The Application of Recoverable Robustness to Airline Planning Problems

Stephen J Maher

Supervisor: Associate Professor Gary Froyland

School of Mathematics and Statistics

University of New South Wales

Sydney Australia

A thesis submitted for the degree of Doctor of Philosophy at the University of New South Wales.

August 2013

Abstract

Schedule disruptions commonly affect airline operations and cause a great disparity between the expected and actual operational costs. Many disruption management methods have been proposed to address this disparity, establishing the classes of *proactive* and *reactive* approaches. While the efficacy of *reactive* approaches is greatly affected by the level of recoverability resulting from *proactive* decisions, there is little research regarding the integration of the disruption management classes. Recoverable robustness is one such method that bridges the gap between the *proactive* and *reactive* approaches. This thesis aims to demonstrate the potential recoverability improvements from applying recoverable robustness to airline planning problems.

The recoverable robust tail assignment and aircraft maintenance routing problems are introduced to demonstrate the potential of this framework. These problems are formulated as stochastic programs, which are efficiently solved by integrating column generation and Benders' decomposition. The development of enhancement techniques is required to solve the large-scale optimisation problems resulting from large flight schedules and sets of disruption scenarios. In addition, the aircraft maintenance routing problem introduces a novel modelling approach designed to minimise the effect of disruptions that occur on preceding days.

A general framework for column-and-row generation is developed in this thesis to improve the solution runtime and quality compared to a standard column generation approach. This framework is presented as an alternative solution approach to Benders' decomposition.

An explicit evaluation of column-and-row generation against column generation is performed using the integrated airline recovery problem as an example. This evaluation demonstrates an improvement in solution runtimes and assesses the suitability of employing the integrated airline recovery problem in the recoverable robustness evaluation stage. A novel modelling approach for passenger recovery is also proposed, attempting to improve the evaluation stage feedback. This modelling approach reallocates passengers to alternative flights following flight cancellations, effectively reducing operational costs and increasing passenger flow.

This thesis demonstrates the ability of recoverable robustness to improve the recoverability of various airline planning problems. We show the necessity of the many enhancement techniques developed in this thesis to achieve the best results from applying the recoverable robustness framework.

Acknowledgements

A PhD may be a task completed by a single person, however it is not undertaken in solitude. There are a number of people that I have had the pleasure to meet and work with throughout my PhD, making this a immensely enjoyable and rewarding experience. While there are many people that have been part of my studies, I would like to take the opportunity here to give some special thanks.

Firstly, I would like to thank my supervisor, Associate Professor Gary Froyland, who has been very supportive throughout my studies. Gary allowed me the freedom to follow my own research ideas and has always been available when I required any advice. I am very appreciative of the time that he has given me. I would also like to thank my co-supervisor, Dr Cheng-Lung Wu, who provided invaluable answers to my aviation related questions.

The School of Mathematics and Statistics has provided a wonderful environment to complete my PhD. Most importantly, the many postgraduate students that I have interacted with during my studies has made this a well-rounded and extremely fulfilling experience.

I was fortunate to spend 3 months of my PhD at GERAD in Montréal working with Guy Desaulniers and François Soumis. I learnt a lot from my time there and it has really added to my PhD experience. I would like to thank Guy and François for the time that they gave me during my visit.

Finally, I would especially like to thank my wife, Penelope Maher, for the significant amount of support that I have received. Having both started our PhD's on the same day, it has been wonderful to be able to share in this experience. Penelope has kept me very well grounded, which has aided a very smooth completion of this thesis. I have nothing but thanks for Penelope and I would like wish her all the best as she embarks on the final stages of her own PhD.

Contents

List of Figures

List of Tables

List of Algorithms

Acronyms

ARP	Aircraft Routing Problem
BMP	Benders Master Problem
CDRP	Crew Duty Recovery Problem
Colgen	Column Generation
CRG	Column-and-Row Generation
CV-PSP	Cancellation Variables Pricing Subproblem
CV-PSPR	Cancellation Variables Reduced Pricing Subproblem
CV-PSPRS	Cancellation Variables Scaled Reduced Pricing Subproblem
DMWAP-s	Dual Magnanti-Wong Auxiliary Problem for Scenario s
GCF	Grönkvist Connection Cost Function
IRP	Integrated Airline Recovery Problem
IRP-PR	Integrated Airline Recovery Problem with Passenger Reallocation
LOF	Line-of-Flying
LP	Linear Programming
MLOF	Maintenance Line-of-Flying
O-D	Origin-Destination
PBSP-s	Primal Benders Subproblem for Scenario s

PMWAP-s	Primal Magnanti-Wong Auxiliary Problem for Scenario s
PRTAP	Pure Robust Tail Assignment Problem
PSP^k	Pricing Subproblem for Crew k
PSP^r	Pricing Subproblem for Aircraft r
PSP^R	Pricing Subproblem for all Aircraft
PTAP	Planning Tail Assignment Problem
RCSPP	Resource Constrained Shortest Path Problem
RMP	Restricted Master Problem
RrCG	Row-reduced Column Generation
RRTAP	Recoverable Robust Tail Assignment Problem
RTAP	Recovery Tail Assignment Problem
SDAMRP	Single Day Aircraft Maintenance Routing Problem
SDAMRP-RR	Recoverable Robust Single Day Aircraft Maintenance Routing Problem
SDW	Structured Dantzig-Wolfe Decomposition
SRMP	Short Restricted Master Problem
ZCF	Zero Connection Cost Function

Supporting Publications

[1] G. Froyland, S. J. Maher, and C.-L. Wu. The recoverable robust tail assignment problem. *Transportation Science*. To appear.

[2] S. J. Maher. A novel passenger recovery approach for the integrated airline recovery problem. In preparation.

[3] S. J. Maher. Solving the integrated recovery problem using column-and-row generation. In preparation.

[4] S. J. Maher, G. Desaulniers, and F. Soumis. Recoverable robust single day aircraft maintenance routing problem. In preparation.

Chapter 1

Introduction

Operations research is an interdisciplinary field that employs a variety of analytical tools to improve decision making. Great success in the application of operations research is witnessed in industries where better planning decisions have the ability to significantly reduce costs. One such industry that is heralded as an operations research success story is the aviation industry. The two largest costs for an airline are fuel and crew, driving great interest in finding the best allocation of these scarce resources. The use of operations research has greatly transformed the airline industry, becoming integral to the planning and operations processes.

The application of operations research to minimise the cost of crew and aircraft has improved the profitability of airlines and lead an expansion of the industry. Lower costs permitted airlines to provide services to a greater number of airports that were previously viewed as uneconomical. This expansion was coupled with an increase in competition as more airlines operated common routes. The positive effect of the improved scheduling decisions was the increase in revenue as air travel became available to a wider population.

The push from airlines to remain price competitive and offer a greater number of services has impacted negatively on efficient operations. The expansion of the airline industry by offering a greater number of flights to each destination has not been mirrored by investment in shared resources such as airport infrastructure. This has placed pressure on these resources, especially during periods of poor weather, seriously affecting the ability of airlines to operate their flight schedule as planned. Additionally, the decrease in ticket prices caused by an increase in competition prompted further efforts to reduce operating costs. The reduction in costs involved the planning of crew and aircraft using optimisation techniques. Unfortunately this process was performed under the expectation that the schedule will be operated as planned. Schedules became *brittle*, greatly reducing the capability of airlines to remain on time.

Negative effects from applying operations research to the airline industry have arisen due to a lack of understanding of how planning decisions affect the magnitude of schedule disruptions. Disruption management attempts to address the disparity between the planned and operational costs through better planning decisions and actions on the day of operations. Robust planning is a *proactive approach* to disruption management that typically focuses on the airline planning process. This approach introduces redundancies into the planned solutions to help avoid schedule disruptions. While robust planning is effective in improving the operational performance of the airline, this is generally at the expense of an overly conservative planning solution. Airline recovery describes the actions taken by an airline to return operations back to what was originally planned, following a disruptive event. The action of recovery is a *reactive approach* to disruption management that is generally not considered during the planning process. Both approaches have been demonstrated to reduce the impact of disruptions on an airline by reducing operational costs.

While disruption management is divided into proactive and reactive approaches, there are very few examples from the literature that link the two. Robust planning attempts to identify features of the planning process that are *expected* to provide improved operational performance. Unfortunately, there is no guarantee that the robustness features will provide the desired improvement without explicit evaluation in an operational environment. Conversely, the ability of an airline to employ recovery actions is impacted by the decisions made during the planning process.

There is a strong relationship between airline recovery actions and robust planning approaches. However, much of the planning and recovery methods have been developed in isolation. The concept of recoverable robustness, introduced by Liebchen *et al.* [57], is the focus of this thesis, which attempts to bridge the gap between proactive and reactive approaches to disruption management. This thesis investigates various optimisation models that explicitly consider the possible recovery actions for airlines during the planning process.

1.1 Aim of Thesis

Recoverability is a measure of the difficulty faced by an airline to return operations back to plan following a disruptive event. A desired outcome from improving recoverability is the reduction in operational costs. The aim of this thesis is to investigate the application of recoverable robustness to improve the recoverability of airline planning solutions.

The airline planning process involves solving a number of interrelated optimisation problems. These problems include, but are not limited to, schedule design, aircraft routing and crew planning. The aircraft routing problem is solved early in the planning process and consequently the solutions to this problem affects the solutions to problems solved in the subsequent stages. As such, the planning decisions made in the aircraft routing problem can have a significant impact on the overall recoverability of the airline. The primary focus of this thesis is the tail assignment problem, which is a form of the aircraft routing problem. To provide an extensive review of applying recoverable robustness to airline planning problems, extensions to the tail assignment problem, such as including maintenance constraints and integrating with crew planning, are also investigated.

1.2 Overview of Study

Recoverable robustness is an approach that permits the explicit consideration of recovery actions in the solution process of planning stage problems. This approach improves the recoverability of airline planning solutions by simultaneously solving planning and recovery problems, where the latter evaluates the recoverability of the former. Any potential recoverability improvements identified as a result of this evaluation are provided as feedback to the planning stage. The modelling of the planning and recovery problems is of critical importance in this thesis, which is discussed in Chapters 2 and 3.

The aircraft routing problem and its derivative, the tail assignment problem, are the main foci of this thesis. Examples of the modelling approaches used to formulate each problem are presented in Chapter 2. A detailed overview of the current approaches developed for the aircraft maintenance planning problem and the various robustness techniques are also presented. This discussion demonstrates the benefits of considering robustness in the airline planning process. In addition, Chapter 2 highlights the limitations of the current robustness approaches to improve the recoverability of airline planning problems.

The evaluation stage of the recoverable robustness framework solves a recovery problem to assess the recoverability of the planning stage solution. Recovery is a critical part of the recoverable robustness framework, as such an understanding of the current solution approaches is essential. A review of the complete airline recovery process is provided in Chapter 3. This review includes examples of the aircraft and crew recovery problems, detailing the various modelling techniques that are relevant throughout this thesis. The efficiency of the recovery problem solved in the evaluation stage of the recoverable robustness framework significantly affects the runtime of the solution approach. The current approaches developed to efficiently solve airline recovery problems are presented in Chapter 3.

The individual stages of airline planning and recovery are within the class of large-scale optimisation problems. Since the recoverable robustness framework simultaneously solves planning and recovery problems, decomposition methods are required to improve the problem tractability. The solution techniques that are employed throughout this thesis are column generation, Benders' decomposition and branch-and-price. Each of these solution techniques are discussed in Chapter 4.

The concept of recoverable robustness is applied to the tail assignment problem in Chapter 5. This represents the first application of recoverable robustness to airline planning problems. The recoverable robust tail assignment problem is a feedback robust approach that extends the robust and recoverable planning approaches presented in Chapter 2. The planning and recovery tail assignment problems presented in Chapters 2 and 3 are used in the planning and evaluation stages of the recoverable robustness framework respectively. As a contribution to the recoverable robustness framework, a full set of recovery options are employed in the evaluation stage to accurately simulate the actions of an airline. The solution methods presented in Chapter 4 are applied to solve the recoverable robust tail assignment problem, with a number of problem-specific enhancement techniques detailed in Chapter 5. The results presented in Chapter 5 demonstrate the potential gains in recoverability for the tail assignment problem achieved by applying recoverable robustness.

The application of recoverable robustness to airline planning problems is extended in Chapter 6 by solving an aircraft maintenance routing problem in the planning stage. Maintenance planning is a critical part of the airline business process to ensure the safe operations of the entire fleet. A contribution of this chapter is a novel modelling technique for the aircraft maintenance planning problem that extends upon the maintenance planning approaches presented in Chapter 2. As an extension to Chapter 5, the planning stage of the recoverable robust problem is more complex, and the larger data sets used in the computational experiments increase the complexity of the evaluation stage. As such, the solution methods presented in Chapter 5 are developed further in Chapter 6, investigating additional enhancements techniques. The results will demonstrate that the recoverable robustness framework is applicable and effective for a variety of airline planning problems.

The solution approach of column-and-row generation is reviewed in Chapter 7. This solution approach is applied to solve the integrated airline recovery problems presented in Chapters 8 and 9. A contribution of Chapter 7 is the development of a general framework to apply column-and-row generation to problems with multiple sets of secondary variables. This framework is presented as a direct alternative to Benders decomposition.

The integration of stages from the sequential planning process is demonstrated in Chapter 2 to reduce operational costs. The integration of the aircraft and crew planning problem is considered as a potential extension of the recoverable robustness framework investigated in Chapters 5 and 6. Chapter 8 introduces the integrated aircraft and crew recovery problem to investigate its implementation in the evaluation stage of the recoverable robustness framework. The solution approach of column-and-row generation is employed to improve the solution runtimes of the integrated airline recovery problem. A contribution of Chapter 8 is the explicit evaluation of the solution runtimes and quality achieved using column-and-row generation compared to a standard column generation approach. A number of enhancement techniques for column-and-row generation are identified in Chapter 8, contributing to the solution approach. In addition, the application of column-and-row generation to solve the integrated airline recovery problem improves upon the solution approaches presented in Chapter 3.

The quality of the feedback from the evaluation stage of the recoverable robustness framework greatly affects the efficacy of the approach. Chapter 9 investigates the consideration of passengers in the recovery problem to improve operational costs and the evaluation feedback. Passenger recovery is modelled through the cancellation variables, detailing the alternative travel arrangements for passengers on cancelled flights. This passenger recovery approach is evaluated using the integrated airline recovery problem developed in Chapter 8 and solved using column-and-row generation. The major contribution of Chapter 9 is the novel modelling approach for passenger recovery. This modelling approach formulates the recovery problem with two sets of secondary variables, demonstrating the strength of the column-and-row generation framework developed in Chapter 7. Chapter 9 also demonstrates the benefits of considering passengers in the recovery process by reducing operating costs and increasing passenger flow through the network.

Chapter 10 will discuss the conclusions from each of the preceding chapters and detail the key contributions. The investigation of the recoverable robustness framework will highlight a number of limitations, which are discussed in this chapter. The conclusions will demonstrate the potential recoverability improvements achieved by applying recoverable robustness to airline planning problems.

Chapter 2

Airline Planning

The complete airline planning process is a large, intractable problem, which is frequently broken into a number of smaller sequential stages. These stages typically consist of, but are not limited to, schedule design, fleet assignment, aircraft routing and crew planning. To address the complexity of the complete planning problem a sequential solution approach is employed. The sequential approach involves solving each stage in the order presented above, using the solutions to preceding stages as input. While this approach significantly reduces the complexity of the complete planning problem, a disadvantage of this process is that there is no feedback between the stages. As such, it is common for the sequential approach to result in a suboptimal global solution and fixing the solution from preceding stages may cause infeasibility [27, 92]. There have been numerous approaches proposed to alleviate these two undesired effects, which include using a feedback process and the integration of problem stages.

