf{0}$, which is employed by Mercier, Cordeau, and Soumis (2005). However, this particular scheme is not useful for our problem given the complexity of the comparison constraints (18)–(19) and generally causes the PMWAP-s to be infeasible. Another scheme, presented by Papadakos (2008), is to set the core point to the initial solution of the BMP, $\mathbf{y}^0 \to \bar{\mathbf{y}}_0$, then after each iteration i of the master problem update the core point by $\mathbf{y}^0 \to \frac{1}{2}\mathbf{y}^0+\frac{1}{2}\bar{\mathbf{y}}_i$. The benefit of this particular scheme is that at each iteration i of the Benders' decomposition algorithm the core point is moving closer toward the $\text{ri}(\mathbf{y}^{LP})$. Even if the initial core point is not within the relative interior, $\mathbf{y}^0 \notin \text{ri}(\mathbf{y}^{LP})$, it is possible to incrementally improve the potential strength of the Benders' cuts with more iterations of the Benders' decomposition algorithm. Through experimental experience this latter scheme from Papadakos (2008) has been shown to be useful

in our problem to find a representative core point. We demonstrate in §4.3 that the Magnanti–Wong method greatly improves the efficiency of the Benders' decomposition solution process.

3.2. Column Generation

Given the exponentially large number of variables in the Benders' master (BMP) and subproblems (PBSP-s), both are solved using column generation. Each of these problems share a similar structure and the column generation subproblems are solved using the same algorithm, so for conciseness we will only describe in detail the implementation of column generation for the PBSP-s.

The PBSP-s is formulated as a LP and can be efficiently solved using column generation. Each iteration of the column generation method improves the master problem by introducing negative reduced cost columns. These columns are generated from the subproblem using the current LP dual solutions of the PBSP-s. In §3.1 we defined the dual variables for each scenario subproblem s as $\boldsymbol{\beta}^s = \{\beta_j^s \mid \forall\, j \in N\}$, $\boldsymbol{\gamma}^s = \{\gamma^{sr} \mid \forall\, r \in R\}$, $\boldsymbol{\lambda}^s = \{\lambda_b^s \mid \forall\, b \in B\}$, and $\boldsymbol{\delta}^s = \{\delta_j^{sr} \mid \forall\, r \in R, \forall\, j \in N\}$ for the constraints (15)–(17) and (18)–(19), respectively.

The column generation subproblem for the PBSP-s is formulated as a network flow problem for each aircraft r. The connection network used to solve the column generation subproblem for the PBSP-s is defined by a set of nodes and a set of arcs. As stated in §2, the nodes for the planning connection network are defined by the flights in the schedule. Because the technique of flight copies is used to implement delays in the recovery subproblem, each flight within the schedule must be duplicated once for each copy. The set of copies v for flight j is given by U_j. There are different sets of copies for each partition of the flight schedule, as such $\forall\, j \in N^{s\text{-pre}}$, $U_j = \{0\}$ indicating the set of copies only contains the copy representing the original schedule departure. Also, for all flights $j \in N^{s\text{-post}}$, U_j contains $v = 0$ and at least one other copy representing some delay on flight j. We define the set $N^s = \{j_v \mid j \in N, v \in U_j\}$ as all nodes that define the connection network for the PBSP-s. The rules presented in §2 that describe a feasible connection are used to define the set of connections for the PBSP-s contained in $C^s = \{(i_u, j_v) \mid i_u, j_v \in N^s\}$. We define the variables $x_{i_u j_v}^{sr}$ that equal 1 if connection (i_u, j_v) is used in a string generated for aircraft r in scenario s, 0 otherwise. The cost of using connection (i_u, j_v) for aircraft r in scenario s is defined by $c_{i_u j_v}^{sr}$. Finally, $b_r \in B$ represents the overnight base where aircraft r is located at the start of the day. The shortest path problem to generate negative reduced cost columns for the PBSP-s is defined as

$$\bar{c}_p^{sr} = \min\Bigg\{ \sum_{(i_u, j_v) \in C^s} c_{i_u j_v}^{sr} x_{i_u j_v}^{sr} - \sum_{j_v \in N^s} \sum_{\substack{i_u \in N^s: \\ (i_u, j_v) \in C^s}} (\beta_j^s + \delta_j^{sr}) x_{i_u j_v}^{sr} - \sum_{b \in B} \sum_{\substack{i_u \in N^s: \\ (i_u, b) \in C^s}} \lambda_b^s x_{i_u b}^{sr} - \gamma^{sr} \Bigg\}, \tag{36}$$

$$\text{s.t.} \quad \sum_{\substack{i_u \in N^s: \\ (i_u, j_v) \in C^s}} x_{i_u j_v}^{sr} - \sum_{\substack{k_w \in N^s: \\ (j_v, k_w) \in C^s}} x_{j_v k_w}^{sr} = 0 \quad \forall\, j_v \in N^s, \tag{37}$$

$$\sum_{v \in U_j} \sum_{\substack{i_u \in N^s: \\ (i_u, j_v) \in C^s}} x_{i_u j_v}^{sr} \le 1 \quad \forall\, j \in N, \tag{38}$$

$$\sum_{\substack{i_u \in N^s: \\ (b_r, i_u) \in C^s}} x_{b_r i_u}^{sr} = 1, \tag{39}$$

$$\sum_{b \in B} \sum_{\substack{i_u \in N^s: \\ (i_u, b) \in C^s}} x_{i_u b}^{sr} = 1, \tag{40}$$

$$x_{i_u j_v}^{sr} \in \{0, 1\}, \quad \forall\, (i_u, j_v) \in C^s. \tag{41}$$

The problem described by (36)–(41) is a network flow problem with one source and multiple sink nodes. Constraints (37) describe the flow balance and constraints (38) ensure that each node in the network is visited at most once. As a variation on the classic network flow problem, the use of multiple flight copies requires that only one copy per flight is included in the shortest path, achieved through the coverage constraints (38). Given that (36)–(41) is formulated as a network flow problem, there are a number of classical algorithms available to efficiently solve this problem.

Because the network we are using for the shortest path problem is a directed acyclic graph, it is possible to construct a topological ordering of the nodes. A topological order is defined as a list where node i is ordered before node j if $\exists\,(i, j) \in C$ (Ahuja, Magnanti, and Orlin 1993). Because the connection network forms a directed acyclic graph, a simple pulling algorithm, that "pulls" the shortest path distance from earlier processed nodes, can be used to efficiently solve the shortest path problem described by (36)–(41). An example of such a pulling algorithm, which we have implemented for this problem, is described in Ahuja, Magnanti, and Orlin (1993).

There is little difference in the column generation subproblem for the BMP and the PBSP-s, because both are formulated as network flow problems. The column generation subproblem of the BMP is formulated with the constraints (37)–(41), however a different connection network is required. Because the BMP solves the planning tail assignment problem, there is no possibility to delay flights. As such, for all flights $j \in N$, we define $U_j = \{0\}$, containing only

the copy representing the original scheduled departure time. So, using this flight copy definition, the connection network can be constructed in the same manner presented earlier for the PBSP-s. Another difference between the column generation subproblems of the BMP and the PBSP-s is the form of the objective function. For the BMP we define the optimal dual solutions as $\mathbf{u} = \{u_j \mid \forall j \in N\}$, $\mathbf{v} = \{v^r \mid \forall r \in R\}$, $\mathbf{w} = \{w_b \mid \forall b \in B\}$, and $\boldsymbol{\rho} = \{\rho^{s\omega} \mid \forall s \in S, \forall \omega \in \Omega^s\}$ for the constraints (25)–(28), respectively. As a result, the objective function for the column generation subproblem of the BMP is given by

$$\bar{c}_p^r = \sum_{(i,j)\in C} c_{ij}^r x_{ij}^r - \sum_{j\in N} \sum_{\substack{i\in N \\ |\,(i,j)\in C}} u_j x_{ij}^r - v^r - \sum_{b\in B} \sum_{\substack{i\in N \\ |\,(i,b)\in C}} w_b x_{ib}^r - \sum_{s\in S} \sum_{\omega\in\Omega^s} \left\{ \sum_{j\in N} \sum_{\substack{i\in N \\ |\,(i,j)\in C}} \delta_j^{\omega sr} x_{ij}^r \right\} \rho^{s\omega} \quad \forall r \in R. \tag{42}$$

Because of the similarities between the column generation subproblems of the BMP and PBSP-s, the same shortest path algorithm can be implemented. Because the structure of the BMP is modified after each iteration through the addition of Benders' cuts, the objective function (42) must be updated with the additional dual variables. The addition of Benders' cuts only affects connection costs while maintaining the network structure.