An airline is a very resource-intensive business requiring the efficient management of these interrelated resources to achieve low operational costs. The two largest costs of an airline are fuel and crew remuneration which has motivated much research into the related problems of aircraft routing and crew pairing. While it is important to efficiently plan the crew and aircraft resources at a low cost, anecdotal evidence has indicated that the resulting solution may be highly susceptible to schedule perturbations. A common effect of any disruption is an increase in operating costs, with the design of the planning solution significantly impacting the magnitude of this increase. Various robust planning approaches for the aircraft routing and crew pairing problems developed to address this undesired planning outcome will be discussed in this chapter. Robust planning attempts to reduce the operating costs of an airline by introducing redundancies and strategies to reduce the effect of schedule perturbations.

The aircraft routing problem is solved early in the sequential process using the solution to fleet assignment as input. The objective of the aircraft routing problem is to allocate aircraft to flights while adhering to structural and operational constraints, such as maintenance requirements. This objective is achieved by constructing a sequence of flights for each aircraft, called aircraft routes, to be performed within a specific time horizon. A solution to the fleet assignment problem assigns a set of flights to each fleet type, as such the aircraft routing problem is separable by type. This allows the set of aircraft considered in the problem to be treated as a homogeneous commodity; therefore this problem can be solved as a feasibility problem. An alternative to the aircraft routing problem is the tail assignment problem. This alternative problem can be solved for a heterogeneous fleet with each aircraft individually identified by its tail number. In the tail assignment problem, the individual characteristics of each aircraft are provided, facilitating fleet decisions.

The crew pairing problem is one of the most critical stages in the airline planning process due to its complexity and high planning costs. A major source difficulty for crew planning problems is the non-linear remuneration structure and complex set of work rules affecting the construction of crew duties and pairings. The work rules for crew dictate the satisfactory working conditions for the staff, which include an upper bound on the total working hours and the number of hours spent flying in a single day. Similar to an aircraft route, a crew pairing is a sequence of flights that is performed by a crew within a time horizon. A crew pairing generally spans multiple days, which can include a number of overnight stays away from their home base. The sequence of flights that are performed by a crew group during a single day is called a duty, and a crew pairing is constructed as a sequence of duties. Due to the complexity of this problem, significant benefits have been realised through the use of operations research techniques.

The aircraft routing problem and its various maintenance and robust planning derivatives are the focus of this thesis. Section 2.1 will provide a basic framework for the aircraft routing and tail assignment problems to help direct the discussion of the relevant literature. This basic framework is followed by a discussion of literature related to the maintenance planning problem, robust and recoverable airline planning problems and recoverable robustness in Sections 2.2.1, 2.2.2 and 2.2.3 respectively. While numerous robust planning approaches have been developed for the crew pairing problem, the discussion of these approaches in this chapter will be limited to cases that also involve the aircraft routing problem.

2.1 Mathematical Formulation

The aircraft routing problem is a fundamental stage of the sequential airline planning process. The mathematical model for the aircraft routing problem is presented in Section 2.1.1 and its derivative, the tail assignment problem, will be presented in Section 2.1.2. The following sections present the basic formulations of these problem which have been the focus of various alternative planning and robustness approaches. Examples of the extensions that have been previously considered are presented in Section 2.2.

2.1.1 Aircraft Routing Problem

There are many variations of the aircraft routing problem (ARP) that have been developed to address different planning objectives. As stated previously, the primary objective of the ARP is to ensure that each flight in the schedule is assigned an aircraft. Secondary objectives, such as maintenance planning, can be satisfied in the construction of flight routes. The solution to this problem describes a set of flight routes for each aircraft in the fleet, which can span time periods ranging from a single day to many weeks. The notation given in Table 2.1 will be used to describe the formulation of the ARP.

The ARP presented in this section forms a single stage within the sequential planning process. The first stage in this process involves the schedule design, which determines the city pairs to serve and the frequency and timing of flights between these cities. The solution to the schedule design stage constructs a set of flights $j \in N$ that are operated by the airline. An aircraft flight route specifies the flights operated by an aircraft between the origination and

R	is the set of all aircraft r
P	is the set of all flight strings p
N	is the set of all flights j
B	is the set of airports b where aircraft flight strings can originate and terminate
C	is the set of all feasible connections in the network, $C = \{(i, j)\|i, j \in N \cup B\}$
y_p	$= 1$ if flight string p is used, 0 otherwise
c_p	$=$ the cost of using flight string p
a_{jp}	$= 1$ if flight j is in string p, 0 otherwise

Table 2.1: Notation for the aircraft routing problem.

termination at overnight airports B. The construction of aircraft flight routes requires the set of all pairs of flights that can be performed in succession, called connected flights, to be defined. A pair of connected flights $(i, j), i, j \in N \cup B$ is identified by i) the destination of i being the same as the origin of j; and ii) the difference between the departure of j and the arrival of i is less than a minimum connection time, called the minimum turn time. All pairs of flights that satisfy these two conditions are called feasible connections and are contained in the set $C = \{(i, j) | i, j \in N \cup B\}$. A connection network is defined with all flights in N representing the nodes (the source and sink nodes are given by B), and the arcs are given by the connections contained in C.

An aircraft flight route is defined as a subset of flights in N that form a connected path through the connection network described above. The flight string formulation introduced by Barnhart *et al.* [12] is used to develop the ARP presented in this chapter. This problem is solved under the assumption that all aircraft are the same type, so the set of feasible flight strings p for all aircraft are contained in P. The decision variables y_p equal 1 if flight string p is operated by an aircraft, 0 otherwise. The parameters a_{jp} equal 1 to indicate that flight j is contained in flight string p, which are the constraint coefficients of the decision variables y_p.

The ARP is defined as,

$$\text{(ARP)} \quad \begin{aligned} \min \quad & \sum_{p \in P} c_p y_p, && (2.1) \\ \text{s.t.} \quad & \sum_{p \in P} a_{jp} y_p = 1 \quad \forall j \in N, && (2.2) \\ & \sum_{p \in P} y_p \leq |R|, && (2.3) \\ & y_p \in \{0, 1\} \quad \forall p \in P. && (2.4) \end{aligned}$$

The objective function of the ARP minimises the total cost of assigning aircraft to flight routes. Flight coverage is enforced through constraints (2.2) and the number of flight routes is restricted to at most $|R|$, which is the number of aircraft, by constraints (2.3). Since the cost of flying each flight is identical for all aircraft of the same fleet, it is possible to set $c_p = 0, \forall p \in P$ and solve the ARP as a feasibility problem. Alternatively, the cost of a flight string can be used to introduce robustness into the planning problem. Such a method involves defining a connection cost function which favours or penalises specific connection times. This type of robustness is called *proxy robustness* and a good example of this approach is presented in Grönkvist [45].

2.1.2 Planning Tail Assignment Problem

An extension of the ARP is the tail assignment problem, whereby each aircraft is identified individually to model aircraft specific characteristics [45]. Greater detail of each aircraft is provided in the tail assignment problem by observing aircraft specific constraints, such as maintenance and restricted flight constraints. Additionally, the tail assignment problem allows the integration of the fleet assignment and aircraft routing problems through the explicit definition of aircraft capacities. The planning tail assignment problem (PTAP) is presented using the additional notation given in Table 2.2.

P^r	is the set of all strings p for aircraft r
y_p^r	$= 1$ if aircraft r uses string p, 0 otherwise
c_p^r	$=$ the cost of aircraft r using string p

Table 2.2: Additional notation for the planning tail assignment problem.

Since each aircraft is individually identified by their tail number, the set of flight routes P is partitioned to define an individual set P^r for each aircraft r. The decision variables y_p^r equal 1 if flight string p is operated by aircraft r, at a cost of c_p^r in the objective function. The cost structure of this problem is identical to that of the ARP. However, it is possible to introduce aircraft specific costs given the individual referencing of decision variables.

The PTAP is given by,

$$\text{(PTAP)} \quad \begin{aligned} \min \quad & \sum_{r \in R} \sum_{p \in P^r} c_p^r y_p^r, && (2.5)\\ \text{s.t.} \quad & \sum_{r \in R} \sum_{p \in P^r} a_{jp} y_p^r = 1 \quad \forall j \in N, && (2.6)\\ & \sum_{p \in P^r} y_p^r \leq 1 \quad \forall r \in R, && (2.7)\\ & y_p^r \in \{0,1\} \quad \forall r \in R, \forall p \in P^r. && (2.8) \end{aligned}$$

The primary objective of the PTAP is identical to that of the ARP; as such there is little difference between the solutions of the two problems. The main difference is observed in the set of constraints (2.7) that ensure at most one flight route is assigned to each aircraft. This is required in the PTAP given that each aircraft is individually referenced, whereas in the ARP an upper bound on the flight route count is sufficient.

The ARP and the PTAP demonstrate the basic form of the aircraft routing and tail assignment problems that are the focus of much research. In many cases, the basic form of the problems remains unchanged with much of the variation seen in the construction of aircraft routes. While the ARP and PTAP have been presented using the flight string notation, an alternative formulation for these problems is as a multi-commodity flow problem.

2.2 Related Literature

The literature presented in this section details the current approaches used to solve the maintenance planning problem and apply robustness to the airline planning process. Section 2.2.1 describes the current maintenance planning approaches, relative to particular business practices. This is followed by a description of robust and recoverable airline planning approaches in Section 2.2.2. Finally, the recoverable robustness framework is described in Section 2.2.3.

2.2.1 Aircraft maintenance planning

The aviation industry is governed by regulatory bodies who define the requirements of airlines to ensure safe operations. One such requirement of airlines is the regular maintenance of aircraft. There are a number of different maintenance checks that must be performed, each having a different *scope*, *duration* and *frequency* [23]. The simplest and most frequent maintenance check is called a type A check which is required once every 65 hours of flight time [23]. Since there are significant penalties for exceeding maintenance limits set by aviation governing bodies, airlines aim to perform type A checks once every 35 to 40 hours. A maintenance check can last for a number of hours, during which the aircraft is inactive. Therefore, airlines aim to perform any maintenance overnight. Given the strict regulatory requirement and the cost of maintenance, efficient maintenance planning is of great interest to an airline.

The aircraft routing process involves the solution to the aircraft routing and maintenance planning problems which are either solved separately or as part of an integrated problem. The business practices of airlines has a direct effect on the formulation of the maintenance planning problem, which are reviewed by Lacasse-Guay *et al.* [52]. In [52], the authors explain that the various formulations fall into three broad categories, *big cycle*, *strings* and *one-day routes*. There are many similarities between these approaches, with each following the primary objective to plan regular maintenance visits for aircraft. The difference between the formulations arise from the methods employed to construct the aircraft flight routes.

The maintenance planning problem can be solved within the aircraft routing problem or using the aircraft routing solution as input. The former approach is more common for the *strings* and *one-day routes* approaches where the latter is generally used in identifying a *big cycle*. While the two different methods effectively solve the maintenance planning problem, suboptimality can result from using a fixed aircraft routing solution. The integration of maintenance planning and aircraft routing provides the necessary flexibility to identify the optimal solution to both problems. Such an integrated aircraft routing and maintenance planning problem is developed in Chapter 6.

Big Cycle formulation

The *big cycle* approach involves identifying a single flight route covering multiple days that includes every flight in the schedule. Solving the aircraft routing problem with this objective alone finds a solution that enforces the equal utilisation of all aircraft. To ensure that all maintenance requirements are satisfied, regular visits to maintenance stations are scheduled in the construction of the cycle. The single flight route is operated by every aircraft in the fleet, with each overnight stop in the cycle representing an aircraft starting point. Since the big cycle is constructed to be maintenance feasible, the operation of all aircraft on this cycle satisfies the maintenance requirements for the entire fleet.

Early work on the aircraft maintenance routing problem is presented by Feo and Bard [37]. The problem considered by [37] combines the maintenance routing problem with an objective to minimise the number of maintenance station locations. The authors assume that the maintenance planning is solved following the solution to the aircraft routing problem. As such, the maintenance planning problem receives a set of generic flight routes that must be assigned to aircraft to satisfy a 4-day maintenance requirement. The solution to this problem identifies a 7-day cycle satisfying a 4-day maintenance requirement while identifying the required locations for maintenance stations. Using the solution to the aircraft routing problem as input reduces the flexibility of the approach, potentially resulting in suboptimal solutions.

A multi-commodity flow formulation is employed for the aircraft routing problem developed by Clarke *et al.* [23], which is modelled to directly consider maintenance requirements in the construction of aircraft routes. In [23], the maintenance planning and aircraft routing problems are integrated, allowing flexibility in the construction of the big cycle. This flexibility in the route construction captures more passenger through revenues from the aircraft routing solution while achieving an optimal maintenance planning.

Gopalan and Talluri [44] introduce the concept of a line-of-flying (LOF) as the sequence of flights performed by an aircraft during a single day. The LOFs are provided as an input from the aircraft routing problem, however some modification is required to identify a solution with a sufficient number of maintenance opportunities. A big cycle is constructed to contain all LOFs with a visit to a maintenance station scheduled once every three days. Similar to [37], the LOFs are set from the solution to the aircraft routing problem. While some modification is permitted on the LOFs, this reduced flexibility in the maintenance planning problem is a limitation to the approach. The integration of aircraft routing and maintenance planning in Chapter 6 addresses this limited flexibility and ensures the optimality of both problem.

The *big cycle* approach is effective in providing a maintenance planning solution that also satisfies the equal utilisation of all aircraft. This secondary outcome from the maintenance planning unfortunately compounds the effect of schedule perturbations, whereby a single disruption affects the complete cycle for all aircraft. The aircraft maintenance routing problem developed in Chapter 6 is solved for a single day, therefore avoiding the effects of disruptions from previous days.

String formulation

The *string* model for maintenance planning introduces the concept of flight strings for aircraft flight routes that are maintenance feasible. This modelling approach identifies flight strings that begin and end at maintenance stations, hence providing an adequate number of maintenance station visits to satisfy regulatory requirements. The ARP presented in Section 2.1.1, is developed using a flight string model formulation, as such all flight routes contained in P are maintenance feasible. The solution to the ARP will provide an optimal aircraft routing and maintenance planning.

The airline planning process is solved as a series of sequential stages to improve problem tractability. However, it is expected that the integration of two or more stages can achieve a higher solution quality. Barnhart *et al.* [12] presents an example of the integrated fleet assignment and aircraft routing problem to capture more through revenues with better fleet assignment decisions. The flight string model permits the use of column generation, demonstrating its effectiveness for solving the aircraft routing problem.

The modelling approach presented by Feo and Bard [37] is applied by Sriram and Haghani [80] in a string model for the maintenance scheduling problem. Sriram and Haghani [80] illustrate the difference between the *big cycle* and *string* formulations where the former requires each aircraft to perform the same cycle and the latter identifies individual cycles for each aircraft. Similar to [37], the LOFs from an aircraft routing solution are used as input for this problem, with no modification of the routes permitted. Sriram and Haghani [80] solve the maintenance routing problem for a heterogeneous fleet allowing flexibility through the reassignment of LOFs to different fleet types.

By design, the *string* approach is less susceptible to disruptions than the *big cycle* approach since each aircraft is assigned an individual flight string. However, the maintenance feasible flight strings span multiple days, as such it is common for maintenance plans to be affected by schedule perturbations from previous days. Any disruption to the maintenance plan requires costly intervention by the airline, rerouting aircraft to satisfy maintenance requirements. The *one-day routes* approach employed in Chapter 6 helps to avoid schedule perturbations by constructing flight strings that only span a single day.