The efficiency of the column generation process depends on the initial solution provided to the LP relaxation of the restricted master problem. For the first iteration of the BMP a set of initial columns is manufactured by allowing the number of aircraft to equal the number of flights, with each aircraft performing only one flight to satisfy the flight cover constraints (25). For the aircraft assignment constraints (26) and the end of day constraints (27) only the first n variables are included, where $n = |R|$. To satisfy all of the constraints in the BMP we must introduce the concept of ferry flights, which is the repositioning of aircraft by flying without passengers. Constraints (27) are satisfied by using these ferry flights, and any columns that contain these flights are given an artificially high cost to ensure that they do not appear in the final solution. For each subsequent iteration of the BMP, the column generation master problem is initialised with the solution found in the previous iteration. The initialisation of the PBSP-s is simpler because the initial recovery variables can be based off the optimal planning variables from the BMP for the current iteration. The methods for generating the initial variables for the PBSP-s is explained in §3.1.

It is a well-known aspect of column generation that symmetry between the variables within the master problem affects the computational performance of the algorithm. This occurs in our model because we identify each aircraft individually, however, they are mathematically identical. To reduce the effects of this symmetry we assign one aircraft to each flight that has no preceding connecting flight; thus the only connection arc to that flight is from a source node. The number of flights with the only preceding connection from a source node for each overnight airport is provided by the constant M_b. This constant is used in constraint (27) to ensure that the required number of aircraft overnight at each end-of-day location b. Difficulties arise when the number of aircraft starting at base b is greater than M_b, allowing some symmetry to still exist. We address this problem by adding a branching rule to exclude aircraft from using particular starting flights. The specifics of this branching rule will be detailed in §3.4.

3.3. The Two-Phase Algorithm

Given the size of the Benders' master problem, it is computationally difficult to solve to integral optimality for every iteration. To overcome this complication we have implemented a two-phase algorithm that is based off the three-phase algorithm developed to solve the integrated crew scheduling and aircraft routing problem with Benders' decomposition (Cordeau et al. 2001; Mercier, Cordeau, and Soumis 2005; Papadakos 2009). The two-phase algorithm is a heuristic that initially solves the linear relaxation of the RRTAP, and reintroduces the integrality requirements to the BMP after the first phase is completed, as described in Algorithm 1. In Cordeau et al. (2001); Mercier, Cordeau, and Soumis (2005); and Papadakos (2009) the third phase is used to check the feasibility of adding integrality to the Benders' subproblem after solving the integral Benders' master problem. This is not necessary for our model because for all scenarios s, the PBSP-s is always feasible for any solution to the master problem, as explained in §3.1. Thus, it is only necessary to implement the two-phase algorithm for our model.

Algorithm 1 (The two-phase algorithm)

PHASE 1
Relax the integrality requirements for the BMP and PBSP-s $\forall s \in S$.
Set $\varphi^s \leftarrow 0$, $\forall s \in S$.
repeat
  Solve the BMP using column generation, (24)–(30).
  for all scenarios $s \in S$ **do**
    Solve the PBSP-s using column generation, (14)–(22).
    if condition (31) is not satisfied **then**
      if solution to PBSP-s is degenerate **then**
        Use the Magnanti–Wong method to find the Pareto optimal Benders' cut.

```
      end if
      Add cut to the BMP, of type (23).
    end if
  end for
until condition (31) is satisfied, ∀s ∈ S.
PHASE 2
Reintroduce the integrality requirements for
  the BMP.
Retain all cuts that have been added in PHASE 1.
repeat
  Solve the BMP using column generation, (24)–(30).
  for all scenarios s ∈ S do
    Solve the PBSP-s using column generation,
      (14)–(22).
    if condition (31) is not satisfied then
      if solution to PBSP-s is degenerate then
        Use the Magnanti–Wong method to find the
          Pareto optimal Benders' cut.
      end if
      Add cut to the BMP, of type (23).
    end if
  end for
until condition (31) is satisfied, ∀s ∈ S.
```

To demonstrate the benefit of the RRTAP we take the integral BMP solution and evaluate the recovery costs by solving the PBSP-s, $\forall s \in S$, to integrality. Given the structure of the two-phase algorithm, it is possible to evaluate the solution at the completion of both phases. Using the solution to the BMP at the completion of phase 1 can provide a good upper bound for the problem. To find this upper bound, the BMP is solved once to integrality using all the cuts added to the BMP by the completion of phase 1. This provides a planning solution to evaluate by solving for all scenarios using the PBSP-s. For all scenarios s, the PBSP-s is then solved to integrality to find the current best recovery solution to the RRTAP. The cuts that are added in phase 2 tighten this upper bound, however in §4 we demonstrate that phase 1 bound is very close to the optimal solution.

It is not necessary to include the Magnanti–Wong method (Magnanti and Wong 1981) in Algorithm 1, however we have found great computational benefit from its use each time the PBSP-s is solved as demonstrated in §4.3. Another possible technique to improve the computational performance of the Benders' decomposition algorithm is to implement the independent Magnanti–Wong method (Papadakos 2008), which is solved to find cuts independent of the subproblem solution. In Papadakos (2008), the benefit of using the independent Magnanti–Wong method is to generate initial cuts that provide a good lower bound from the master problem without having to solve the Benders' subproblems. Because our problem requires a large number of cuts to find the optimal solution, the addition of one initial cut does not create a significant enough improvement in the BMP. As a result we have chosen not to include the independent Magnanti–Wong method (Papadakos 2008) and only implement the original Magnanti–Wong method (Magnanti and Wong 1981).

3.4. Branching Rules

Three different branching rules have been implemented for this problem, two are designed to eliminate the fractional solutions from the optimal LP relaxation and a third is to break symmetry in the BMP. The former of these rules is based on Ryan/Foster branching (Ryan and Foster 1981) because this technique handles the 0-1 binary variables in a more meaningful manner than variable branching. The first branching rule implemented for this problem is similar to the branching performed in Barnhart et al. (1998), branching on the most fractional pair of flights forming a feasible connection. Given a set of fractional variables, we search over all variables to find the most fractional flight connection and identify a flight pair to branch on, (i^*, j^*). The left branch forces the flight connection to be used, $(i^*, j^*) \in p$ if i or j are contained in p, and the right branch requires the connection is not used, $(i^*, j^*) \notin p$, for all aircraft. However, on the right branch it is not a requirement that the two flights do not exist on the same string for that aircraft, so $(i^*, k) \in p$, $k \in N_{\text{legs}}$, and $(l, j^*) \in p$, $l \in N_{\text{legs}}$, where $k \neq j^*$ and $l \neq i^*$ is permissible. This method of branching is called branching on follow-on's.

Algorithm 2 (Aircraft/Arc branching)

```
Let i be the position indicator for the current flight.
Given a pair of flight strings p1 and p2 for aircraft r*,
set i to the position of the first flight in both strings,
  i ← 0,
set the divergence point d to the starting position,
  d ← 0, this assumes that the two strings only
  share the source node.
Initialise arc1 and arc2 as empty flight strings.
while i < length of p1 and i < length of p2 do
  if flight at position i in string p1 ≠ flight at
    position i in string p2 then
    Set the divergence point d ← i.
    Exit loop.
  end if
  Increment i by 1.
end while
for all flights from position d to the length of p1 do
  append flight to arc1.
end for
for all flights from position d to the length of p2 do
  append flight to arc2.
end for
```

On the left branch, exclude all connections in arc_1 for aircraft r^*,
on the right branch, exclude all connections in arc_2 for aircraft r^*.

The second of the branching rules, which we define as aircraft/arc branching, excludes strings of flights being allocated to a particular aircraft. This method of branching has been developed from the rule presented in Barnhart, Hane, and Vance (2000) for multicommodity flow problems. In their problem, the commodity to branch on was determined by the largest amount of flow; in our case each of our commodities have the same amount of flow, hence the method requires a variation. We search over all fractional variables to find the two most fractional for each aircraft. Comparing the sum of the fractional values from the pair of variables for each aircraft, the largest sum indicates the aircraft r^* and the fractional variables to branch on. Each variable in the pair represents a string of flights with some of the flights common between the two strings. There is at least one common node between the strings because there is only one origin for each aircraft, hence the source node is always common. By ordering the flights in both strings by their position, the divergence point is identified as the earliest position that contains a different flight between the two strings. Two new strings of flights are identified from the divergence point to the sink node and arc_1 and arc_2 are defined to contain the connections between these flights in these two strings, respectively. The branching is performed by excluding aircraft r^* from using the connections contained in arc_1 on the left branch, and arc_2 on the right branch. A detailed description on identifying which arc_1 and arc_2 to branch on is presented in Algorithm 2.

With the two branching rules implemented there must be a priority assigned to each specifying when they should be used. In our model we assign a higher priority to the follow-on branching than the aircraft/arc branching. We assign the priorities in this way because the follow-on branching takes a more global view of the problem, branching on pairs of flights for all aircraft. At the point that no valid follow-on branching exists, then we exclude strings of flights from use by each aircraft.