One-day Routes formulation

The *big cycle* and *string* models presented previously involve the construction of maintenance plans with little consideration given to schedule perturbations. Hence, any schedule disruption can cause the current maintenance plan to become infeasible. The *one-day routes* approach schedules maintenance checks for a single day to minimise the impact of disruptions from previous days. As such, this approach is a form of robust planning.

A key feature of the *one-day routes* approach is the construction of flight routes that span a single day, originating and terminating at permissible overnight airports. The objective of this approach is to identify a sufficient number of flight routes terminating at maintenance stations, satisfying all maintenance requirements. Regardless of any disruption that may occur on previous days, the solution to this problem ensures that on average all maintenance critical aircraft originating from each airport will receive maintenance the following night.

Heinhold [47] describes an example of a one-day routes approach implemented by Southwest Airlines. In [47], the minimum number of required maintenance routes is calculated as an expectation of the number of aircraft at each overnight airport requiring maintenance the following day. An optimisation problem is formulated to identify swapping opportunities that minimise any penalties for not assigning maintenance routes to maintenance critical aircraft. This simple approach effectively ensures that the majority of maintenance critical aircraft are provided maintenance routes and that only small changes are required to achieve maintenance coverage for the whole fleet.

The concept of one-day routes is investigated further by Lapp and Cohn [54] to achieve adequate maintenance planning through the modification of LOFs. The approach of Lapp and Cohn [54] is similar to that presented by Heinhold [47]. However, a multi-stage optimisation problem is used to improve the tractability of the problem. The input for this problem is the set of originally planned LOFs departing from each overnight airport; the subset of these terminating at maintenance stations are termed MLOFs. Since the original LOFs do not provide complete maintenance coverage for all aircraft, the first stage of this model identifies the overnight airports that require additional MLOFs. A LOF is converted to a MLOF by performing a single swap with a MLOF departing from a different airport that has an over supply. This process is performed to increase the maintenance reachability from all airports in the network compared to the original aircraft routing solution.

The *one-day routes* maintenance planning problems by Heinhold [47] and Lapp and Cohn [54] demonstrates the potential of this modelling approach. A limitation of both [47] and [54] is the requirement of the aircraft routing solution as an input to the maintenance planning problem. This is addressed in Chapter 6 with the integration of the aircraft routing and maintenance planning problems.

2.2.2 Robust and recoverable airline planning

In the airline planning process there has been great interest in the development of proactive approaches to avoid disruptions. These approaches are broadly termed *robust airline planning*, which focus on developing solutions that are less susceptible to disruptive events. This has stemmed from an awareness of the significant increase in operating costs that can result from ill-considered planning approaches. Broadly, the solution to robust airline planning problems can be executed to plan, even in the event of a disruption. As a variation on robust planning, the concept of *recoverable airline planning* in introduced. Recoverable airline planning integrates the proactive and reactive, also called airline recovery, approaches of disruption management in the planning process. A key feature of recoverable planning is the expectation that the recovery process will be required during daily operations. Therefore, the design of the planned solution should improve the efficacy of recovery techniques to reduce the additional operational costs. Recoverable robustness is an approach that attempts to improve the recoverability of planning stage solutions. This approach is investigated throughout this thesis with applications to various airline planning problems.

The development of airline planning solutions to improve operational performance, as measured by a reduction in costs, is achieved through two main approaches; proxy and feedback robustness. While both approaches attempt to improve operational performance, vastly different solution methods are employed for each. A detailed discussion of robust solution approaches will be presented in the following section including the relevant solution methods.

The *proxy robust* approach to airline planning identifies and exploits particular planning characteristics that are a proxy for robustness. Such proxies include increasing aircraft turn-times or promoting crew and aircraft to use the same flight connections to avoid the spread of delay through the network. The strength of proxy robust approaches depends on how efficacious the identified aspect is at capturing the desired robustness goals, which can vary across data sets. Furthermore, additions to the model (eg. 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.

Conversely, the *feedback robust* class introduces this feedback via second-stage (recovery) decisions. Feedback robust approaches are a superior method for reducing the weighted recovery costs since an explicit evaluation of the planned solution is performed during the optimisation process. This evaluation generally involves a simulation of the recovery process, the outcome of which is used to improve the robustness and recoverability of the planned solution. Feedback robustness has a strong dependency on the scenarios used in the solution process by having a significant effect on the expected operational costs. In addition, the iterative scheme increases solution runtimes, which can greatly exceed that of an equivalent proxy robust model. This demonstrates a trade-off between the solution quality and runtimes for each of the robustness approaches, both critical aspects for large scale optimisation problems. The recoverable robustness framework is an example of a feedback robust approach whereby an evaluation stage solves a recovery problem to assess the recoverability of the planning stage solution.

Proxy robustness

The integration of stages in the sequential planning process is expected to improve the solution quality by providing greater flexibility in the problem stages. Also, the integration of multiple stages allows the use of additional proxy robust approaches to further improve robustness and operational performance. An example of this is presented by Cordeau *et al.* [27], integrating the aircraft routing and crew pairing problems. The integration of these two planning stages permits the use of short connections by crew, which are defined by having a connection time less than the minimum sit time for crew, only when the two flights are performed by the one aircraft. Ensuring that the short connections are used by the same aircraft and crew improves the robustness by reducing the possibility of delays spreading to multiple flight strings. The complexity of this problem is addressed by employing Benders' decomposition, improving tractability by separating the two planning stages into a master and subproblem formulation.

The integration of the crew pairing and aircraft routing problems in a Benders' decomposition framework is extended by Mercier *et al.* [63], introducing the concept of restricted connections. Restricted connections are defined as connections just long enough not to be considered short, and are penalised if the two flights of the connection are serviced by the same crew but not the same aircraft. This penalty is an attempt to further improve the robustness of the integrated problem. The solution process developed by Cordeau *et al.* [27] is improved by Mercier *et al.* [63] by implementing acceleration techniques, such as the Magnanti-Wong method [60] to identify Pareto-optimal Benders' cuts. The Magnanti-Wong and Three-phase methods are reviewed in the implementation of recoverable robustness in Chapters 5 and 6.

Two further extension to the integrated planning problems of Cordeau *et al.* [27] and Mercier *et al.* [63] are presented by Mercier and Soumis [64] and Papadakos [69]. Mercier and Soumis [64] extend upon Cordeau *et al.* [27] to include flight retiming while implementing solution methods presented by Mercier *et al.* [63]. Papadakos [69] presents a model which integrates the fleet assignment, maintenance routing and crew pairing. The results from these extensions further demonstrate the potential improvements in planning costs and robustness achieved through the integration of problem stages.

The primary motivation for integrating multiple planning stages in the approaches presented above is to reduce operational costs. However, explicitly modelling the use of short and restricted connections achieves an improvement in robustness as a positive externality. Weide *et al.* [93] and Dunbar *et al.* [33] present alternative approaches to the integrated planning problem, focusing on the improvement in specific robustness measures. The major focus of [93] and [33] is reducing the propagation of delay through the flight network. Propagated delay results from crew changing aircraft following a disruption and spreading delays to previously undisrupted flights. Explicitly modelling features designed to reduce the prevalence of delay propagation greatly improves the operational performance.

The iterative solution approach for the integrated problem uses a feedback process to improve the solution quality. The solution approach implemented by [93] and [33] iterates between the two integrated problems, solving the crew pairing (aircraft routing) problem using a fixed aircraft routing (crew pairing) solution from the previous iteration. To achieve greater operational performance Weide *et al.* [93] introduce a non-robustness measure to the crew pairing problem that increases the slack time for crew using restricted connections to mitigate delay propagation. This is further enforced in the aircraft routing problem, which has an objective to maximise the number of restricted connections performed by aircraft that are also performed by crew. The robust planning model of Dunbar *et al.* [33] provides much greater detail of delay propagation through flight network, resulting in a more robust planning solution. Dunbar *et al.* [33] explicitly measures the probability of propagated delay across flight strings for crew and aircraft with an objective to minimise this. Both Weide *et al.* [93] and Dunbar *et al.* [33] determine that the length of the connections is a contributing factor to the operational performance of an airline planning solution.

Significant improvements in operational costs are observed from the integration of multiple planning stages [27, 33, 63, 64, 69, 93]. This potential improvement is the motivation for considering the integrated recoverable robust aircraft routing and crew duty problem in Chapter 8. The application of recoverable robustness is expected to directly improve the recoverability of the integrated planning problem through an explicit evaluation of the planned solution. This is an improvement upon the integrated planning problems presented above, whereby improved operational performance is achieved through proxy robust approaches.

Integration of two or more airline planning stages 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 *et al.* [53] propose the use of flight re-timing to reduce delay propagation. The objective of this model is to find an optimal aircraft routing while reducing the amount of delay experienced by passengers and missed connections. Borndörfer *et al.* [18] present an alternative model 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 [18] the probability of the length of delay is explicitly modelled, and the expected delay is included in the objective function. In practice, a general method of introducing robustness into the tail assignment problem is via *key performance indicators*, examples of such methods are explained in Wu [94]. Using a set of scenarios, developed from airline data, the authors [18] compare the amount of propagated delay resulting from their model with a tail assignment developed from a traditional *key performance indicator* method. Chapters 5 and 6 demonstrate an improvement in operational performance with the application of recoverable robustness to the tail assignment and aircraft routing problem respectively.

Recoverability

The work presented above describes robust planning approaches that attempt to improve operational performance by exploiting specific planning characteristics. There are many characteristics that are identified as favourable, such as the better use of short connections in Cordeau *et al.* [27] and minimising the expected propagated delay in Dunbar *et al.* [33]. While these approaches help to avoid and minimise the effect of disruptions, there is no consideration of the actions employed by an airline during recovery.

An alternative approach to robust planning involves the enhancement of characteristics that improve the *recoverability* of the planned solution. Recoverability is a measure of how difficult or costly it is to return the operational schedule to plan following a disruption. This can be evaluated in a variety of ways, most commonly through the expected recovery costs. However, a measure of recoverability can also include the effort required by the airline during recovery, such as the number of changes implemented. Improving the recoverability of the planned solution is achieved through proxy and feedback robust approaches. The recoverable robustness framework applied in Chapters 5 and 6 is an example of such a feedback robust approach.

There are many features of the planning solution that can be enhanced to improve recoverability. An example of a proxy robust approach is given by Ageeva [4] which is solved with the objective of increasing the prevalence of aircraft swapping opportunities. Swapping opportunities are periods of time when two aircraft are planned to be on the ground at the same airport. In the event that one of the aircraft is disrupted, a swap can be made to allow the higher-valued route to continue on time. Since swapping opportunities can be identified in the planning stage, this approach has the potential to improve recoverability by proxy robustness.

The concept of recoverability is investigated by Eggenberg [34] in the development of both robust and recoverable aircraft routing problems. The aircraft routing problem is solved by optimising a number of favourable planning characteristics using uncertainty feature optimisation. Robustness is achieved in the aircraft routing problem by focusing on the amount of time between flights in each connection, using different metrics to achieve different optimisation objectives. Enforcing larger connection times between flights is a very simple and effective method to avoid delay propagation, however this results in a very conservative solution. Recoverable airline schedules are also presented in [34], again with a focus on the connection time between flights. The improved recoverability for the aircraft routing problem in [34] is achieved by incorporating the aircraft swapping technique of Ageeva [4]. To determine the potential recoverability of the planned solution, Eggenberg [34] solves a recovery problem over a series of disruption scenarios.

The fundamental difference between recoverable and robust airline schedules is the explicit focus on recovery decisions during the planning stage. Kang [51] presents a proxy robust approach described as *degradable scheduling*, developed by prescribing a simple recovery policy. Degradable scheduling follows the idea that disruptions will occur in operations and that the highest revenue earning flights should be protected from delays and cancellations. This unique method for robust planning involves decomposing the airline schedule into a number of different layers, partitioning the flights by their expected yield. The different layers provide a priority in which flights are protected in a disruption, with flight delays and cancellations first occurring in the lowest layer. The design of the planning solutions guides the recovery process to achieve the highest revenue following any disruptive event.

Some US airline networks are designed with a hub and spoke structure, with the majority of the activity occurring at the hubs. Rosenberger *et al.* [74] exploits this particular network structure by introducing the concept of hub isolation and short cycles. By limiting the number of aircraft that service each hub in the network, it is possible to isolate a disruption to a particular hub, protecting flights servicing other airports. The introduction of short cycles is a scheduling decision that takes account of the actions performed by the operations controller. This approach attempts to minimise the number of flights in each string between the departure and arrival at the same hub. Short cycles ensure that if a cancellation occurs at a hub, the number of flights in an aircraft flight route subsequently requiring cancellation is minimised. These concepts focus on possible recovery decisions, providing the operations controller with many low cost recovery options.

While the network design for an airline is critical in determining the markets that can be serviced there is also a significant effect on recoverability. Smith and Johnson [79] address the impact network design has on recovery options and introduce the concept of *station purity*. This concept attempts to limit the number of fleets that can service each station. When each aircraft within a fleet is interchangeable, *station purity* can provide more aircraft swapping opportunities at each base, which has been shown to have a positive effect on the expected recovery costs. Gao *et al.* [41] present an extension to this concept by applying *station purity* to the integrated crew planning and fleet assignment problem. In the integrated model, the station purity constraints attempt to limit the number of fleets and crew bases that service each station. This approach attempts to improve the recoverability of the crew planning solution by providing more options for crew to return to base, avoiding costly overnight stays and deadheads.

Proxy robust approaches are limited in their ability to improve recoverability without the explicit evaluation of the planned solution. This is addressed by feedback robust approaches, where an evaluation stage is fundamental to the solution approach. The recoverable robustness framework is such a feedback robust approach, with the potential improvements in recoverability demonstrated in Chapters 5 and 6.

Feedback robustness

Yen and Birge [99] present an example of a feedback robust approach to solve the crew pairing problem. This problem is formulated as a stochastic program which has the inherent characteristic of improving the master problem with feedback from a number of scenario subproblems. In this stochastic programming model each subproblem describes a disruption scenario related to flight delays. The authors evaluate the effect of particular disruption scenarios on the propagation of delay caused by crew pairings and the interaction with the aircraft routing solution. The benefit of feedback robust approaches is the ability to accurately simulate airline recovery in the second-stage of the solution process. This accurate simulation greatly improves the recoverability of the first-stage through the feedback from the second-stage recovery decisions. As an improvement upon the feedback robust approach by Yen and Birge [99], Chapters 5 and 6 solve recovery problems in the evaluation stage that implement a full set of recovery policies.

Robustness and recoverability of airline planning problems can be achieved with both proxy and feedback robust approaches. As presented above, proxy robustness focuses on enhancing specific characteristics of the airline planning problem. Consequently, problems formulated in a proxy robust approach are purely deterministic, with no feedback from the evaluation of the resulting solution. Without feedback, it is very difficult in the planning stage to ensure improvements are made to the expected operational performance. It is necessary to use feedback in the development of recoverable planning solutions as an iterative evaluation of the recovery process. In feedback robust approaches, the planned solution can be evaluated with a full set of recovery policies, which explicitly and implicitly incorporate all proxy robust techniques to improve recoverability. Therefore, feedback robustness provides a superior approach for developing more recoverable airline planning solutions. The results presented in Chapters 5 and 6 demonstrate the significant improvements in recoverability achieved by the application of recoverable robustness.