As mentioned in §3.2, we have implemented a third branching rule to help eliminate the symmetry in the master problem. This branching rule searches all of the fractional variables looking at the starting flights to identify a particular flight that is used by two different aircraft. We define $V_j^r = \{v \in V \mid r^v = r, s^v = j\}$, where V is the set of all fractional variables and r^v and s^v are the aircraft and starting flight for variable v, respectively. The fractionality of a particular aircraft pair and starting flight is calculated by $f_j^{r_1 r_2} = \sum_{v \in V_j^{r_2}} v$, if $V_j^{r_1} \neq \varnothing$. We branch on the tuple (r_1, r_2, j) with the largest $f_j^{r_1 r_2}$, excluding r_1 from starting with flight j on the left branch and excluding r_2 from starting with flight j on the right branch. This branching rule is assigned the highest priority forcing the earliest branches to break the symmetry of the master problem.

4. Computational Experiments

To test the effectiveness of using the recoverable robustness technique we compare the difference in cost for a simulated recovery scenario between our solution and a representative proxy robust algorithm. We develop a proxy robust algorithm using the connection cost of Grönkvist (2005) in defining the cost of a flight string c_p^r. The new proxy robust model consists of the objective function $\sum_{r \in R} \sum_{p \in P^r} c_p^r y_p^r$ and the constraints (2)–(4), and (10) is solved to find an optimal tail assignment. Because the aircraft routing and, by extension, the tail assignment problem is a feasibility problem, the selection of a cost function is used only as a proxy to favour specific connection lengths and improve robustness. The solutions found using the Grönkvist connection cost function and the RRTAP are just two of a large number of feasible solutions to the tail assignment problem. As such, there are potentially many other feasible solutions that have better and worse recoverability than the two compared here. Therefore, the Grönkvist connection cost function is selected only as an example of proxy robust approaches and to demonstrate the improvements in recoverability that can be achieved with the RRTAP.

The connection cost function presented in Grönkvist (2005) attempts to improve the robustness of tail assignments by assigning costs to flight connections based on their length. We illustrate the connection cost function implemented for our representative proxy robust model and in the first stage of our recoverable

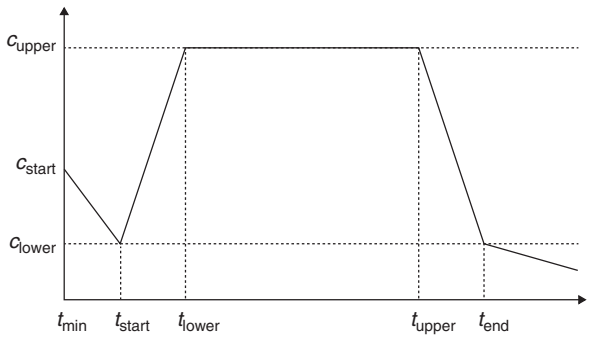

Figure 1 Connection Cost Function Presented in Grönkvist (2005)

Notes. Time parameters (minutes): $t_{min} = 40$, $t_{start} = 120$, $t_{lower} = 180$, $t_{upper} = 300$, $t_{end} = 360$. Cost Parameters ($ AUD): $c_{start} = 500$, $c_{lower} = 100$, $c_{upper} = 5{,}000$.

robust model in Figure 1. The motivation for this form of connection cost function is related to the potential of propagated delay and recovery possibilities. Grönkvist argues that very short connections, $t_{\min} \leq t \leq t_{\text{start}}$, although ideal in regards to aircraft utilisation, are more prone to propagating delay, and suggests a compromise ideal connection length of $t = 120$. The medium length connections, $t_{\text{lower}} \leq t \leq t_{\text{upper}}$, are penalised heavily because they do not provide high enough utilisation and are too short to service extra flights in a recovery situation. The long connections, $t \geq t_{\text{end}}$, are also favoured because in a recovery situation the aircraft can be used to service additional flights within the connection time.

We evaluate the effectiveness of the Grönkvist connection cost function by individually optimising the recovery decisions using the PBSP-s for each disruption scenario s. The resulting cost indicates the recovery performance of the Grönkvist planning solution. We expect the results to already be very good, given the intelligent choice of objective function and exact optimisation of recovery through our model. Indeed, such an exact quantification of the performance of the Grönkvist solution by explicitly determining the optimal recovery strategies and evaluating the recovery costs, has to our knowledge not been carried out. It is against this representative proxy robust model that we review the performance of the RRTAP. We compare the weighted recovery costs and the constituent costs of swaps, cancellations, and delay minutes in our experiments.

We mention in §3 that it is possible to use any formulation for the planning tail assignment problem. The Grönkvist solution used in our experiments is a proxy robust model based on a connection cost function. Without much further work the RRTAP could be reformulated to use any proxy robust planning model in the BMP and improve the recoverability of that model through intelligent allocation of flights to aircraft. For example, the RRTAP provides great flexibility (i) in the models used for the planning and recovery problems, and (ii) the data and flight networks used in the solution.

A major benefit to the RRTAP is that the weighted recovery cost of the final solution will be no worse than that of the solution to the model used in the BMP. So there is always potential benefit in applying the recoverable robustness technique to any model.

We implemented this model in C++ and called SCIP 2.0.1 (Achterberg 2009) to solve the integer program using CPLEX 12.2 as the linear programming solver.

4.1. Description of Scenarios and Model Parameters

The test data for this model consists of 53 flights with 341 feasible connections in a domestic network operating with three major airports, serviced by 10 aircraft. Two different types of disruption scenarios have been implemented, airport closures and aircraft grounding, resulting in 102 scenarios. The specifics of the disruption scenarios are presented in Table 2.

Table 2 Disruption Scenarios Used

Type	Affected	Start time	Duration	Weight w^s (%)	Scenarios
Airport closure	One scenario for each major airport	6 A.M.	180 min	0.07	0–2
			300 min	0.03	3–5
		12 P.M.	180 min	1.4	6–8
			300 min	0.6	9–11
Aircraft grounding	One scenario for each aircraft	6 A.M.	60 min	3.5	12–21
			120 min	0.7	22–31
			240 min	0.14	32–41
Aircraft grounding	One scenario for each aircraft	12 P.M.	60 min	3.5	42–51
			120 min	0.7	52–61
			240 min	0.14	62–71
Aircraft grounding	One scenario for each aircraft	5 P.M.	60 min	3.5	72–81
			120 min	0.7	82–91
			240 min	0.14	92–101

We estimate the relative probability of each of the previous scenarios occurring in a single day and encode these probabilities as the weights w^s in Equations (1) and (24). There are a number of different disruptions that could affect the operations of an airline, and within our model we only include a subset of them. Given that each of these scenarios are not mutually exclusive, the sum of the probabilities assigned to each scenario do not equate to 1. To determine the probability of an airport closure we assume that this schedule is a summer schedule, so it is more likely for an afternoon closure to occur than a morning closure. In the summer there is very little chance of fog, which is the main contributor to morning airport closures. Also in the summer, severe storms, characterised by high winds and regular lightning strikes, generally occur in the afternoon. We estimate that in a season, a single airport may experience an afternoon closure approximately three–four times, so we assign a daily probability of 2% for a closure of any length. Similarly, we expect that there is little chance of a morning closure during the summer season so we assign a probability of 0.1% for a closure of any length. Further, we have estimated that in the case of an airport closure there is a 70% chance that it will last for 180 minutes, and a 30% chance that it will last for 300 minutes. For example, we estimate that an afternoon airport closure for 300 min will occur with a probability of $2\% \times 30\% = 0.6\%$.

An aircraft grounding could be attributed to a number of different factors, including technical issues, delays in the cleaning of an aircraft, or baggage loading/unloading issues. The scenarios represent the situation when an aircraft is not ready for a scheduled departure. Using data for U.S. airlines published

by the Bureau of Transportation Statistics (Research and Innovative Technology Administration 2011b), in August 2011, approximately 18% of all flights were delayed and out of all flight delays approximately 26% were caused by factors in the airline's control, so $18\% \times 26\% = 4.68\%$ of all flights were delayed because of airline factors. We assume that an aircraft grounding causes flight delays because of airline factors, so for a single aircraft grounded for 60, 120, and 240 minutes we assign the probabilities of 3.5%, 0.7%, and 0.14%, respectively. Now, $3.5\% + 0.7\% + 0.14\% = 4.34\%$, which is less than 4.68%, the percentage of all flights delayed because of airline factors, and this difference occurs because we are approximating the rate of delay.

To determine the lowest cost recovery solution we have assigned costs for each minute delayed and for flight cancellations. As mentioned in §2, we aim to use actual costs in evaluating our model. It is very difficult to determine the actual costs for both delays and flight cancellations because there is an unknown component of lost revenues. The delay costs are set at \$100 AUD per minute for a full aircraft, a figure based on the EUROCONTROL report by Cook, Tanner, and Anderson (2004) that estimates the cost of delays at €74 per minute. The cancellation cost per passenger for all flights in the network is estimated using an average ticket price of \$350 AUD multiplied by a lost revenue parameter or "loss rate." The loss rate would be an airline specific value indicating the expected amount of passenger recapture after a cancellation. A loss rate less than one indicates that a percentage of passengers are recaptured by rebooking themselves on another provided flight, whereas a value greater than one represents the loss of all passengers and possible future bookings with the airline. The inclusion of a loss rate is an attempt to capture the direct and indirect costs, such as lost revenues and loss of goodwill, respectively, associated with the cancellation of a flight. Because the loss rate is very difficult to estimate we have presented our results using a set of values ranging from 0.01 to 3 $(0.01, 0.125, 0.25, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0)$ to provide a broad test of our model. Both the average ticket price and loss rate could be flight specific and the implementation of this is trivial.

In determining the cost of flight delays and cancellations we assume that the aircraft are at 75% capacity. Because the aircraft is not booked to capacity we have developed a simple method for calculating the cost of cancellations for each individual flight. We assume that one-third of the passengers on the cancelled flight will be rebooked to the next available flight with the same O-D pair at a cost of 25% of \$100 AUD per minute to the next departure. This cost of rebooking passengers onto the next flight is simply the cost of the delay experienced in waiting for the next departure. The rest of the passengers on the cancelled flight do not get rebooked and as a result the revenue is lost. The revenue that is lost from the cancelled flight is calculated from this proportion of passengers. Because the passengers are not accommodated on the next available flight, there is the option of rebooking themselves with this airline or another. This rebooking process is captured in the lost revenue parameter, the "loss rate." Also, the delay cost per flight given in the EUROCONTROL report (Cook, Tanner, and Anderson 2004) is based on a full aircraft, and given that we are assuming a 75% capacity, the delay cost of an aircraft per minute in our model is \$75 AUD.

As mentioned in §2 we handle the flight delays by including a set of flight copies in the recovery network. For our experiments we have used a maximum allowable delay of 180 minutes, with flight copies for every 30 minutes. Given that the delay increment is 30 minutes, we will be overestimating the delay costs, because many shorter feasible connections exist in the delay window for each flight. Greater granularity is possible by decreasing the delay copy increment, however this degrades the computational performance with the addition of more flight copy arcs to the connection network.

4.2. Comparison of Recoverable Robust Solutions and Grönkvist Solution

The RRTAP attempts to find a planned solution that requires minimal changes in recovery from disruptions. To represent the difficulty faced by operations controllers to reroute aircraft we use a swap penalty parameter g^s. This penalty parameter can be likened to a cap on the number of allowable changes during a disruption, however a cap is more restrictive than a penalty. In our model, low swap penalties result in more changes made in the recovered solution, which provides a lower recovery cost. We have found that the lower recovery cost occurs because greater flexibility is allowed in the model, so it is possible to reroute more aircraft to avoid costly delays and cancellations. As a review of the different trade-offs and outcomes, we present our results with swap penalties in the range $10 \leq g^s \leq 10{,}000$ $(10, 500, 1000, 2500, 5000, 10000)$. A swap penalty of 10 illustrates the case where there is virtually no penalty for swaps. It is also trivial to implement this parameter with different values for each flight or aircraft in the model.

We are using the Grönkvist connection cost function in the BMP of the recoverable robust model. With this we investigate whether our algorithm can provide superior recovered solutions when compared to the proxy robust model. The relative performance between the two robust tail assignment models is

Table 3 **Percentage Difference Between the Grönkvist Solution (x) and the Recoverable Robust Solution, Phase 2 (y) and Phase 1 (z), $((y-x)/x$ (%), $(z-x)/x$ (%))**

	Penalty				
Loss rate	500	1,000	2,500	5,000	10,000
0.01	(−1.32, −0.19)	(−1.8, −1.8)	(−1, 0.55)	(−0.73, −0.73)	(−0.71, 0.67)
0.125	(−1.73, −1.5)	(−1.46, −0.71)	(−0.2, 0.23)	(−0.66, −0.51)	(−0.62, −0.39)
0.25	(−1.27, −0.99)	**(−14.83, −14.66)**	(−1.21, −**7.65**)	(−1.73, −1.73)	(−1.71, −1.71)
0.5	(−1.08, −**9.97**)	**(−14.54, −14.4)**	**(−12.83, −12.8)**	(−1.37, −0.99)	(−1.3, −0.5)
1	(−0.96, −0.73)	**(−12.83, −12.79)**	**(−13.26, −13.19)**	**(−5.95, −5.57)**	(−0.97, −0.93)
1.5	(−0.86, −**7.64**)	**(−11.68, −11.65)**	**(−12.33, −12.22)**	(−**5.17**, −0.67)	(−0.99, −0.37)
2	(−0.78, −0.47)	**(−10.73, −10.73)**	**(−11.51, −11.45)**	(−4.91, −4.98)	(−1.16, −0.81)
2.5	(−0.72, −0.64)	**(−9.93, −9.93)**	**(−10.8, −11.23)**	(−4.75, −4.7)	(−0.9, −0.38)
3	(−0.66, −0.39)	**(−9.23, −9.13)**	**(−10.18, −10.18)**	(−4.38, −4.44)	(−0.89, −0.89)

evaluated using the weighted sum of the recovery costs over all scenarios. This is defined as

$$WeightedCost = \sum_{s\in S} w^s \left\{ \sum_{r\in R} \sum_{p\in P^{sr}} c_p^{sr} y_p^{sr} + \sum_{j\in N} d_j z_j^s + \sum_{r\in R} \sum_{j\in N} g^s \epsilon_{jr}^{s-} \right\}, \qquad (43)$$

which is the second term in the objective function (1). We use Equation (43) to calculate the weighted simulated recovery cost using the solution from the proxy robust model and the solution at the completion of phase 1 and 2 of the two-phase algorithm, Algorithm 1. We compare the weighted recovery costs calculated with different cancellation loss rates and penalty weights. For all parameter sets, the *weighted* recovery cost of the recoverable robust solution at the completion of phase 2 either equals or improves upon the proxy robust solution that is demonstrated in Table 3.

Table 3 presents the percentage improvement in the weighted recovery costs at the completion of phase 1 and 2. The largest improvement in the weighted recovery costs achieved at the completion of phase 2 for a specific penalty weight occurs with $g^s = 1{,}000$ and a loss rate ≥ 0.25; a 14.83% cost reduction is achieved with a loss rate equal to 0.25. There is an apparent nonlinear relationship between recoverable robustness improvement and penalty weight, and the greatest improvement occurs with weights in the range $1{,}000 < g^s < 5{,}000$. In that range and with a loss rate ≥ 0.25 the RRTAP achieves an average improvement over the proxy robust solution of 8.77%, with a minimum improvement of 1.21%. The most important practical feature of the RRTAP is the ability to reduce the weighted recovery costs when compared to the proxy robust model, and in general to any planning tail assignment model. Because the tail assignment can be formulated as a feasibility problem, any improvement in the recoverability of the planned solution is made at no extra cost.

There are many methods of weighting the length of connections to achieve a desired tail assignment solution. The Grönkvist connection cost function is an example of this; it attempts to avoid propagating delays and create recovery opportunities. To demonstrate the relative performance of alternative connection cost functions against the Grönkvist function, we have solved the tail assignment problem using linear (t), quadratic (t^2), square root ($\sqrt{t}$), and hyperbola ($1/t$) functions. The results in Table 4 present the number of test cases where the selected function dominates Grönkvist and the average relative difference in the weighted recovery cost. The set of test cases, 54 in total, contains all of the experiments used to create Table 3 including the experiments with a penalty equal to 10. Each of the functions presented in Table 4 construct a tail assignment with different characteristics, for example the linear, quadratic, and square root functions favour shorter connections compared to the hyperbola function that favours long connections. Although the results presented in Table 4 show that the linear and square root functions dominate Grönkvist in most test cases, Grönkvist outperforms the quadratic and hyperbola functions. Table 4 also shows that the range of mean relative differences is not particularly large, even over this collection of disparate cost functions.

The RRTAP could be solved using *any* connection cost function as the master objective, and one would be guaranteed to obtain a planned solution better than or equal to the solution obtained using that particular proxy robust cost function. Because the Grönkvist function has appeared in the literature to solve the tail assignment problem (Grönkvist 2005, 2006) we choose to solve RRTAP relative to this function.