2.2.3 Recoverable robustness

Outside the airline literature, Liebchen *et al.* [57] have developed a concept called recoverable robustness with an application to railway transportation. Recoverable robustness is a feedback robust approach that evaluates the recoverability of the planning solution during the optimisation process. This technique focuses on finding a planning solution that is recoverable in a limited number of steps, or with *limited effort*. Liebchen *et al.* [57] present the recoverable robust timetabling problem as a demonstration of this technique. In addition to finding a planned solution that is recoverable with limited effort, the recoverable robust timetabling problem also attempts to minimise the number of changes made to the planned timetable in the recovery stages. In this work, 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. As a key feature of recoverable robustness, this technique recognises that in a disruption the planned solution will need to be changed in operations, so the objective is to reduce the expected recovery costs and the number of changes required. Chapter 5 presents the first application of recoverable robustness for airline planning problems.

The work of Liebchen *et al.* [57] presents a generic framework for recoverable robustness, however it does not provide many details of practical application. The recoverable robustness concept is applied to the rolling stock planning problem by Cacchiani *et al.* [20] to demonstrate its use in real world applications. This problem addresses the allocation of rolling stock to trips, which is similar to the aircraft routing problem. A major difference is that the composition of the rolling stock can be modified through coupling and uncoupling decisions at termination locations. Recoverable robustness for this problem aims to protect the rolling stock planning solution from blockages that restrict flow through the network. This is achieved through the evaluation of scenarios with the objective of minimising trip cancellations, the number of additional coupling and uncoupling decisions and the changes to the end-of-day planning. Benders' decomposition is employed to efficiently separate each scenario into individual subproblems. Employing Benders' decomposition to improve the tractability of the recoverable robustness framework is investigated in Chapters 5 and 6.

2.3 Summary

The planning approach of the aircraft routing problem has been the focus of much research on efficient maintenance planning and robustness. While the aircraft routing problem is solved under the expectation of perfect operating conditions, it is evident from the prevalence of disruptions that robustness is a necessary consideration. Robust planning can involve a number of different approaches, such as one-day routes maintenance planning and minimising propagated delay, each with various strengths and weaknesses. An example of robust planning using the *one-day routes* approach is presented in Chapter 6.

It is common for robust approaches to attempt to avoid disruptions, however it is also possible to improve recoverability through better planning decisions. With the use of proxy and feedback robust approaches, the recoverability of the planned solution can improve the performance of the recovery process by providing a set of tools that are expected to reduce the operational costs. While both the proxy and feedback robust approaches are used for robust and recoverable planning, feedback robustness provides a superior approach through the explicit evaluation of the planning solution during the optimisation process. The feedback robust approach of recoverable robustness is applied to the tail assignment and aircraft maintenance routing problems in Chapters 5 and 6 respectively.

The next chapter discusses the reactive approach to airline disruption management, airline recovery. An example of the aircraft and crew recovery problem is presented along with the current solution approaches.

Chapter 3

Airline Recovery

Airline disruption management is described as having two main forms, the proactive and reactive approaches. In Chapter 2, the proactive approach to disruption management is discussed in regards to the aircraft routing problem. Proactive approaches attempt to identify solutions to the aircraft routing problem that are less susceptible to disruptions. This can involve using techniques to avoid disruptions completely, such as longer connection times, or providing strategies to improve the recoverability of the planned solution. The reactive approach to disruption management, commonly referred to as airline recovery, is a vital aspect of the airline business process. This approach is employed to provide efficient, continued operations for an airline in the event of a schedule disruption. Regardless of the proactive approach employed in the planning stage, the prevalence of schedule disruptions indicates the need for the airline recovery process.

Airline recovery involves the redistribution of resources following a schedule disruption to minimise any additional operating costs. The resources considered in the airline recovery process are identical to those in the planning stage, as such it is common to apply a sequential solution approach. The complete airline recovery process involves the stages of schedule, aircraft and crew recovery, each sharing many characteristics with the comparable planning stage. Similar to the planning stage, the sequential solution approach for airline recovery solves each stage in order, using the fixed decisions from preceding stages as input. An additional stage is included in the recovery process, passenger recovery, attempting to maintain a high level of passenger satisfaction. Passenger satisfaction is an important consideration for airlines since this indicates the willingness for passengers to travel with the airline in the future, which is greatly affected by flight delays and cancellations. Since passengers do not contribute a direct cost to the airline, historically this stage is only considered following the solution to all other stages. A greater understanding of passenger satisfaction and more attention to passenger related performance metrics has prompted a recent surge in interest in the passenger recovery problem. A very good review of all stages in the complete airline recovery process can be found in Clausen *et al.* [24].

While the airline planning and recovery process share many similar characteristics, the most significant difference arises from the allowable solution runtimes. The airline planning process is undertaken across a number of months prior to the day of operations; as such each of the stages are provided runtimes ranging from many hours to days. Comparatively, airline recovery is executed following a disruptive event as an immediate intervention by the operations control centre. Therefore, the complete recovery process must be solved with runtimes in the order of minutes to be of practical use to the airline. Much of the research regarding airline recovery stages involves strategies and techniques that are used to reduce the size of the problem and improve the solution runtimes. The following sections will discuss the current strategies developed for the recovery problem, describing the different approaches applied for each stage.

Continuing the comparison between the airline planning and recovery process, this chapter will be presented in a similar form to Chapter 2. Two different mathematical models are presented in Section 3.1, the aircraft and crew recovery problems, to direct the discussion of the relevant literature. The modelling of the aircraft and crew recovery problems are described in this chapter due to their importance in the airline recovery process and to provide an introduction to models that are developed throughout this thesis. This is followed by a discussion in Section 3.2 of the current approaches employed to solve each of the airline recovery stages. As mentioned previously, the airline recovery process involves the solution to the schedule, aircraft, crew and passenger recovery problems. Each of these stages define very discrete fields of research and will be used to direct the discussion throughout this chapter.

3.1 Mathematical Formulation

A disruption is characterised by an incident that prevents flights from arriving or departing as scheduled. Incidents causing disruptions are related to a variety of factors, including poor weather, unplanned maintenance and late arriving passengers. Since a disruption perturbs the schedule used to make airline planning decisions, it is possible that the aircraft routes and crew duties can no longer be operated as expected. Without intervention by the airline operations control centre, schedule perturbations can result in aircraft not receiving maintenance as required or crew exceeding the maximum allowable work hours.

The airline operations control centre employs a variety of strategies to reduce the effect of the disruption and construct aircraft routes and crew duties to adhere to operational constraints. Such strategies, or *recovery policies*, involve flight delays and cancellations, crew deadheading (transporting crew as passengers), aircraft ferrying (flying an aircraft without passengers) and the use of reserve crew. Historically, the recovery process was undertaken manually without the use of automated decision support systems and with personal experience playing a very significant role in the decision making. With the advancement of computing technology, there has been a greater interest in the development of automated recovery solutions approaches. The following sections will present examples of the recovery tail assignment and crew duty problems. These examples introduce a number of features that are implemented in the evaluation stage of the recoverable robustness framework investigated throughout this thesis.

3.1.1 Recovery Tail Assignment Problem

The aircraft recovery problem is generally solved following the schedule recovery, however it is common for formulations to integrate these two problems. Since aircraft are allocated an individual flight route on the day of operations, the aircraft recovery problem presented in this section attempts to maintain that detail by identifying aircraft by their tail number. The primary objective of the recovery tail assignment problem (RTAP) is to identify new flight routes for each aircraft while minimising the length of flight delays and the number of cancellations. It is common for a disruption to cause schedule perturbations which can prohibit aircraft from terminating at required locations affecting maintenance planning decisions. As such, these effects must be directly considered in the RTAP to reduce the impact of the disruption on the planned solution. Since there is little difference between the formulation of the planning and recovery tail assignment problems, to provide consistency the notation presented in Tables 2.1 and 2.2 along with the additional notation of Table 3.1 is used to describe the RTAP.

There have been many different approaches developed to solve the aircraft recovery problem, with much of the variation related to strategies designed to improve the solution runtimes. These strategies include restricting the set of allowable recovery policies and selecting only a subset of aircraft that are considered in the problem. The RTAP is developed by implementing the recovery options of flight delays and cancellations and aircraft rerouting. These recovery policies

N^D	is the set of disruptable flights, $N^D \subseteq N$
N_{in}	is the set of all carry-in activities j, including flights and origination nodes
N_{out}	is the set of all carry-out activities j, including flights and termination nodes
U_j	is the set of all delay copies v for flight $j \in N$
$\hat{N}$	is the set of all nodes in the connection network defined by flight-copy pairs j_v
$\hat{N}^D$	is the set of disruptable nodes in the connection network defined by flight-copy pairs j_v
$\hat{C}$	is the set of all feasible connections between flight-copy pairs in the network, $C = \{(i_u, j_v) \mid i_u, j_v \in \hat{N} \cup B\}$
M_b	is the minimum number of aircraft required to start the following days flying from base b
z_j	$= 1$ if flight j is cancelled, 0 otherwise
d_j	$=$ the cost of cancelling flight j
o_{bp}	$= 1$ if string p terminates at airport b, 0 otherwise

Table 3.1: Additional notation for the recovery tail assignment problem.

cover the most practical policies available to the airline, omitting the ferrying of aircraft. The decision to formulate a model without aircraft ferrying is two-fold, i) this action is extremely costly for an airline since no passengers are conveyed, and ii) with a greater awareness of anthropogenic climate change, aircraft should be only used for revenue generating purposes. By not including aircraft ferrying the optimal solution to the RTAP includes more flight delays and cancellations, increasing the potential recovery costs. This set of recovery policies implemented for the RTAP are also implemented in the evaluation recovery problems developed throughout this thesis.

One major difference between the planning and recovery problems is the flight schedule used to form the connection network. There are two features of the recovery problem that restrict the flight schedule, the start time of the disruption and the length of the recovery window. A *recovery window* is a period of time during which recovery actions are permitted and commences immediately after a disruption occurs. Thus, the disruptable flight schedule N^D, is defined to contain all flights from N that depart after the disruption occurs but before the end of the recovery window. An additional method for selecting flights to include in the disruptable schedule is based upon which aircraft are considered in the model. The aircraft, and crew for the crew recovery problem, can be partitioned into disruptable and non-disruptable sets where the flight routes for the former can be modified but not for the latter. The use of a recovery window and the selection of aircraft greatly reduces the problem size, having a significant effect on the solution runtimes. The use of a recovery window to reduce the problem size is presented in Chapters 8 and 9 with the results in Section 8.3.2 demonstrating the effect this approximation technique has on solution runtimes.

The implementation of flight delays in the RTAP uses the technique of *flight copies*. The flight copies technique involves the multiple duplication of each flight contained in N with each copy assigned a progressively earlier or later departure time. This technique is used for modelling flight delays, as seen in Thengvall *et al.* [87], and also to model flight retiming options, for example Mercier and Soumis [64]. Since recovery is performed on the day of operations, the only retiming options available to the airline are given by copies representing later departure times. Figure 3.1 provides a simple example of a flight schedule, displayed as a time-line network, with three possible flight copies for each originally scheduled flight. For each flight j the set of allowable copies is given by U_j, with a flight-copy pair described by j_v, where $j \in N$ and $v \in U_j$. Since the flight schedule is partitioned into sets of disruptable and non-disruptable flights, a different set of allowable flight copies must be defined for each flight in the schedule. In the RTAP, recovery actions are only permitted on flights contained in N^D, implying that the flights contained in $N \backslash N^D$ must depart as scheduled. This is achieved by including only a single flight copy in the set $U_j = \{0\}, \forall j \in N \backslash N^D$. Flight delays are permitted on all flights contained in N^D, which is modelled by including the additional copies $v_i, i = 1, \ldots, n$ in the set of allowable copies $U_j = \{0, v_1, \ldots, v_n\}, \forall j \in N^D$. The flight-copy approach is a very simple, but effective, method to model flight delays. The modelling approach and notation presented here is used in all evaluation recovery problems developed in this thesis.

The connection network for the RTAP must include the flight-copies representing delay options for the disruptable flights. Using the definition for the flight-copy pairs, j_v, the set of all flight-copy pairs for the full and disruptable schedule is given by $\hat{N}$ and $\hat{N}^D$ respectively. The definition of a feasible connection is given in Section 2.1.1 and is used to construct the set $\hat{C} = \{(i_u, j_v) | i_u, j_v \in \hat{N}\}$, containing all connections between the flight-copy pairs in $\hat{N}$. Thus, the connection network used for the RTAP is defined by the flight-copy pairs in $\hat{N}$ representing the nodes and the arcs are represented by the connections in $\hat{C}$.

The recovery policy of flight cancellations is modelled in the RTAP with the introduction of the decision variables z_j. In the solution to the RTAP, flight j is cancelled if z_j equals 1 and the flight is covered by an aircraft if z_j equals 0. The cancellation of flight j contributes a cost

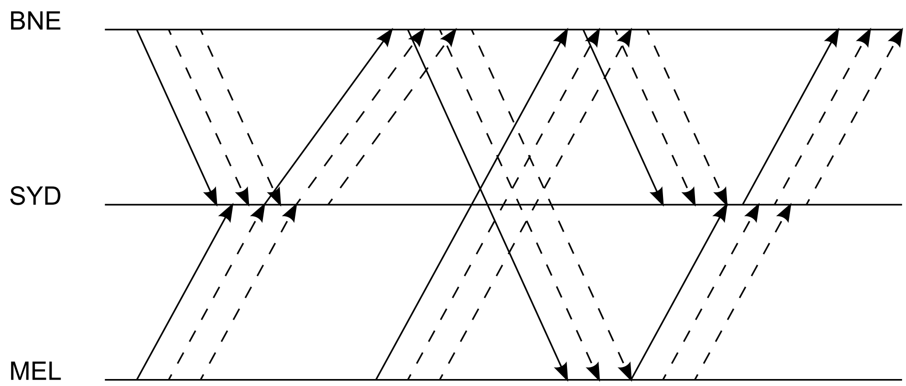

Figure 3.1: Flight copies example. Solid lines represent the scheduled flights, the dashed lines represent the flight copies. For a time-line network, time is on the horizontal axis and the airport locations are provided on the vertical axis. Note: BNE = Brisbane, SYD = Sydney and MEL = Melbourne.

of d_j to the objective function of the RTAP.

A flight string formulation is used for the RTAP to define flight routes spanning a period from the start of the disruption until the end of the recovery window. Since a disruption can occur at any point during the day, the recovery flight strings must originate from the current location of each aircraft. Additionally, at the conclusion of the recovery window the recovery flight strings must terminate at locations to continue the operation of originally planned flight routes. To model this aspect of the recovery problem, the concepts of carry-in and carry-out flights are introduced. A carry-in activity, contained in N_{in}, specifies the current location of an aircraft immediately prior to a disruption, given as flights or origination airports. The carry-in activity describes an origination airport if an aircraft flight route has not commenced prior to the disruption. Similarly, a carry-out activity specifies the expected location of aircraft at the end of the recovery period, given as flights or termination airports. Termination airports are included in N_{out} if there exist originally planned aircraft flight routes that terminate within the recovery period. These concepts are used to define the starting and ending activities of the flight routes constructed for the RTAP. Since a recovery window is implemented in Chapters 8 and 9, the concepts of carry-in and carry-out activities are fundamental to the recovery problem formulations.

Flight strings in the RTAP are constructed to specify the flights performed by each individual aircraft during the recovery period. Using the flight-copy notation, referencing flight j without identifying any copy v collectively represents all flight-copy pairs $j_v, v \in U_j$. So the parameter a_{jp} equals 1 to indicate that flight j, delayed by any length of time, is included on string p. Since the recovery policy of flight delays is implemented in this problem, the departure time of each flight in the string must also be defined. The length of flight delays is used to define the cost for flight string p operated by aircraft r, c_p^r, which is estimated to reflect the indirect cost of passenger dissatisfaction. In addition to describing a set of connected flights, the flight string also indicates end-of-day location b, either aircraft bases or overnight airports, for each aircraft. The set of all end-of-day locations is given by B. While flight strings must terminate at carry-out activities, the parameters o_{bp} equal 1 to indicate that flight string p continues from that activity and terminates at base b at the end of the day. To maintain feasibility for the following days' schedule, a minimum number of aircraft M_b are required to terminate at each end-of-day location b.