The results in Table 3 demonstrate that the RRTAP improves upon the weighted recovery costs of the proxy robust solution in most cases. To illustrate the performance of the RRTAP solution when compared to the proxy robust solution for each individual recovery scenario we have selected four representative cases: (i) penalty = 500, loss rate = 0.5; (ii) penalty = 1,000,

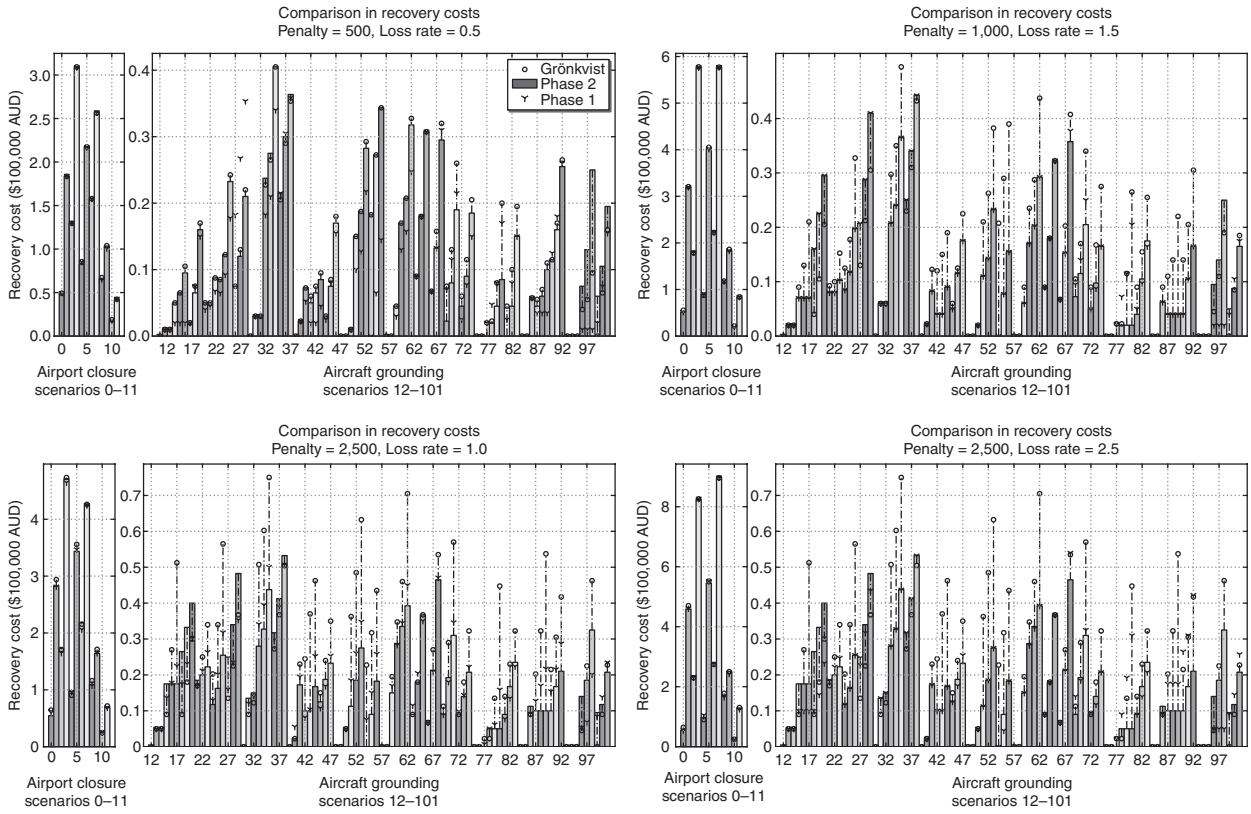

Figure 2 Individual Recovery Costs for All 102 Scenarios—Comparison Between Grönkvist, Phase 1, and Phase 2 Results

Note. The order for the scenarios is presented in Table 2.

Table 4 Analysis of Connection Cost Functions

Function	Linear	Quadratic	Square root	Hyperbola
Number of cases dominating Grönkvist (/54)	38	15	47	15
Average relative difference (%)	−3.62	0.62	−5.74	0.62

Note. The constructed tail assignment is evaluated against 54 test cases using different penalties and loss rates.

loss rate = 1.5; (iii) penalty = 2,500, loss rate = 1; and (iv) penalty = 2,500, loss rate = 2.5. Figure 2 presents the individual recovery costs in each scenario for the proxy robust phase 1 and 2 solutions. Comparing the results of the RRTAP and the proxy robust solution we find much more improvement variability in the individual recovery costs across the scenario set. The individual recovery costs for each scenario s are calculated by

$$IndividualCost(s) = \sum_{r\in R}\sum_{p\in P^{sr}} c_p^{sr} y_p^{sr} + \sum_{j\in N} d_j z_j^s + \sum_{r\in R}\sum_{j\in N} g^s \epsilon_{jr}^{s-}, \quad (44)$$

which is presented for all cases in Figure 2. We have selected two types of scenarios for use in this model, airport closure and aircraft grounding, as explained in Table 2. An airport closure has a much greater impact on an airlines operations than an aircraft grounding because a large number of flights and aircraft are potentially affected. In Table 5 we present the number of flights that are affected in the airport closure scenarios. The flights are classified "affected" if they are scheduled to arrive at, or depart from, the closed airport within the closure time. Comparing Table 5 with Figure 2 it is clear that the weighted recovery cost is dependent on the number of affected flights. Because there is a much greater recovery cost associated with an airport closure, compared to an aircraft grounding, we have separated out their costs in Figure 2. It is difficult to determine the affected flights for the aircraft grounding scenarios because this is dependent on the planned routing that we are optimising.

Although the RRTAP solution may perform better across the sum of the individual recovery costs than the proxy robust model, there are a number of scenarios where it performs worse. The largest relative improvements occur for the aircraft grounding scenar-

Table 5 Number of Flights Affected in the Airport Closure Scenarios

Scenario	0	1	2	3	4	5	6	7	8	9	10	11
Flights affected	2	8	8	16	5	10	8	13	4	5	1	1

ios. This demonstrates that for the larger disruption scenarios, which involve more flights and aircraft, the planned routing does not have much impact on the recovery costs. The results in Figure 2 show that the improvement in the sum of the recovery costs is attributable to an improvement in the cost for a large proportion of the individual scenarios. This is quite important because we are attempting to improve the recoverability of the tail assignment for a broad range of disruptions, so if the improvements were restricted only to a few scenarios the efficacy of this technique would be reduced.

The RRTAP is a robust model that is closely aligned with stochastic programming. Because we are applying a weight to the recovery cost for each scenario, the objective of this model attempts to minimise the weighted recovery cost for the given scenario set. There are other robust modelling formulations that can be applied to the tail assignment problem, each with their own emphasis and advantages and disadvantages. For example, one may wish to minimise the maximum recovery cost across the entire scenario set. Such a model, which we will label the "pure robust" tail assignment problem (PRTAP), can be easily formulated from the model described by (1)–(13) with a change in the objective function and the addition of a set of constraints. To formulate the PRTAP we modify the objective function by removing all variables relating to the recovery subproblems. To minimise the maximum recovery cost we introduce the variable $\mu^{\max}$ to the objective function, representing the upper bound on the recovery costs from all scenarios. To enforce this upper bound we add a set of constraints that require $\mu^{\max}$ to be greater than or equal to the recovery solution cost for all scenarios. We define the pure robust model as

(PRTAP)

$$\min \left\{ \sum_{r\in R} \sum_{p\in P^r} c_p^r y_p^r + \mu^{\max} \right\}, \tag{45}$$

$$\text{s.t. constraints (2)–(13)} \tag{46}$$

$$\mu^{\max} \geq \sum_{r\in R} \sum_{p\in P^{sr}} c_p^{sr} y_p^{sr} + \sum_{j\in N} d_j z_j^s + \sum_{r\in R} \sum_{j\in N} g^s \epsilon_{jr}^{s-} \quad \forall s \in S, \tag{47}$$

$$\mu^{\max} \geq 0. \tag{48}$$

The PRTAP decomposes between the planning and all recovery scenario variables and Benders' decomposition may be applied. This model is identical to the RRTAP, with the addition of the maximum recovery cost variable $\mu^{\max}$ to the objective function and the inclusion of additional constraints (47) to enforce the maximum recovery cost across the complete scenario set.

Because of the differing objectives, the recoverable robust (RRTAP) and pure robust (PRTAP) formulations should produce different results. Given these distinct objective functions, it is very difficult to identify a metric that provides a fair comparison between the two models. Taking this into account, both the weighted recovery costs, as calculated by Equation (43), and the 90th percentile recovery cost over all scenarios for the RRTAP and the PRTAP is presented in Figure 3. Across all parameter sets, the difference in the maximum recovery costs between the two models is within the range of −3.73%–6.05%, with an average of 1.04% indicating only a small improvement in the PRTAP over the RRTAP. Because there is little variation in the maximum recovery costs, the 90th percentile has been selected for Figure 3 to demonstrate any difference between the two models. The results show that the RRTAP outperforms the PRTAP in terms of the weighted recovery costs for all selected cases, which is to be expected because this value is a term in the objective function of the RRTAP. Also, in comparing these two models by the 90th percentile, the PRTAP only improves upon the RRTAP in three of the selected cases and only with a swap penalty of 10,000. From these results it is clear that in a feedback robust model using a full recovery problem, the minimisation of the weighted cost results in an acceptable reduction in the magnitude of all recovery costs.

The runtimes for the PRTAP provide some interesting results when compared to the RRTAP. Our experiments for the RRTAP were run to a maximum time of three hours to ensure that a solution could be found in a reasonable time frame. We find that, on average, the PRTAP requires 4.01 times as long as the RRTAP to find the optimal solution, with average runtimes of 7,371.18 and 3,093.93 seconds for the PRTAP and RRTAP, respectively. We provide a greater analysis of the runtime for the RRTAP in §4.3.

4.3. Behaviour of Solution Runtimes

The RRTAP is a large-scale model for which a number of enhancement techniques have been applied to improve the solution runtime. In §3 we documented the techniques that have been implemented to solve RRTAP. The key feature of the solution methodology is the use of Benders' decomposition and column generation, which are commonly used in solving large-scale optimisation problems. To improve the runtimes even further we introduced the Magnanti–Wong method (Magnanti and Wong 1981), as illustrated in §3.1.1, to generate Pareto optimal Benders' cuts. We also implemented a number of branching rules described in §3.4 for use in phase 2 of the

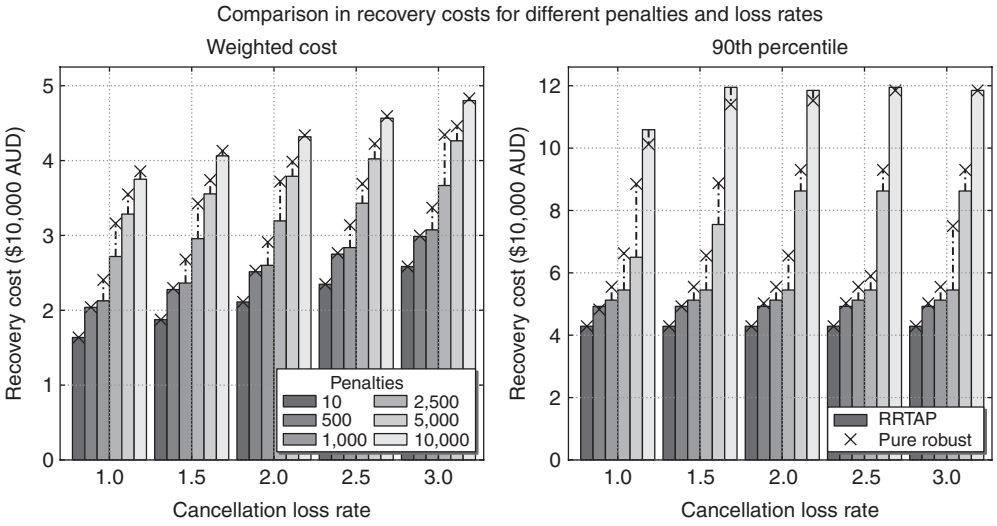

Figure 3 Weighted and 90th Percentile Recovery Costs—Comparison Between the RRTAP and a Pure Robust Formulation

two-phase algorithm. In Figure 4 we present the runtimes for the RRTAP using different algorithmic enhancements on the four cases selected for Figure 2. The different results presented in Figure 4 are defined as (i) including all enhancements presented in §3; (ii) including only the Magnanti–Wong method (Magnanti and Wong 1981) with the default branching rules in the SCIP 2.0.1 distribution; (iii) using the branching rules described in §3.4 without the Magnanti–Wong method; and (iv) a standard Benders' decomposition formulation, i.e., without the Magnanti–Wong method and with the default branching rules. The expectation is that the phase 1 runtimes are similar for the pairs of results (i, ii) and (iii, iv), however, it is possible to have fluctuations in the computational experiments. We also expect that the phase 2 runtimes are similar for the result pairs (i, iii) and (ii, iv). There is a greater chance of a variability in the phase 2 runtimes because the Magnanti–Wong method generates different cuts to the standard Benders' decomposition approach. The addition of Magnanti–Wong cuts generate a different problem by the end of phase 1, which affects the difficulty of the integer program and the number of branches required.

We find that the runtimes for the RRTAP using all enhancements are significantly lower than when none are used. This is to be expected, especially in regards to the branching rules, because the rules included in the SCIP 2.0.1 distribution do not use any problem specific information. The inclusion of the branching rules has the effect of reducing the amount of time spent in phase 2 of the two-phase algorithm. By using the Magnanti–Wong method, we are able to reduce the amount of time spent in phase 1 of the two-phase algorithm, however it does not provide a marked improvement in phase 2 when run with the SCIP 2.0.1 branching rules, and in two cases it is worse. Even though solving the auxiliary problem for the Magnanti–Wong method adds extra runtime to each iteration, it is clear that this method genuinely improves the generated cuts. Figure 4 demonstrates that although each enhancement can improve the runtimes for the algorithm, it is necessary to include all enhancements to achieve the greatest runtime improvement.

Although the enhancements developed for this algorithm have a significant effect on the runtimes, we found that the parameters used can also have an affect. Figure 5 illustrates the runtimes required to calculate the results presented in Table 3 for the RRTAP. In our experiments we limited the runtimes to three hours. In the cases where the runtimes exceeded three hours, the model was terminated after the first run of the Benders' decomposition algorithm that completes after the three-hour time limit. Given that the Benders' subproblems' solutions, the upper bounds, are not strictly nonincreasing, the best solution found during the runtime is used in the calculations of Table 3. In Table 3 we see that there is little difference between the optimal solution at the completion of phase 2 and the upper bound calculated at the completion of phase 1. Figure 5 shows that in some cases the time spent in phase 2 attempting to find the optimal solution, which is not always found, can be quite significant. To achieve a fast solution that is close to optimal, one could simply complete only phase 1 of the two-phase algorithm.

4.4. Investigation of Constituent Recovery Costs

The recovery costs calculated from the evaluation of the proxy robust and RRTAP models can be broken down into their constituent costs for swaps, delays, and cancellations. Figure 6 presents this breakdown,

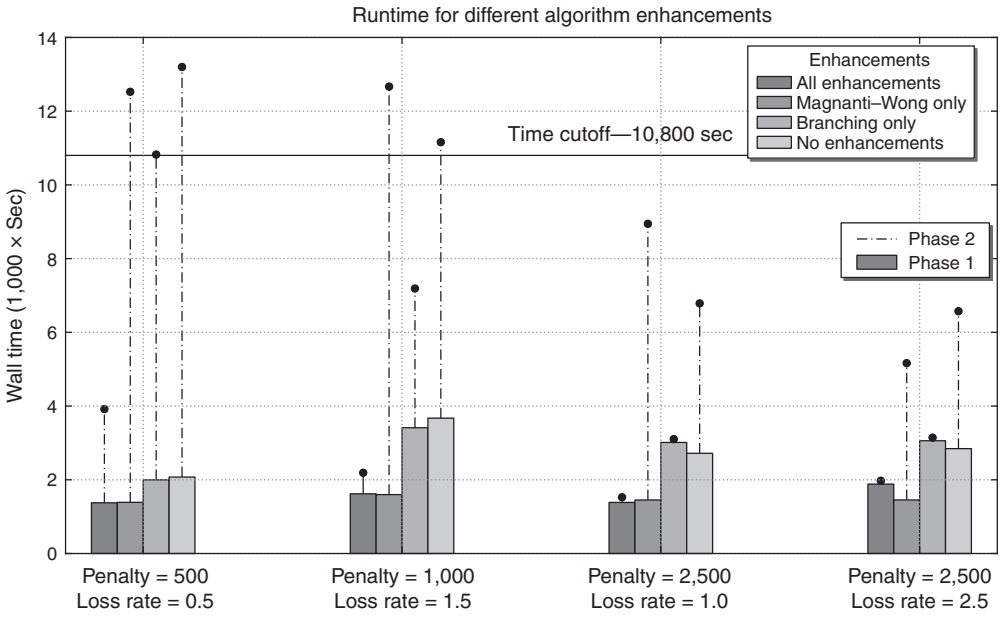

Figure 4 Runtime with the Use of Different Algorithm Enhancements

Note. The dots represent the total runtime.