The model used to solve the RTAP is defined as,

$$\text{(RTAP)} \quad \begin{aligned} \min \quad & \sum_{r\in R}\sum_{p\in P^r} c_p^r y_p^r + \sum_{j\in N^D} d_j z_j, && (3.1)\\ \text{s.t.} \quad & \sum_{r\in R}\sum_{p\in P^r} a_{jp} y_p^r + z_j = 1 \quad \forall j \in N^D, && (3.2)\\ & \sum_{r\in R}\sum_{p\in P^r} a_{jp} y_p^r = 1 \quad \forall j \in N_{out}, && (3.3)\\ & \sum_{p\in P^r} y_p^r \leq 1 \quad \forall r \in R, && (3.4)\\ & \sum_{r\in R}\sum_{p\in P^r} o_{bp} y_p^r \geq M_b \quad \forall b \in B, && (3.5)\\ & y_p^r \in \{0,1\} \quad \forall r \in R, \forall p \in P^r, && (3.6)\\ & z_j \in \{0,1\} \quad \forall j \in N^D. && (3.7) \end{aligned}$$

The objective of the RTAP minimises the additional cost of aircraft recovery related to flight delays and cancellations. The flight coverage constraints (3.2) ensure that every flight is either covered by an aircraft or cancelled. Since each flight string must terminate at a carry-out activity, constraints (3.3) ensure that each of these activities are covered by exactly one flight string. Constraints (3.4) impose the restriction that each aircraft can only operate at most one route in a feasible solution. Since flight cancellations are implemented as a recovery policy, the flow balance of the original flight schedule is not maintained. Hence, constraints (3.5) impose a lower bound on the number of aircraft required to terminate at each airport to commence the following day's flying.

3.1.2 Crew Duty Recovery Problem

In the sequential airline recovery process, the crew recovery problem is solved following the schedule and aircraft recovery problems. In this framework, it is common to fix the solutions to each of these preceding problems and use the decisions as input. Unfortunately fixing the decisions from preceding stages provides little flexibility to the crew recovery problem resulting in suboptimal, or even infeasible, solutions. An alternative approach involves implementing the recovery policies of flight delays and/or cancellations, permitting changes to the recovered schedule. This approach provides feedback to the preceding stages in the sequential recovery process to improve the solution quality and address the possible suboptimality and infeasibility in crew recovery. The crew duty recovery problem (CDRP) is an example of the latter approach, formulated to include the recovery policies of flight delays and cancellations.

This thesis focuses on the application of recoverable robustness to the tail assignment problem, as such the CDRP is not considered in isolation. The features presented in this section are used to develop an integrated airline recovery problem, integrating the RTAP and CDRP. The formulation of the integrated airline recovery problem is discussed in Chapters 8 and 9.

The CDRP presented in this section is formulated using the flight string notation of Barnhart *et al.* [12], defining strings as a sequence of flights performed by a crew group within a specific time horizon. For the CDRP, the time horizon is defined by a recovery window contained within a single day; as such the focus of this problem is the construction of crew duties. The CDRP is modelled using a framework similar to that used in the formulation of the planning tail assignment problem (PTAP) and RTAP; therefore the notation presented in Tables 2.1, 2.2 and 3.1 is used to describe this problem. However, additional notation given in Table 3.2 is required to describe crew specific recovery policies.

Similar to the crew planning problems, the construction of pairings and duties for crew recovery must respect a set of complex work rules. Since the CDRP is solved using a recovery window within a single day, only the duty rules are considered in the construction of flight strings. The most important duty rules that are considered in the CDRP limit the number of working hours performed in a day, in particular the upper bound on the number of working hours is 13 and only 8 of which can be spent flying.

K	is the set of all planned and reserve crew k
K^{res}	is the set of all reserve crew, $K^{res} \subset K$
P^k	is the set of all flight strings p for crew k
a^v_{jp}	$= 1$ if flight-copy j_v is in string p, 0 otherwise
κ^{v+}_j	= the number crew deadheading on flight-copy j_v
κ^{v-}_j	dummy variable for counting the number of deadheading crew on flight-copy j_v
ν_k	$= 1$ if crew k deadheads back to their crew base from the start of the disruption period, 0 otherwise
g^{DHD}	= the cost of deadheading crew on one leg within a duty
g^{DHB}	= the cost of deadheading crew back to their crew base

Table 3.2: Additional notation for the crew duty recovery problem.

The remuneration structure for crew in the planning and recovery process is defined by a complex function based upon the number of working and flying hours. Barnhart *et al.* [13] presents an example for the cost of a duty for crew k, $DutyCost(k)$, as a function of the flying hours, $fly(k)$, the total elapsed hours, $elapse(k)$, and the minimum number of guaranteed hours, $minGuar$. The cost of a duty is given by,

$$DutyCost(k) = \max\{fly(k), f_d \cdot elapse(k), minGuar\}, \tag{3.8}$$

where $minGuar$ is set at 6 hours [13] and f_d is a fraction that is airline specific. For the crew recovery problem, the cost of recovering a duty for crew k, $RecDutyCost(k)$, is the difference between the cost of the recovered duty and the originally planned duty, $OrigDutyCost(k)$. Therefore, the cost of recovering a crew duty is given by,

$$RecDutyCost(k) = \max\{0, \max\{fly(k), f_d \cdot elapse(k), minGuar\} - OrigDutyCost(k)\}, \tag{3.9}$$

where the parameters $minGuar$ and f_d are set identical to equation (3.8).

The CDRP is formulated to include the recovery policies of flight delays and cancellations, crew deadheading and the use of reserve crew. The recovery policies of flight delays and cancellations are implemented in the CDRP using the same techniques presented in Section 3.1.1. This involves the use of flight copies to model flight delays and the introduction of the decision variables z_j to represent flight cancellations.

The use of reserve crew and crew deadheading are recovery policies specific to the crew recovery problem. Reserve crew, K^{res}, are additional crew located at each crew base that are available to operate duties to prevent the originally planned crew violating work rules. In the event that reserve crew are required, the flight strings assigned to each reserve crew k must also respect the crew duty work rules. While employing reserve crew increases the crew recovery cost, their tactical use can limit the number of required flight delays and cancellations.

Crew deadheading involves the transportation of crew as passengers to reposition them throughout the network. This recovery action is a costly option for airlines since the crew are still paid for the deadheaded flights, and passengers are potentially lost if these flights are fully booked. Two different types of crew deadheading are considered in the CDRP, deadheading within a pairing and back to base. Deadheading within a pairing is modelled using the variables κ_j^{v+} to count the additional number of crew assigned to flight-copy j_v. This also requires the additional variables κ_j^{v-} in the CDRP to ensure that the number of deadheading crew on flight-copy j_v is one less than the total number of assigned crew. Deadheading crew back to base contributes a large cost to the airline, since this action either results in flight cancellations or the use of reserve crew. This type of crew deadheading in modelled in the CDRP using the variables ν_k, which equal 1 if the originally planned crew group k is deadheaded back to base immediately following the disruption or 0 otherwise. The cost of deadheading crew within a pairing and back to base is given by g^{DHD} and g^{DHB} respectively.

As explained in Section 3.1.1, the flight schedule for recovery problems is partitioned into disruptable and non-disruptable flights. Since flight copies are used to model flight delays in the CDRP, the sets $\hat{N}$ and $\hat{N}^D$ are defined in the same manner as for the RTAP. One major difference between the connection network for aircraft and crew is the minimum time between flights that defines a feasible connection. The minimum time between two consecutive flights for crew is called the minimum sit time, which is generally longer than the minimum turn time for aircraft. This has the effect of reducing the total number of possible connections between flights, however it is possible for crew to operate connections less than the minimum sit time provided that same connection is also used by an aircraft. This is an important consideration of the integrated airline recovery problems presented in Chapters 8 and 9. The connections with a ground time between the minimum sit and turn times are call *short connections*.

A recovery window is used in the CDRP to restrict the set of disruptable flights, as such the concepts of carry-in and carry-out activities are defined in the same way as for the RTAP. Using these definitions, the recovered crew duty flight strings span the recovery window by traversing through the connection network from a carry-in to carry-out activities. The recovered crew duties must satisfy the duty work rules, including the hours worked before and after the recovery window.

The CDRP is given by,

$$
\text{(CDRP)}\quad
\begin{aligned}
\min \quad & \sum_{k\in K}\sum_{p\in P^k} c_p^k y_p^k + \sum_{j\in N^D} d_j z_j + \sum_{j\in N^D}\sum_{v\in U_j} g^{DHD}\kappa_j^{v+} + \sum_{k\in K} g^{DHB}\nu_k, && (3.10)\\
\text{s.t.} \quad & \sum_{k\in K}\sum_{p\in P^k} a_{jp} y_p^k - \sum_{v\in U_j}\kappa_j^{v+} + z_j = 1 \quad \forall j\in N^D, && (3.11)\\
& \sum_{k\in K}\sum_{p\in P^k} a_{jp} y_p^k = 1 \quad \forall j\in N_{out}, && (3.12)\\
& \sum_{k\in K}\sum_{p\in P^k} a_{jp}^v y_p^k - \kappa_j^{v+} + \kappa_j^{v-} = 1 \quad \forall j\in N^D, \forall v\in U_j, && (3.13)\\
& \sum_{p\in P^k} y_p^k + \nu_k = 1 \quad \forall k\in K\backslash K^{res}, && (3.14)\\
& \sum_{p\in P^k} y_p^k \le 1 \quad \forall k\in K^{res}, && (3.15)\\
& y_p^k \in \{0,1\} \quad \forall k\in K, \forall p\in P^k, && (3.16)\\
& z_j\in\{0,1\}\ \ \forall j\in N^D, \quad \nu_k\in\{0,1\}\ \ \forall k\in K, && (3.17)\\
& \kappa_j^{v+}\ge 0, \kappa_j^{v-}\ge 0 \quad \forall j\in N^D, \forall v\in U_j. && (3.18)
\end{aligned}
$$

The objective of the CDRP minimises the recovered crew duty cost, the costs related to flight delays and cancellations and the number of deadheaded crew. The cost of a crew flight string c_p^k is the sum of the $RecDutyCost(k)$, given by equation (3.9), and the cost of delays on that string. The constraints (3.11) ensure that every flight is either assigned a crew group or is cancelled. All carry-out activities must be operated by exactly one crew, which is given by constraints (3.12). Constraints (3.13) ensure that exactly one less than the number of crew assigned to flight-copy j_v are deadheading crew. Each of the crew groups originally planned to operate the current days schedule, $k \in K\backslash K^{res}$, must either be assigned a recovered flight string or deadheaded back to base, captured by constraints (3.14). The reserve crew, $k \in K^{res}$, are located at each crew base to operate flight strings if required. Therefore each reserve crew k need not be assigned a flight string in the recovered solution, as described by constraints (3.15). Since there is not a strict restriction on the number of crew available to the airline there is no need to ensure adequate coverage at each crew base at the end of the day. However, all originally planned flight strings are constructed to terminate at either their crew base or at a permissible overnight location. This is also enforced in the construction of each feasible flight string in the CDRP.

3.2 Related Literature

Each stage within the complete airline recovery process presents a very difficult and complex problem. Consequently, there have been various approaches employed to efficiently solve these problems while attempting to achieve high-quality solutions. Such solution approaches for the aircraft and crew recovery problems involve variations on the RTAP and the CDRP, such as including only a subset of all recovery policies or considering only a subset of all aircraft or crew. The stages of the sequential recovery process have different characteristics that are exploited to develop efficient solution approaches.

The recoverable robustness framework solves a recovery problem to evaluate the recoverability of the planning stage solution. As such, the efficiency of the solution approach for the recovery problem has a significant effect on the runtimes of recoverable robustness framework. In this section, various approaches that attempt to improve the solution runtimes are presented. Many of these approaches are employed in the evaluation stage of the recoverable robustness framework implemented in Chapters 5 and 6. In addition, the integrated airline recovery problem is considered in Chapters 8 and 9. Column-and-row generation is used to solve the integrated airline recovery problem in Chapters 8 and 9, which is a contribution to the current solution approaches for this problem type. Finally, Chapter 9 presents a novel, alternative modelling approach for the passenger recovery problem.

The following sections will provide an analysis of the current approaches that have been developed for each stage in the complete recovery process. The discussion will be separated into reviews of the aircraft, crew and passenger recovery stages, given in Sections 3.2.1 - 3.2.3. A current focus of airline recovery solution approaches is the integration of two or more stages from the sequential process. The various methods used to formulate and solve the integrated airline recovery problems will be discussed in Section 3.2.4.

3.2.1 Aircraft recovery

Early work on airline recovery is presented by Teodorović and Guberinic [84] focusing specifically on the aircraft recovery problem. The aircraft recovery problem of [84] attempts to minimise the amount of delay experienced by passengers when one or more aircraft become unavailable. This optimisation model is developed by implementing only the recovery policy of flight delays and is solved using a branch-and-bound approach. This work is extended by Teodorović and Stojković [85] to include the additional recovery policy of flight cancellations. In [85], the problem is formulated as a multi-objective optimisation problem, attempting to minimise the amount of delay and the number of cancelled flights. Extending upon [84] and [85], Teodorović and Stojković [86] present an aircraft recovery problem considering the effect of recovery actions on crew. This work is one of the first attempts to develop an integrated airline recovery problem.

The *time-line network* is the most common approach used to describe the flight schedule in the aircraft recovery problem. An example of an aircraft recovery problem developed using this network definition is presented by Jarrah *et al.* [49]. Two alternative models are developed by Jarrah *et al.* [49] to focus on specific recovery policies, one using flight delays and the other using cancellations. While both recovery policies of flight delays and cancellations are considered in [49], since they are included in two separate models it is difficult to evaluate the trade-off between the recovery decisions. This limitation is addressed in an extension presented by Cao and Kanafani [21, 22]. In [21] and [22], the authors develop a quadratic zero-one programming model implementing recovery policies of flight delays and cancellations, and aircraft ferrying.

The use of the time-line network is developed further by Yan and Yang [97], presenting a unique design that concisely describes the effect of disruptions on the planned schedule. By employing this network design, Yan and Yang [97] efficiently solve the aircraft recovery problem with a full set of recovery policies. The network design developed by Yan and Yang [97] has received considerable attention, with two key extensions presented by Yan and Tu [96] and Yan and Lin [95]. Yan and Tu [96] consider a very similar model to [97], extended to consider multiple fleet types and Yan and Lin [95] develop an aircraft recovery problem with a specific focus on airport closure disruptions.

Thengvall *et al.* [87] follows on from Yan and Yang [97], by presenting an alternative aircraft recovery problem formulated on a time-line network. A key feature of [87] is the objective of minimal deviation from the planned aircraft routes, which is introduced to produce a more *human friendly* solution. This objective is achieved by adding protection arcs to the network to preserve the use of connections that are included in the original aircraft flight routes. Thengvall *et al.* [87] also demonstrates the use of flight copies to model flight delays in the aircraft recovery problem. The concept of minimal deviation is fundamental to the recoverable robustness framework, with an alternative modelling approach investigated in Chapters 5 and 6. In addition, the minimal deviation objective is implemented in the integrated airline recovery problems in Chapters 8 and 9.

Two extensions to [87] are presented by Thengvall *et al.* [89] and Thengvall *et al.* [88] to consider multiple fleets, allowing ferry flights and performing aircraft swaps between fleet types and subtypes. In [87], the authors present an interesting discussion stating that the costs associated with flight delays and cancellations are difficult to quantify and are generally provided as an estimate. An alternative approach is suggested whereby the costs are set by the operations controller in the form of weights to achieve a recovered solution with desired characteristics.

The *time-band network* presented by Argüello [8] is developed as an alternative network design to provide an improvement in the solution runtimes of the aircraft recovery problem. To reduce the size of the recovery network, time is discretised into bands and all activities occurring in these bands are aggregated by location. The discretised time-bands form an approximation of the original network with the aggregation of nodes, however this comes at the expense of errors in the estimation of the delay costs. Since the exact arrival and departure times are no longer available, aircraft connections are made using a first-in-first-out policy. An aircraft recovery model developed on a time-band network is presented by Bard *et al.* [11] and solved as a mixed-integer program. This problem is solved with different time-band lengths to evaluate the sensitivity of the runtimes and solution to this network approximation. The authors demonstrate that as the length of the time bands increase, the runtimes of the model decrease and, as a result of the approximation, the solution cost also decreases. The results for the approach indicate its potential use in a real-time environment.

The time-band network is further explored in the aircraft recovery problem developed by Eggenberg *et al.* [35]. In [35], the time-band network is extended by including aircraft maintenance opportunities and generating individual networks for each aircraft. The inclusion of aircraft maintenance opportunities in the network description is used to limit the effect of recovery actions on the maintenance feasibility of each aircraft. As explained in Chapter 2, maintenance planning is an important consideration of the airline business process since severe penalties exist for exceeding the set maintenance limits. The results presented improve upon previous approaches developed using this network design in regards to the recovery cost and solution runtimes.

The *connection network* is a network design traditionally used for airline planning and recovery problems. Rosenberger *et al.* [73] present an aircraft recovery problem using this network design, employing the recovery policies of flight delays and cancellations and the rerouting of aircraft. Given the potential size of this problem, the authors employ a heuristic to select a minimal number of aircraft to include in the model for possible rerouting. Heuristic approaches to restrict the affected equipment considered in recovery models is a common method applied to reduce the problem complexity and improve solution runtimes.

The cost of recovery must account for a number of different factors including the additional cost of fuel for aircraft and the use of any extra resources. In addition to the many real and tangible costs for the airline, there are indirect costs, such as the effect of a disruption on passengers, that are difficult to quantify. Andersson [6] considers the difficulty in calculating all airline recovery costs and presents two different methods in which they can be defined. The first method uses exact costs for each feature of the recovery problem, and the second uses weights to achieve a solution with desired characteristics. The advantage of using real costs is that the optimal recovery solution estimates the cost that will be incurred by implementing recovery actions. However, the use of weights, as presented by Thengvall *et al.* [87], empowers the operations controller to find a recovery solution that enhances particular characteristics.

The recoverable robustness framework requires the evaluation stage to provide high quality feedback in short runtimes. It is observed from the solution approaches presented above that the techniques employed either improve the solution quality or runtimes, but not both. For example, a feature of the time-line network [21, 22, 49, 87–89, 95–97] is the accurate description of the recovery problem. However, this network design negatively affects solution runtimes as a result of a very large problem formulation. By contrast, the time-band network [8,11,35] approximates the recovery problem to achieve fast solution runtimes. To satisfy the requirements of the recoverable robustness framework, Chapters 5 and 6 employ a connection network [73] in the evaluation stage. This network description accurately models the recovery problem and is consistent with the modelling approach used for the planning stage. Additionally, high quality feedback is achieved in Chapters 5 and 6 by employing a full set of recovery policies in the evaluation stage, modelled using real costs.

3.2.2 Crew recovery

The crew recovery problem is the recovery stage that has received the most interest due to the high operational costs. The cost of crew represents the second largest cost to an airline, which is significantly impacted by schedule perturbations. Comparing the crew and aircraft recovery problems, the crew problem involves a larger number of individual flight strings and complex work rules resulting in a more complex problem. During the airline planning stages, each crew member is assigned a personalised schedule detailing a set of flights to work within a duty period. A major difficulty that arises during a disruption is the management of these personalised schedules, attempting to minimise the number of changes that are required.

The research regarding the crew recovery problem can be classified into two different modelling approaches, using a fixed flight schedule or allowing flight delays and cancellations. The fixed flight schedule models are designed to operate within the sequential recovery process where the schedule and aircraft recovery is solved first and then fixed for subsequent stages. This technique simplifies the crew recovery problem by reducing the size of the recovery network and limiting the possible recovery decisions. Alternatively, allowing flight delays and cancellations increases the complexity of the crew recovery model. However, this modelling approach improves the solution quality and the possibility of an infeasible crew recovery solution is reduced. The solution approaches developed for both modelling types are discussed in the following sections.

Fixed flight schedule

An example of the crew recovery problem to fit within the sequential recovery process is presented by Wei *et al.* [91]. The crew recovery problem, described as the crew pairing repair problem, is formulated as a multi-commodity flow problem with an objective to repair the broken pairings with as little modification to the planned solution as possible. The authors implement a depth-first search to solve the crew recovery problem, attempting to replicate the actions performed by the airline operations control centre. The search process involves the generation of a small set of pairings, which are tested for feasibility and help direct the next step in the search. Experiments demonstrate very fast runtimes for this solution approach, with the first feasible solution found within two seconds for all cases.

The operational airline crew scheduling problem is proposed by Stojković *et al.* [83] as an integer non-linear multi-commodity flow problem. A key contribution of this approach is the consideration of the planned monthly blocks for each crew member during the recovery process. Since this model uses a fixed flight schedule, the recovery policies include the over or under covering of flights, within the airline requirements, to form feasible recovery solutions. The schedule is restricted to an operational period of one or seven days to reduce the problem size and improve solution runtimes. The tasks that are performed by the crew preceding and succeeding this period are frozen, with operations expected to be back to plan by the end of the operational period. This method of fixing activities is investigated further in Chapters 8 and 9 in regards to carry-in and carry-out activities. Additionally, the authors implement a heuristic approach to reduce the complexity of recovery problem by selecting a subset of crew to include in the model. The technique of column generation coupled with an early branching strategy is employed to solve this problem.

Medard and Sawhney [62] present a crew recovery problem as the integration of the crew pairing and rostering problems. This problem is solved using two different approaches, a depth-first search heuristic and column generation. The solution runtimes of this problem are improved by employing a preprocessing stage to construct crew dependent networks used in the generation of feasible crew pairings. The depth-first search approach involves the traversal of these crew dependent networks to generate pairings, which are checked for legality using the Carmen Rave system. Column generation is implemented with a pricing subproblem that identifies the k-shortest paths, each of which are also checked for legality using the Carmen Rave system. The two different methods demonstrate fast solution runtimes, with the best performance observed from the depth-first search.

There are significant differences between the crew remuneration structure for US and European airlines that greatly affect the solution to the crew recovery problem. Nissen and Haase [67] highlight these differences and develop a crew recovery problem focusing specifically on European airlines. The major difference presented by [67] is that Europe commonly remunerate crew using a salary as opposed to a wage in the US. Since very little difference in crew costs is observed by European airlines during recovery, Nissen and Haase [67] present a model to minimise the number of changes that are made to an individual crew members planned pairing. The model partitions the crew members by their qualification group to provide a higher solution quality. To achieve fast solution runtimes a recovery window is implemented to reduce the number of flights considered in the recovery problem. This set of flights is further reduced by identifying the disruptable flights on which recovery actions are performed. The authors present a series of results using different lengths of recovery windows demonstrating the trade-off between the solution runtime and quality.

Using flight delays and cancellations

An unfortunate result from using a fixed flight schedule is the limited flexibility provided to the crew recovery problem causing suboptimal, or even infeasible, results. Solving the crew recovery problem with flight delays and cancellations attempts to avoid this difficulty, facilitating a feedback process with the aircraft recovery problem. One of the first examples of the crew recovery problem considering flight cancellations is presented by Lettovsky *et al.* [56]. This work extends upon Johnson *et al.* [50] and forms part of the PhD thesis of Lettovsky [55]. The authors present a number of approaches that attempt to reduce the computation time of the recovery algorithm such as a heuristic to select the included crew and compact storage for the generated columns. The model is tested against three scenarios, which include different numbers of affected crew. In all cases the solution times are within an acceptable range for use in an online situation.

An extension on the fixed schedule crew recovery problem by Stojković *et al.* [83] is given by Stojković and Soumis [81] considering the use of flight delays. Flight delays are implemented using the technique of time windows, with the windows individually defined for each flight based upon operational constraints. As a further extension, Stojković and Soumis [82] address a more realistic problem by considering multiple crew types within each crew group. Column generation is used to solve both [81] and [82] with the subproblem solved by a multi-label dynamic programming algorithm. The results presented demonstrate high quality solutions, but unfortunately the solution runtimes do not encourage use in an online environment.

A novel approach to the crew rescheduling problem is presented by Abdelghany *et al.* [1], formulated using the recovery policies of flight delays and cancellations. A novel solution approach is presented in [1] where the flight schedule is partitioned into chronologically ordered sets of resource-independent flights. Resource independence implies that each crew group can only operate a single flight within each set. A sequential process is then employed by solving a crew recovery problem for each set of flights. A number of preprocessing steps are employed to improve the computation time of the optimisation problem, including shifting the disruption to the hubs and using a heuristic to identify the included crew. This model is evaluated against one test case, demonstrating the solution runtime improvements achieved by this approach for this crew recovery problem.

While solving the crew recovery problem solved using flight delays and cancellations increases the problem flexibility, global optimality is still affected by the sequential solution approach. It is necessary to integrate the aircraft and crew recovery problems to improve the solution quality of the airline recovery process. The integrated airline recovery problem is the focus of Chapters 8 and 9, with the development of a novel passenger recovery approach in Chapter 9.

3.2.3 Passenger recovery

Passenger recovery is the final stage in the sequential recovery process that attempts to identify new itineraries for all disrupted passengers. While there are no direct costs associated with passengers, the indirect costs related to passenger satisfaction must be considered. Assessing the impact of disruptions on passengers is very difficult, as such few recovery approaches exist that explicitly consider passenger flows through the network. The approaches that have been developed for passenger recovery are generally formulated as part of an integrated problem with aircraft or crew. The integration of passenger recovery with other stages aids in making flight delay and cancellation decisions that minimise the impact of disruptions on passengers.

One of the first examples of a passenger recovery problem is presented by Bratu and Barnhart [19], which also considers aircraft routing and the use of reserve crew. The authors describe the difficulty of modelling the costs associated with passenger delays, and develop two different models as part of this analysis. The first model attempts to minimise the number of passengers that are disrupted by a schedule perturbation. In this case, the cost for disrupting each passenger assigned to the same itinerary is identical, and there is no consideration to the actual cost of delay. Consequently, the objective value of this problem is an approximation of the true cost of the disruption. The second model more accurately describes the cost of disrupting passengers by providing an estimate of the actual delay costs. The indirect cost related to passenger satisfaction is an important consideration of any passenger recovery problem. However, this indirect cost can only be calculated as an estimate.

An example of an integrated aircraft and passenger recovery problem is presented by Jafari and Zegordi [48]. This integrated problem is solved by using the modelling approach presented by Abdelghany *et al.* [1]. The objective of this problem is to minimise the costs associated with reassigning aircraft to recovered flight routes and the construction of new itineraries for disrupted passengers. The costs associated with disrupted passengers are provided only as an estimate, further demonstrating the difficulty in accurately modelling the true cost of disruptions.

The integrated aircraft and passenger recovery problem was the focus of the 2009 ROADEF challenge. This challenge considered real world instances of an airline experiencing schedule disruptions with a requirement to identify a solution within 10 minutes. The objective of this challenge was to minimise the operating costs of the airline, related to the direct costs of aircraft and ground services, and indirect costs of passenger inconvenience. The passenger inconvenience costs attempt to model the effect of flight delays, missed connections and downgrading of flight class. The best solution approach for this problem is presented by Bisaillon *et al.* [17], solving the recovery problem with a large neighbourhood search heuristic.

The consideration of passengers in the recovery process has a significant impact on the operational performance of the airline. Therefore, considering passengers in the evaluation stage of the recoverable robustness framework is expected to provide a higher quality feedback to improve the planning stage solution. Chapter 9 investigates a novel modelling approach for passenger recovery that identifies alternative travel arrangements for disrupted passenger in the event of a flight cancellation. The approach in Chapter 9 will present an effective passenger recovery without requiring the unnecessary complexity of identifying new itineraries for disrupted passengers.

3.2.4 Integrated recovery

The solution approaches developed for the complete airline recovery problem involve a trade-off between solution quality and runtime. The sequential solution approach is a prominent example of this with suboptimal results arising from fixed scheduling decisions. The previous sections describe examples of approaches developed for each of the recovery stages to either improve the runtimes or solution quality. In the same manner, the integrated recovery problem has been considered to alleviate the suboptimal results from the sequential recovery process. Since the integrated problem is a very large and complex problem, the trade-off between solution runtimes and quality is still very evident. Improvement in solution techniques and computing capabilities has aided the development of integrated airline recovery problems, which are solvable in real time.

An early attempt to develop an integrated airline recovery is presented in the PhD thesis of

Lettovsky [55]. This problem integrates all aspects of the airline recovery process, which includes schedule, aircraft, crew and passenger recovery. To address the potential size of this problem, Benders' decomposition is applied to improve the tractability of the model by separating each recovery stage into individual subproblems. The subproblem for crew recovery is implemented separately in Lettovsky *et al.* [56], as a demonstration of the modelling approach. Unfortunately the integrated recovery problem is not implemented, so no evaluation of the solution approach is provided.

The crew recovery model of Abdelghany *et al.* [1] is extended by Abdelghany *et al.* [2], with the integration of pilots, flight attendants and aircraft. The sequential solution approach developed by Abdelghany *et al.* [1] is also applied in Abdelghany *et al.* [2] as a further demonstration of this technique. The benefits of integrating the aircraft and crew in the optimisation model ensures that the recovered solution remains feasible for all resources with consistent flight delays and cancellations. The model is evaluated against a set of scenarios based upon disruptions caused by the US Ground Delay Program. The results from experiments show that this model is able to achieve significant delay reductions within very short runtimes.

The Benders' decomposition framework presented by Lettovsky [55] for the integrated recovery problem is developed further by Petersen *et al.* [70]. The approach presented by Petersen *et al.* [70] combines the solution techniques of Benders' decomposition and column generation to improve the tractability of the problem. To further reduce problem complexity, a number of preprocessing steps are executed to select a subset of flights, aircraft and crew to include in the model. The integrated recovery problem is evaluated against a number of scenarios, which include varying the flow rate of aircraft for different lengths of time at hub and spoke airports. The results demonstrate that the integrated approach is able to achieve a lower recovery cost than the equivalent sequential recovery process. The runtimes for this model are promising for online applications where most of the presented scenarios are solved within the set 30 minute time limit.

It is clear from the above approaches that decomposition techniques are required to efficiently solve the integrated airline recovery problem. Unfortunately, the decomposition techniques of Benders' decomposition [55, 70] and the partitioning of the flight schedule [2] do not guarantee integral optimality. Chapter 8 solves the integrated airline recovery problem using column-and-row generation to improve solution runtimes while providing a guarantee of near optimal solutions.