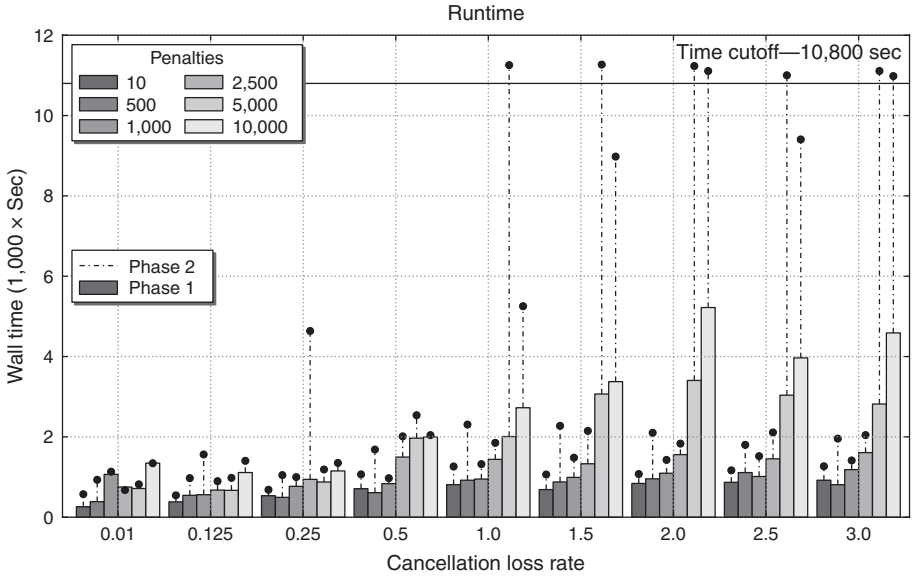

Figure 5 Runtime for the Recoverable Robust Model

Note. The dots represent the total runtime.

demonstrating the trade-offs that can occur when setting the model parameters. In §4.2 we have demonstrated that the recoverable robust solution either equals or improves on the weighted recovery cost of the proxy robust solution. Given that the model optimises the weighted recovery costs, it is possible for the proxy robust solution to outperform the recoverable robust solution for individual recovery policy costs. One such example is the swap costs with a penalty weight of $g^s = 5{,}000$, where the number of swaps in the proxy robust solution is *lower* than the recoverable robust solution. However, for the same penalty weight of $g^s = 5{,}000$, the recoverable robust solution significantly outperforms the proxy robust result for the *delay* costs. For other penalty weights, such as $g^s = 1{,}000$ and $g^s = 2{,}500$, the results are quite varied with the improvement being attributed to a decrease in swaps and delays, respectively. This is attributable to the minimum length of delay for an individual flight and its associated cost. Because the flight delays are discretised to be every 30 minutes from the original departure, the minimum cost of delay is \$75 AUD ×

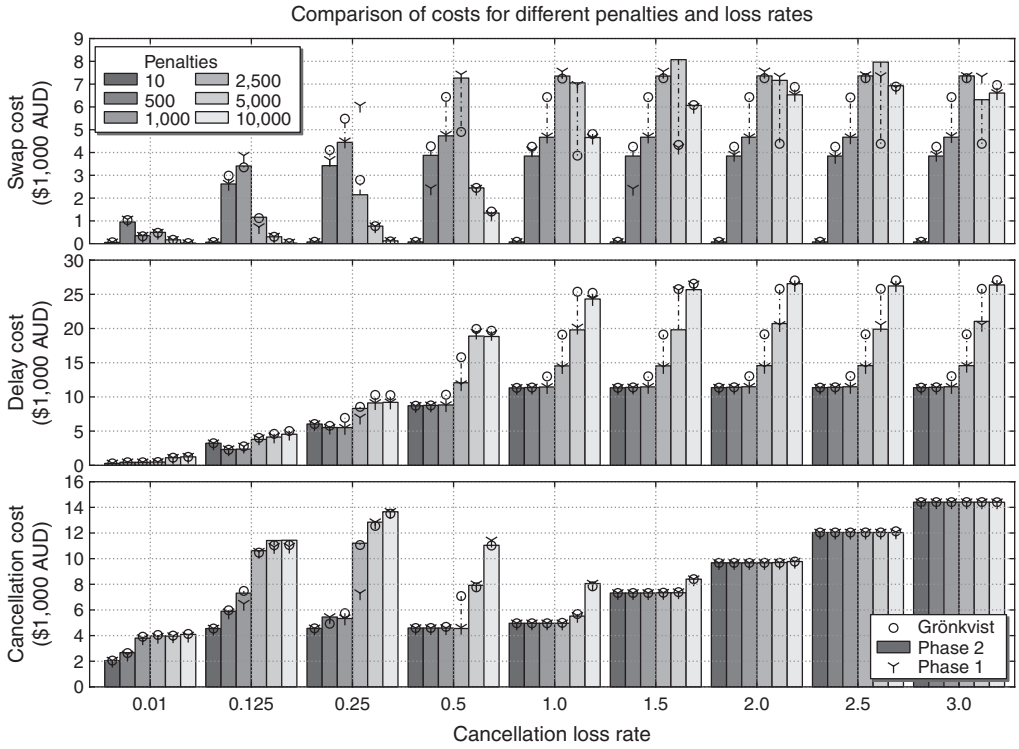

Figure 6 **Swap, Cancellation, and Delay Costs—Comparison Between Grönkvist, Phase 1, and Phase 2**

30 minutes = \$2,250 AUD. For the proxy robust solution, the tendency is to allow more swaps, rather than to delay flights for penalty weights less than 2,250, and for weights greater than 2,250 to allow more flight delays than swaps. Thus, in the case of the penalty weight of $g^s = 5{,}000$ we find that the preferred recovery policy for the proxy robust solution is to delay flights, so any improvement from the RRTAP will be found through reducing the number of delayed flights.

In Table 3 it is possible to see that the greatest weighted improvement in the recovery costs occurs when $g^s = 1{,}000$ and $g^s = 2{,}500$. Above we have presented the case where $g^s = 5{,}000$, the benefit is attributable to a decrease in the total *delay* costs. This is combined with an increase in the total number of *swaps* performed in the recovery process. The largest weighted recovery cost improvement occurs when $g^s = 2{,}500$ and we see that this benefit is largely because of a reduction in the *delay* costs. Although the greatest improvement is in the *delay* costs, there is still an improvement in the *swap* costs. As mentioned earlier, we find that the improvement in both costs is because of the minimum cost of delay being close to the penalty weight of $g^s = 2{,}500$. The result of decreased flight delays is particularly important because this directly reduces the number of delay minutes experienced by passengers, which has a real effect on the on-time performance of an airline.

4.5. Effect of Flight Copy Increments

In our model we have used discrete flight copies to handle flight delays for the recovery subproblems. As previously mentioned we have used a delay copy increment of 30 minutes for each flight copy, with a maximum delay of 180 minutes. In Figure 7 we present the weighted recovery costs with the runtime for the RRTAP using different delay increments with a penalty weight $g^s = 2{,}500$ and loss rate of 1.5. As expected, Figure 7 shows that as the delay copy increment decreases, resulting in a larger recovery flight network because of more flight copies, the solution time increases. We also see that there is a great difference in the weighted recovery costs that are calculated using different delay copy increments. Using discrete flight copies in the recovery model has the effect of overestimating the recovery costs by forcing larger delays than may be necessary. Comparing the resulting recovery costs between the smallest delay increment in Figure 7 (15 minutes) and a delay increment of 30 minutes, we calculate the weighted recovery costs of 25,995 and 29,563, respectively. This equates to a 13.73% overestimate of the recovery costs for the given parameter sets. For each of the results in Figure 7 we have omitted the breakdown of the recovery costs for brevity, however we will provide a concise description of our findings. By decreasing

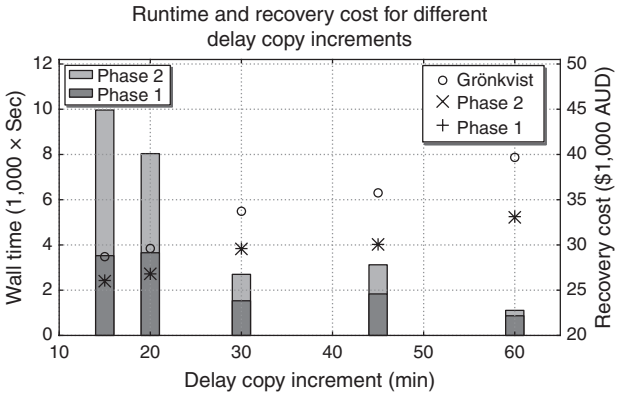

Figure 7 Comparison of Recovery Costs Using Different Delay Copy Increments

Notes. The bars and the points represent the runtimes and recovery costs, respectively. These results have been calculated using penalty weight of 2,500 and loss rate of 1.5. Maximum possible delay is 180 minutes.

the delay copy increment we see different results in recovery components of swaps, delays, and cancellations: the number of delayed flights and number of swaps increases, whereas the number of cancellations and delay minutes decreases. Because there are now more delay options available we find a decrease in the number of cancellations. With the increase in the number of delayed flights and a decrease in the number of delayed minutes, we see that the average delay per delayed flight decreases from 81.6 minutes to 63.9 minutes. These results show that by using discrete flight copies we are overestimating the weighted recovery costs for the model, however by changing the delay copy increment the resulting recovery solutions can change quite dramatically.

5. Conclusions

In this paper we have presented a novel recoverable robustness model for the tail assignment problem. We showed that the solutions to this model guarantee reduced recovery costs and have no increase in planning costs. Further, our model has a full set of recovery decisions, including flight cancellations and delays and aircraft rerouting. The sophisticated recovery subproblems create a complex mixed-integer program, which we are able to solve in a reasonable time frame using enhancement techniques. We have formulated the RRTAP as a stochastic program drawing on solution methods such as Benders' decomposition. By solving this problem as a two-stage stochastic program, it is possible to separate the planning and recovery tail assignment problems. This structure allows for the use of different recovery algorithms and a wide range of different planning algorithms and methods that can be improved upon using the RRTAP.