3.3 Summary

Airline recovery is a necessary aspect of disruption management to continue operation of aircraft routes and crew pairings following schedule perturbations. The regular nature of disruptive events and the potential increase in operating costs indicates the need for an efficient and accurate recovery process. Since airline recovery is a large and intractable problem, there is a trade-off between the solution runtime and quality. This chapter has presented various techniques that have been developed to address these issues and efficiently solve airline recovery problems. The column-and-row generation framework developed in Chapter 7 is applied to the integrated airline recovery problem in Chapters 8 and 9 as a contribution to the previously discussed solution methods.

The vast majority of literature presented in this chapter attempts to improve the runtime and quality through solution and modelling techniques for the recovery problem. A key feature of recoverable robustness is identifying planning solutions that are recoverable in *limited effort*. By contrast, the intelligent planning decisions achieving *limited effort* improve the runtimes and quality of the airline recovery process. The recoverable robustness framework has not previously been applied to airline planning problems, and its application with a full set of recovery policies in Chapters 5 and 6 is a novel extension to the approach.

The following chapter will present a collection of solution techniques commonly applied to airline planning and recovery problems. These techniques will be employed to a variety of mathematical models presented throughout this thesis.

Chapter 4

Solution Methods

In the previous chapters, examples of the mathematical models and current modelling approaches for airline planning and recovery problems were introduced. It was explained that due to a high level of complexity, the complete planning and recovery problems are commonly separated into a series of sequential stages in an attempt to improve tractability. While each stage in the planning and recovery processes is much simpler to solve than the complete problem, these stages still fit within the class of large-scale optimisation problems. There have been a variety of solution approaches that have been developed specifically for this problem class, both heuristic and exact methods. The heuristic approaches involve exploiting problem characteristics to quickly identify feasible solutions, while exact methods use decomposition techniques to define more tractable formulations. In this thesis, the problems developed for airline planning and recovery are solved using exact solution methods. In addition, strategies and new techniques are investigated to improve upon the current solution methods.

Two exact solution methods commonly applied to large-scale optimisation problems are *column generation* and *Benders' decomposition*. Column generation is a solution technique that is used to solve problems with a large number of variables, each displaying a special combinatorial structure. Classical examples of problems solved by column generation are the cutting stock and bin packing problems. This solution approach involves the construction of a master and subproblem, where the subproblem dynamically generates variables (columns) to include in the master problem. By contrast, Benders' decomposition is applied to mixed-integer programs that have a large number of constraints that display a special structure, such as block-diagonal with a set of linking variables. The most prominent examples of problems applying Benders' decomposition are in the field of stochastic programming. This decomposition approach involves the construction of a master problem, as a relaxation of the original problem, and a series of independent subproblems. The solution to each subproblem is used to generate cuts which tighten the relaxation of the master problem. The application of column generation and Benders' decomposition to airline planning and recovery problems is discussed in Chapters 2 and 3 as methods to improve the efficiency of the solution process.

Airline planning and recovery problems are formulated as mixed-integer programs, generally requiring the use of *branch-and-bound* to identify the optimal integer solution. The approaches of column generation and Benders' decomposition aid in efficiently solving the linear programming (LP) relaxation to optimality, however this solution may be far from integer optimality. Branch-and-bound is an approach that systemically partitions the feasible region of integer variables by restricting the variable bounds. Examples of branching techniques will be presented in Section 4.3, including a description of *branch-and-price*, which is the integration of branch-and-bound and column generation.

4.1 Column Generation

The column generation solution approach efficiently solves large-scale optimisation problems by exploiting the fact that most variables are non-basic in the optimal solution. This involves defining a smaller, more tractable master problem as a restriction on the original problem and a subproblem to identify columns to add to the master. An important characteristic of problems solved by column generation is the structure of the variables that permits their generation with the solution to a subproblem. Airline planning and recovery problems display such a structure with flight string variables defining a connected path through a network. As such, the subproblem can be formulated as a network flow problem and solved by various dedicated solution algorithms. A detailed review of column generation and related applications is provided in Desaulniers *et al.* [29] and Lübbecke and Desrosiers [59]. This section will describe the column generation solution approach using the planning tail assignment problem (PTAP), presented in Section 2.1.1, as an example to direct the discussion.

The column generation approach described in this section is applied to solve the planning and evaluation stages of the recoverable robustness problems in Chapters 5 and 6. An extension of this solution approach, column-and-row generation, is investigated in Chapter 7. This extension is then applied to solve integrated airline recovery problem in Chapters 8 and 9, evaluating the improvement in solution runtime and quality against the standard column generation approach.

4.1.1 Master problem

The restricted master problem (RMP) for the column generation solution approach is initially formulated with a subset of all possible variables from the original problem. This initial problem formulation significantly reduces the complexity of a previously intractable problem. For example, the PTAP is formulated using flight strings that describe paths through a connection network, resulting in a highly combinatorial set of variables. Since a set of flight string variables, P^r, are constructed for each aircraft r considered in the problem, it is not difficult to see the potential size of this problem. The enumeration of all possible variables results in an extremely large and intractable problem that is difficult to solve with the use of commercial solvers. To address this difficulty, the RMP is formulated to contain only a very small subset of all variables, $\bar{P}^r \subseteq P^r \; \forall r \in R$, forming a more easily solvable problem.

The RMP is constructed as a smaller, restricted version of the original problem, therefore the optimal solution can be found very quickly. By the LP duality theory, the optimal dual solution is also found and used to calculate the reduced cost of all variables. The dual solutions for the PTAP are defined as $\boldsymbol{\beta} = \{\beta_j | \forall j \in N\}$ and $\boldsymbol{\gamma} = \{\gamma^r | \forall r \in R\}$ for constraint (2.6) and (2.7) respectively. Using these definitions, the reduced cost of variable p for aircraft r is given by,

$$\bar{c}^r_p = c^r_p - \sum_{j \in N} \beta_j a_{jp} - \gamma^r. \tag{4.1}$$

Since the RMP is solved to optimality, the variables $p \in \bar{P}^r, \forall r \in R$ have a reduced cost $\bar{c}^r_p \geq 0$ and $\bar{c}^r_p = 0$ for all basic variables. This property implies that any improvement in the objective function of the RMP is only possible through the addition of variables. A subproblem is solved to identify variables with a negative reduced cost to add to the RMP.

4.1.2 Subproblem

The restriction on the original problem given by the RMP is relaxed through the addition of variables generated in the column generation subproblem. Only variables that are expected to improve the objective value of the RMP are added to the problem. These variables are identified by having a reduced cost $c^r_p < 0$. While any variable with a negative reduced cost will provide an improvement to the objective of the RMP, a minimisation problem is formed to ensure that such a variable is found. In general, the column generation subproblem for each aircraft r is of the form,

$$\hat{c}^r = \min_{p \in P^r} \{c_p^r - \sum_{j \in N} \beta_j a_{jp} - \gamma^r\}. \tag{4.2}$$

The strength of the column generation solution approach relies on identifying the variable structure to formulate a more tractable subproblem. As explained previously, airline planning and recovery problems exhibit a variable structure permitting the formulation of the subproblem as a network flow problem, or shortest path problem. Identifying this variable structure allows the set of variables P^r to be defined as the feasible solutions of a shortest path problem. The shortest path problem used to solve (4.2) for the PTAP will be described using the notation presented in Table 2.1.

The shortest path problem involves identifying a set of connections (i, j) to form a path through the connection network C with a minimum cost. This is modelled by the decision variables w_{ij}^r that equal 1 to indicate that aircraft r uses connection (i, j) in a connected flight route through the network and 0 otherwise. In the PTAP, an aircraft flight route may originate and terminate at airports $b \in B$, representing the source and sink nodes of the network. For each aircraft r a single source node, given by s_r, is defined and the allowable sink nodes are all $b \in B$. Using these definitions, the column generation subproblem for aircraft r is given by,

$$\hat{c}^r = \min \sum_{(i,j) \in C} c_{ij}^r w_{ij}^r - \sum_{(i,j) \in C} \beta_j w_{ij}^r - \gamma^r, \tag{4.3}$$

$$\text{s.t.} \sum_{\substack{i \in N \\ |(i,j \in C)}} w_{ij}^r \leq 1 \quad \forall j \in N, \tag{4.4}$$

$$\sum_{\substack{j \in N \\ |(s_r,j \in C)}} w_{s_r j}^r = 1 \quad \forall j \in N, \tag{4.5}$$

$$\sum_{b \in B} \sum_{\substack{j \in N \\ |(i,b \in C)}} w_{ib}^r = 1 \quad \forall i \in N, \tag{4.6}$$

$$\sum_{\substack{i \in N \cup \{s_r\} \\ |(i,j \in C)}} w_{ij}^r - \sum_{\substack{k \in N \cup B \\ |(j,k \in C)}} w_{jk}^r = 0 \quad \forall j \in N, \tag{4.7}$$

$$w_{ij}^r \in \{0, 1\} \quad \forall (i, j) \in C. \tag{4.8}$$

The objective of (4.3)-(4.8) is to find the minimum reduced cost variable $p \in P^r$. Constraints (4.4) ensure that each flight is included at most once in flight string p. The origination and termination of the flight string is given by constraints (4.5) and (4.6) such that aircraft r originates from s_r and terminates at any $b \in B$ respectively. Finally, constraints (4.7) ensure that a connected flight route through the network is identified. Constraints (4.7) are called *flow balance* constraints which force the number of incoming connections to each node to equal the number of outgoing connections. The solution to (4.3)-(4.8) defines a sequence of arcs to form a connected path from the source node to a sink node.

Given a feasible solution to the column generation subproblem, the flight string variable p can be constructed by setting the parameters a_{jp} by the following expression,

$$a_{jp} = \sum_{\substack{i \in N \\ |(i,j) \in C}} w_{ij}^r \quad \forall j \in N. \tag{4.9}$$

While the feasible region of the column generation subproblem describes all variables contained in P^r, it is not possible that the solution to this problem will identify a variable $p \in \bar{P}^r$ with a negative reduced cost. This stems from the statement made previously where all variables contained in the RMP have a reduced cost of $c_p^r \geq 0$. Therefore, only variables $p \in P^r \backslash \bar{P}^r$ will be identified by the solution to the subproblem to add to the RMP.

Column generation is an iterative solution process that involves i) solving the RMP to optimality to identify an upper bound on the original problem and the current optimal dual solution, and ii) solving the subproblem to identify negative reduced cost columns to improve upon this upper bound. This process continues until no further columns are identified by the subproblem to add to the RMP, which is indicated by $\hat{c}^r \geq 0, \forall r \in R$. Since the minimum reduced cost variables for all aircraft are non-negative, this indicates that no further improvement to the objective of the RMP can be made. Therefore, the current solution to the RMP is the optimal solution to the original problem.

4.1.3 Solution algorithms

Since the column generation subproblem, given by (4.3)-(4.8), is defined as a mixed integer program, improved efficiency in the solution process is achieved through the use of problem specific algorithms. As mentioned previously, the subproblem for the PTAP is defined as a network flow problem for which a number of solution algorithms have been developed. Examples of the available solution algorithms for network flow problems are presented by Ahuja *et al.* [5]. The solution algorithms presented in this section are used throughout this thesis.

Shortest path problem

The column generation subproblem, given by (4.3)-(4.8), is a shortest path problem solved by a reaching algorithm. Such a reaching algorithm is described by Algorithm 4.1. A reaching algorithm is a method of finding the shortest path by propagating the distance from the current node to all connected nodes of a higher index value [5]. This algorithm is employed to solve the column generation subproblems for the various models developed throughout this thesis.

The connection network that is used for the PTAP is an acyclic directed graph. This network structure is common across airline planning and recovery problems. A benefit of this network structure is that the nodes can be sorted in a topological order, where node i is ordered before node j if $\exists(i, j) \in C$ [5]. This ordering of the nodes aids in the development of efficient solution algorithms since the shortest path to each node is found by examining each arc only once. The complexity of such a solution algorithm is $O(m)$, where m is the number of arcs in the connection network.

The parameters and variables that are used to describe Algorithm 4.1 are introduced in Table 4.1. Each of the nodes within the network are identified by the indices i and j, with the arc connecting the two nodes labelled as (i, j). In the network for the PTAP there are multiple source and sink nodes representing the overnight airports for aircraft. Since Algorithm 4.1 finds the shortest path for a single aircraft, only one source node is considered and labelled -1 for convenience. Also, the aircraft flight route is permitted to terminate at any overnight airport in the network, so multiple sink nodes are considered in the algorithm.

There are two main variables in Algorithm 4.1 that are used to calculate the shortest path through the network, $dist(j)$ and $prev(j)$. The variable $dist(j)$ is used to store the distance of the shortest path to node j and $prev(j)$ stores the node i that immediately precedes node j in that path. In the initialisation of Algorithm 4.1, the distance accumulated at the source node is set to 0, i.e. $dist(-1) \leftarrow 0$, and since there are no nodes preceding the source, the

$dist(j)$	is the distance stored at node j
$dist(-1)$	is the distance stored at the source node, set to 0
$prev(j)$	is the node, i, previous to node j in the shortest path to j, $(i, j) \in C$
$prev(-1)$	is the previous node to the source node, set to -1
$cost(i, j)$	is the cost of including connection $(i, j) \in C$ in the shortest path

Table 4.1: Definitions for variables used in Algorithm 4.1.

Algorithm 4.1 Algorithm to find the shortest path through an acyclic network

Finding the shortest distance

Set $dist(-1) \leftarrow 0$ and $prev(-1) \leftarrow -1$ for the source node,
set $dist(i) \leftarrow \infty$ and $prev(i) \leftarrow -2$ for all nodes $i \in N$.
for all nodes, i, in the topologically sorted list **do**
  for all nodes, j, such that $(i, j) \in C$ **do**
    Set $tempDistance \leftarrow dist(i) + cost(i, j)$.
    if $tempDistance < dist(j)$ **then**
      Set $dist(j) \leftarrow tempDistance$,
      set $prev(j) \leftarrow i$.
    end if
  end for
end for

Identifying the path with the shortest distance

Let i be the sink node for the shortest path.
while $prev(i)$ is not the source node **do**
  Add $prev(i)$ to the shortest path,
  set $i \leftarrow prev(i)$.
end while

previous node is set to $prev(-1) \leftarrow -1$. For all other nodes in the network, the distance is set to $dist(j) \leftarrow \infty$ and the previous node is arbitrarily assigned to $prev(j) \leftarrow -2$. The shortest path from the source node to a sink node is given by a set of nodes j and a set of connections (i, j). Using the cost for each connection (i, j), $cost(i, j)$, the distance of the shortest path is calculated as the sum of this cost for all connections that exist in the path.

The execution of Algorithm 4.1 involves processing of each node in the network and updating the shortest path to any connected node if required. The processing of a node i calculates the shortest path from i to each connected node j. This involves taking the distance stored at the current node $dist(i)$ and adding on the connection cost $cost(i, j)$. If $dist(j) > dist(i) + cost(i, j)$, then the distance of the shortest path to node j must be updated, $dist(j) \leftarrow dist(i) + cost(i, j)$. Additionally, the previous node in the shortest path to node j must also be updated to $prev(j) \leftarrow i$.

The algorithm terminates when all nodes in the topologically sorted list have been processed, indicating that all arcs have been examined. The shortest path is then found by identifying the sink node where the minimum distance is stored. A path is made through the network from this sink node back to the source by identifying the previous node stored in the variables $prev(j)$. The shortest path is constructed by appending node $i = prev(j)$ to the path and iteratively updating $j \leftarrow i$ until the source node is reached.