Through the use of Benders' decomposition and column generation we have demonstrated that RRTAP is able to be solved in an efficient way. The acceleration technique of the Magnanti–Wong method (Magnanti and Wong 1981) has been applied to the Benders' decomposition formulation, which together have provided a significant improvement in the computational performance. Further, we have improved the column generation process for the Benders' decomposition master problem by introducing a branching technique to eliminate symmetry, branching on an aircraft pair, and starting flight.

We have compared the results of the RRTAP against a proxy robust solution developed using a connection cost function presented in Grönkvist (2005). We found this connection cost function provided a simple, but effective, way to introduce robustness into the tail assignment problem. Through the use of our recovery algorithm, we have evaluated this connection cost function to demonstrate the recoverability of the planning tail assignment solution. Further, using this connection cost function in the planning stage of our recoverable robustness algorithm we have been able to develop an even lower cost recoverable planned solution compared to the Grönkvist solution.

With the use of the two-phase algorithm for solving the Benders' decomposition problem we have found that the integral master problem solution using the cuts added by the completion of phase 1 provides a good upper bound on the optimal solution. A large proportion of the runtimes are spent in phase 2 adding cuts to improve the lower bound toward the phase 1 upper bound. Terminating the algorithm at the completion of phase 1 results in near optimal solutions with very fast computational times.

Our results have been presented with a range of values for airline specific parameters, the swap penalties, and lost revenue rates. The method used to assign costs to swaps, delays, and cancellations can have varied effects on the cost benefits from the RRTAP. We demonstrated that because of the value of the minimum delay cost and the penalty weight there can be a tendency toward greater swaps or delays in the final solution. By presenting a range of values we have demonstrated that a trade-off between different recovery costs can be achieved. This allows each airline to value the delays, cancellations, and swaps differently depending on their individual situations.

A common method of reducing the complexity of a recovery model is to use discrete flight copies to handle flight delays. In our analysis we compared the results of using a different number of flight copies, which has shown that the level of overestimation of recovery costs can be high. Further, we find that changing the number of flight copies has an effect on minimum possible delay cost and hence affects the composition of delays and swaps in the final solution. The discovery of a good upper bound at the end of

phase 1 permits a richer result for the RRTAP to be found using a greater number of flight copies without substantial increases in runtimes.

In future work, we aim to extend the recoverable robustness technique for airline problems. With our current formulation it is possible to integrate one or more airline planning stages with the tail assignment problem. Given the high cost of crew to the airline, this is a potentially important aspect to add to this problem. Also, in a recovery situation it is important to monitor passenger flow in the network. Explicit modelling of passengers in the recovery models could provide a more robust set of results.

Acknowledgments

The authors thank the referees for several suggestions that improved the presentation of the manuscript. The second author is supported by the Australian Research Council Centre of Excellence for Mathematics and Statistics of Complex Systems (MASCOS) and an Australian Postgraduate Award. The first author is partially supported by MASCOS.

References

Achterberg T (2009) SCIP: Solving constraint integer programs. *Math. Programming Comput.* 1(1):1–41.

Ageeva Y (2000) Approaches to incorporating robustness into airline scheduling. Master's thesis, Massachusetts Institute of Technology, Cambridge.

Ahuja RK, Magnanti TL, Orlin JB (1993) *Network Flows: Theory, Algorithms, and Applications* (Prentice Hall, Upper Saddle River, NJ).

Barnhart C, Hane CA, Vance PH (2000) Using branch-and-price-and-cut to solve origin-destination integer multicommodity flow problems. *Oper. Res.* 48(2):318–326.

Barnhart C, Boland NL, Clarke LW, Johnson EL, Nemhauser GL, Shenoi RG (1998) Flight string models for aircraft fleeting and routing. *Transportation Sci.* 32(3):208–220.

Borndörfer R, Dovica I, Nowak I, Schickinger T (2010) Robust tail assignment. *Proc. Fiftieth Annual Sympos. of AGIFORS, Nice, France.*

Bureau of Infrastructure, Transport and Regional Economics (2011) Domestic airline on time performance. Technical report, Australian Government—Department of Infrastructure and Transport, Canberra, Australia.

Central Office of Delay Analysis (2011) Digest—Delays to Air Transport in Europe—August 2011. Technical report, EUROCONTROL, Brussels.

Clausen J, Larsen A, Larsen J, Rezanova NJ (2010) Disruption management in the airline industry–concepts, models and methods. *Comput. Oper. Res.* 37(5):809–821.

Cook A, Tanner G, Anderson S (2004) Evaluating the true cost to airlines of one minute of airborne or ground delay. Performance Review Commission, EUROCONTROL, Brussels.

Cordeau J-F, Stojković G, Soumis F, Desrosiers J (2001) Benders decomposition for simultaneous aircraft routing and crew scheduling. *Transportation Sci.* 35(4):375–388.

Dunbar M, Froyland G, Wu C-L (2012) Robust airline schedule planning: Minimizing propagated delay in an integrated routing and crewing framework. *Transportation Sci.* 46(2):204–216.

Eggenberg N (2009) Combining robustness and recovery for airline schedules. Ph.D. thesis, École Polytechnique Fédérale De Lausanne, Lausanne, Switzerland.

Gao C, Johnson E, Smith B (2009) Integrated airline fleet and crew robust planning. *Transportation Sci.* 43(1):2–16.

Grönkvist M (2005) The tail assignment problem. Ph.D. thesis, Chalmers University of Technology and Göteborg University, Göteborg, Sweden.

Grönkvist M (2006) Accelerating column generation for aircraft scheduling using constraint propagation. *Comput. Oper. Res.* 33(10):2918–2934.

Kang LS (2004) Degradable airline scheduling: An approach to improve operational robustness and differentiate service quality. Ph.D. thesis, Massachusetts Institute of Technology, Cambridge.

Kohl N, Larsen A, Larsen J, Ross A, Tiourine S (2007) Airline disruption management–perspectives, experiences and outlook. *J. Air Transport Management* 13(3):149–162.

Lan S, Clarke J-P, Barnhart C (2006) Planning for robust airline operations: Optimizing aircraft routings and flight departure times to minimize passenger disruptions. *Transportation Sci.* 40(1):15–28.

Liebchen C, Lübbecke M, Möhring R, Stiller S (2009) The concept of recoverable robustness, linear programming recovery, and railway applications. Ahuja R, Möhring R, Zaroliagis C, eds. *Robust and Online Large-Scale Optimization*, Lecture Notes in Computer Science, Vol. 5868 (Springer, Berlin), 1–27.

Magnanti TL, Wong RT (1981) Accelerating Benders decomposition: Algorithmic enhancement and model selection criteria. *Oper. Res.* 29(3):464–484.

Mercier A, Cordeau JF, Soumis F (2005) A computational study of Benders decomposition for the integrated aircraft routing and crew scheduling problem. *Comput. Oper. Res.* 32(6):1451–1476.

Papadakos N (2008) Practical enhancements to the Magnanti-Wong method. *Oper. Res. Lett.* 36(4):444–449.

Papadakos N (2009) Integrated airline scheduling. *Comput. Oper. Res.* 36(1):176–195.

Rei W, Cordeau J-F, Gendreau M, Soriano P (2009) Accelerating Benders decomposition by local branching. *INFORMS J. Comput.* 21(2):333–345.

Research and Innovative Technology Administration (2011a) On-time performance—Flight delays at a glance. Technical report, Bureau of Transportation Statistics. Accessed December 3, 2012, http://www.transtats.bts.gov/HomeDrillChart.asp.

Research and Innovative Technology Administration (2011b) Understanding the reporting of causes of flight delays and cancellations. Technical report, Bureau of Transportation Statistics. Accessed December 3, 2012, http://www.bts.gov/help/aviation/html/understanding.html.

Rosenberger JM, Johnson EL, Nemhauser GL (2004) A robust fleet-assignment model with hub isolation and short cycles. *Transportation Sci.* 38(3):357–368.

Ryan DM, Foster BA (1981) An integer programming approach to scheduling. Wren A, ed. *Computer Scheduling of Public Transport: Urban Passenger Vehicle and Crew Scheduling* (North-Holland, Amsterdam), 269–280.

Smith BC, Johnson EL (2006) Robust airline fleet assignment: Imposing station purity using station decomposition. *Transportation Sci.* 40(4):497–516.

Thengvall BG, Bard JF, Yu G (2000) Balancing user preferences for aircraft schedule recovery during irregular operations. *IIE Trans.* 32(3):181–193.

Weide O, Ryan D, Ehrgott M (2010) An iterative approach to robust and integrated aircraft routing and crew scheduling. *Comput. Oper. Res.* 37(5):833–844.

Wu CL (2010) *Airline Operations and Delay Management: Insights from Airline Economics, Networks, and Strategic Schedule Planning* (Ashgate, Farnham, UK).

Yen JW, Birge JR (2006) A stochastic programming approach to the airline crew scheduling problem. *Transportation Sci.* 40(1):3–14.