Multi-label shortest path problem

Algorithm 4.1 can be extended to store multiple labels at each node in the network to track resources in addition to cost. This is a required modification to solve the subproblem of crew planning and recovery problems. The use of multiple labels for these problems arises from the many resources that must be tracked through the network, such as flying and working hours for crew, to satisfy the complex work rules. The labels allow for paths with a suboptimal distance but a favourable resource consumption to be retained for further propagation through the network. Examples of multi-label shortest path problems are presented by Desrochers and Soumis [30] and Dumitrescu and Boland [32].

Alternative definitions for the parameters in Table 4.1 are given in Table 4.2 to introduce an additional index to represent label l at node j. The multi-label shortest path problem given by Algorithm 4.2 is described using the notation provided in Table 4.2. Algorithm 4.2 is implemented in Chapters 8 and 9 to solve the column generation subproblem for the crew duty recovery problem.

While a similar structure is observed in Algorithms 4.1 and 4.2, there are key differences

$dist(j, l)$	is the distance to node j stored on label l
$consum(m, j, l)$	is the consumption of resource m to node j stored on label l
$prev(j, l)$	is the node, i, previous to node j in the shortest path to j stored on label l, $(i, j) \in C$
$prevLabel(j, l)$	is the label, k, extended from $prev(j, l)$ to node j in the shortest path to j stored on label l
$use(m, i, j)$	is the amount of resource m consumed by including connection $(i, j) \in C$ in the shortest path
$maxLabels$	the maximum number of labels that can be stored at a node

Table 4.2: Definitions for variables used in Algorithm 4.2.

Algorithm 4.2 Multi-label algorithm to find the shortest path through an acyclic network

Finding the shortest distance

Set $dist(-1) \leftarrow 0$ and $prev(-1) \leftarrow -1$ for the source node,
set $dist(i, 0) \leftarrow \infty$ and $prev(i, 0) \leftarrow -2$ for all nodes $i \in N$.
for all nodes, i, in the topologically sorted list **do**
    for all labels, k, stored at node i **do**
        for all nodes, j, such that $(i, j) \in C$ **do**
            Set $tempDistance \leftarrow dist(i, k) + cost(i, j)$,
            set $tempConsum(m) \leftarrow consum(m, i, k) + use(m, i, j), \forall m$.
            Create new label l,
            set $dist(j, l) \leftarrow tempDistance$ and $consum(m, j, l) \leftarrow tempConsum(m), \forall m$,
            set $prev(j, l) \leftarrow i$ and $prevLabel(j, l) \leftarrow k$.
            Using a dominance condition, compare label l against currently stored labels.
            Eliminate any currently stored labels which are dominated by l.
            if label l either dominates or is **not** dominated by a currently stored label **then**
                Add label l to the set of label at j, sorted by distance.
                if the number of labels exceeds $maxLabels$ **then**
                    Eliminate the label that has the greatest distance to node j.
                end if
            end if
        end for
    end for
end for

Identifying the path with the shortest distance

Let i be the sink node.
while $prev(i, l)$ is not the source node **do**
    Set $j \leftarrow prev(i, l)$,
    set $l \leftarrow prevLabel(i, l)$.
    Add j to the shortest path.
    Set $i \leftarrow j$.
end while

that affect the implementation. Firstly, line 4 in Algorithm 4.2 is added to iterate over all stored labels at a node. It is important to note that while there is a maximum number of labels that can be stored at each node, there need not be any more than one. Secondly, while the previous node in the shortest path is recorded with the variables $prev(j, l)$, it is also important to record the label propagated from the previous node with the variable $prevLabel(j, l)$. Finally, in addition to updating the shortest distance at each node, Algorithm 4.2 also stores the consumption of each resource m with the variables $consum(m, j, l)$.

The complexity of the shortest path problem is significantly increased with the introduction of multiple labels. Since multiple labels are employed to propagate suboptimal paths, a naïve algorithmic approach is to store every path entering each node. The difficulty with this approach is the enormous number potential paths through the network, each requiring an individual label. This is addressed by a dominance condition to ensure that only Pareto optimal labels are stored. The dominance condition described here is developed from the condition presented by Desrochers and Soumis [30]. Using the crew duty recovery problem (CDRP) presented in Section 3.1.2 as an example, the dominance condition that is implemented throughout this thesis is detailed below.

In the CDRP there are numerous rules that dictate a feasible crew duty, most importantly the number of flying and working hours. Thus, label l at node i stores the cost of the current shortest path to the node, $dist(i, l)$, the number of flying hours, $consum(1, i, l)$, and the total elapsed hours, $consum(2, i, l)$. Using these variables, the following dominance condition is used to identify the Pareto optimal labels.

Definition 4.1.1. *(Dominance Condition)*

Given two labels at node i, ($dist(i, 1)$, $consum(1, i, 1)$, $consum(2, i, 1)$) and ($dist(i, 2)$, $consum(1, i, 2)$, $consum(2, i, 2)$), that are not equal. Label 1 dominates label 2 if

$$dist(i, 1) \leq dist(i, 2),\ consum(1, i, 1) \leq consum(1, i, 2) \text{ and } consum(2, i, 1) \leq consum(2, i, 2).$$

Using Definition 4.1.1, the dominance of any new label arriving at a node is evaluated against all currently stored labels. The comparison between the new label and all currently stored labels has three possible results. Firstly, if the new label dominates any stored label, the dominated labels are removed from the node. Second, if the new label is dominated by any stored label, then the new label is discarded. Finally, if no dominance is established between the new label and the stored labels, then the new label is added to the list of labels stored at that node. For the first and third case, if the number of labels exceeds $maxLabels$, the label that has the largest distance to the node is eliminated. It is important to note that if labels are eliminated by the $maxLabels$ condition, Algorithm 4.2 becomes a heuristic multi-label shortest path algorithm. At the sink node, the label that achieves the lowest cost is selected and the resulting path is the minimum reduced cost path.

The algorithms presented in this section are simple approaches that efficiently solve network flow problems. Given that the algorithms require a topological ordering of the nodes in the associated network graph, this indicates a strong compatibility with airline planning and recovery problems. While Algorithms 4.1 and 4.2 have been presented with an application to the PTAP and CDRP respectively, this does not represent a limitation to their use. It is possible to apply Algorithm 4.1 and 4.2 to any shortest path problem on a network with a topologically ordered set of nodes. Examples of such applications are presented in Chapters 5, 6, 8 and 9.

4.2 Benders' Decomposition

Benders' decomposition is a solution approach originally proposed to solve mixed-integer programming problems [16]. This decomposition approach is effective in reducing the complexity of mixed-integer programs that display a block diagonal constraint matrix. The application of Benders' decomposition constructs multiple independent problem with subsets of variables and constraints. The partitioning of the block diagonal structure results in a set of independent subproblems that are more efficiently solved in isolation than as a whole. Benders' decomposition is employed to partition the planning and evaluation stages of the recoverable robustness framework implemented in Chapters 5 and 6.

The typical form of a problem that Benders' decomposition can be applied to is given by,

$$(\mathrm{P}) \qquad \begin{aligned} \min \quad & c^T x + \sum_{s \in S} \alpha^s f_s^T y^s, && (4.10)\\ \text{s.t.} \quad & Ax = b, && (4.11)\\ & Bx + D_s y^s = d_s \quad \forall s \in S, && (4.12)\\ & x \geq 0, \;\; y^s \geq 0 \quad \forall s \in S. && (4.13) \end{aligned}$$

The problem P is defined by two variable types, x and y^s, which are labelled as the first-stage and second-stage decision variables respectively. Also, there are two distinct sets of constraints, constraints (4.11) which only contain the variables x and constraints (4.12) which provide the link between the first and second-stage decision variables.

The benefit of Benders' decomposition is that a master problem (BMP) is formulated to contain only the x variables and constraints (4.11). The resulting BMP is much less complex than the original problem and hence it is easier to solve. The BMP describes a relaxation of the original problem that is tightened through the addition of cuts, generated from the solutions to a series of independent subproblems. This introduces an iterative process between solving the master problem and using this solution to solve subproblems to identify cuts.

The primal Benders' subproblem for scenario s (PBSP-s) is formulated with the second-stage decision variables and constraints (4.12). This decomposition reduces the complexity of the original problem by fixing the values of the first-stage decision variables in the PBSP-s to the current solution of the BMP, $\bar{x}$. As a result, solving each individual PBSP-s in isolation is much simpler than solving all subproblems simultaneously in the original formulation. The decomposition of problem P into the BMP and the PBSP-s is given by,

BMP

$$\min \quad c^T x + \sum_{s \in S} \alpha^s \varphi^s, \tag{4.14}$$

$$\text{s.t.} \quad Ax = b, \tag{4.15}$$

$$(p_s^i)^T (d_s - Bx) \leq \varphi^s \quad \forall i, s \tag{4.16}$$

$$(w_s^i)^T (d_s - Bx) \leq 0 \quad \forall i, s \tag{4.17}$$

$$x \geq 0. \tag{4.18}$$

PBSP-s

$$\mu^s(\bar{x}) = \min \quad f_s^T y^s, \tag{4.19}$$

$$\text{s.t.} \quad D_s y^s = d_s - B\bar{x}, \tag{4.20}$$

$$y^s \geq 0. \tag{4.21}$$

Since the BMP is formulated without any reference to the second-stage variables, cuts are added to reflect the second-stage decisions made in the PBSP-s. The constraints (4.16) and (4.17) represent the *optimality* and *feasibility* cuts which are generated from the solution to the PBSP-s. Also, the set of variables, φ^s, are included in the BMP to provide a lower bound on the optimal objective value of the PBSP-s, $\mu^s(\bar{x})$, for each scenario s.

The PBSP-s is solved using a fixed solution to the BMP, as such the feasibility of the PBSP-s is not guaranteed. If the PBSP-s is proved to be infeasible, this results in an unbounded dual problem, hence a dual ray, $(w_s^i)^T$, can be identified to construct a *feasibility* cut. The addition of a feasibility cut to the BMP "cuts-off" the solution, $\bar{x}$, that caused the infeasibility in the PBSP-s. If the PBSP-s is feasible, the optimal solution to the subproblem provides an extreme point, $(p_s^i)^T$, in the dual feasible region which is used to construct an *optimality* cut.

The feasibility cuts are added to the BMP whenever the PBSP-s is infeasible, while the optimality cuts are only added if an optimality condition is violated. If $\mu^s(\bar{x}) > \bar{\varphi}^s$, then the addition of a cut generated using the optimal dual solution to the PBSP-s, $(p_s^i)^T$, will tighten the current relaxation of the BMP. If $\mu^s(\bar{x}) \leq \bar{\varphi}^s$, then solving the PBSP-$s$ with the current solution to the BMP achieves the minimum objective value for subproblem s. This indicates that the addition of an optimality cut is not required in the current iteration. If the optimality condition for subproblem s^* is satisfied in the current iteration, this does not indicate that no further cuts are required from this subproblem for the remainder of the solution process. As the relaxation of the BMP is tightened, it is possible that the minimum objective value for subproblem s^* may not be achieved with each solution $\bar{x}$. Hence, the optimality condition must be checked for each subproblem $s \in S$ in every iteration. If the optimality condition for all scenarios is satisfied in the one iteration, i.e. $\mu^s(\bar{x}) \leq \bar{\varphi}^s, \forall s \in S$, then the current solution to the BMP is the optimal solution to the original problem.

Benders' decomposition is described in this section with a focus on stochastic programming problems. The concept of recoverable robustness shares many characteristics with stochastic programs, as such there are solution methods that are common between the two. The recoverable robustness framework solves a planning stage problem which is evaluated against a set of disruption scenarios by solving a recovery problem. Comparing this concept to problem P, the planning stage decision are given by the variables x and the recovery decisions for each disruption scenario s are given by the variables y^s. The implementation of Benders' decomposition for the recoverable robustness framework, including a number of enhancement techniques, is described in Chapters 5 and 6.

4.3 Branch-and-Price

Branch-and-bound is a general solution algorithm that is applied to integer programming problems to find the optimal integer solution. The algorithm is most simply described by two key stages i) the *branching* stage, which enumerates the feasible region by restricting the bounds of integer variables, and ii) the *bounding* stage, which defines upper and lower bounds on the optimal solution. This process forms a sequence of increasingly restrictive subproblems that attempt to more closely represent the convex hull of the feasible region. The use of bounding truncates the search of these subproblems based upon the best found upper and lower bounds.

Branch-and-bound can be used as an exact solution approach, since, in the absence of bounding, the complete enumeration of the feasible region will identify the optimal integer solution.

Identifying the optimal solution to an integer program does not require the complete enumeration of the feasible region if the LP relaxation can be defined. The branch-and-bound algorithm begins by solving the LP relaxation of the original problem to identify a lower bound on the optimal integer solution. The node that represents the LP relaxation of the original problem formulation is called the root node of the branch-and-bound tree. Using the solution to the LP relaxation at the root node, a fractional integer variable, $\bar{z} = a, a \notin \mathbb{Z}$, is selected and subproblems are created by setting the bounds $\bar{z} \leq \lfloor a \rfloor$ and $\bar{z} \geq \lceil a \rceil$ on the left and right branch respectively. This *branching* stage partitions the feasible region into smaller subproblems, eliminating the solution that caused the fractionality of $\bar{z}$. The LP relaxation of each subproblem is then solved and if any fractional integer variables exist, further branching is performed.

The *bounding* stage defines the best upper U and lower L bounds identified in the branch-and-bound tree. At the root node, U is set to the best feasible integer solution, generally identified using a heuristic such as the simple rounding of the fractional variables, and L is set to the objective value of the LP relaxation. Since each branch defines a restriction on the original problem, the lower bound at the current node l, given by the solution to the LP relaxation, must be at least as large as the bounds identified at preceding nodes. Heuristic approaches attempt to find a feasible integer solution at each node to identify the local upper bound u, if u exists and $u < U$ the global upper bound is updated to $U = u$. If the LP relaxation of a subproblem is infeasible, or $l > U$ indicating that the best integer solution at that node will be greater than U, then the node is deleted and that branch is no longer propagated.

Branch-and-bound is an iterative process that continues to partition the feasible region until no further branching is possible or a stopping condition is met. If no further branches are possible, then the optimal integer solution is given by a leaf node in the branch-and-bound tree. While this process guarantees that the optimal integer solution is found, it is very time consuming and more efficient approaches are preferred. Considering the best found upper and lower bounds, it is possible to define a stopping condition to guarantee integer optimality without the complete enumeration of the branch-and-bound tree. If $U = L$, then the integer solution that gives U is the optimal solution for the original problem. Alternatively, an optimality gap, ϵ, can be defined as part of the stopping criteria, terminating the branch-and-bound process if $(U - L)/L < \epsilon$. The optimality gap is a useful approach to terminate the solution process early while achieving near optimal solutions.

In the branch-and-bound process, the solution to the LP relaxation at each node in the tree is used to partition the feasible region, eliminating any fractional solutions. This process becomes more complicated when the LP relaxation is solved by column generation since the optimal LP solution is found using only a subset all possible variables. At the termination of the column generation solution approach, the variables added to solve the LP relaxation to optimality may not include the variables necessary to identify the optimal integer solution. Consequently, branching only on the variables contained in the RMP at the root node may result in suboptimal, or even infeasible, solutions to the integer program. The solution algorithm of *branch-and-price* [15, 90] has been developed to alleviate this difficulty by solving the LP relaxation at each node in the branch-and-bound tree with column generation. This ensures that the variables required in the integer optimal solution are generated during the branch-and-bound process. In the branch-and-price algorithm, the variables may be generated either locally or globally, however the branching decisions from each preceding node must be observed. Column generation is employed in all models developed throughout this thesis, hence branch-and-price is necessary to identify the integer optimal solutions.

4.3.1 Constraint branching

The branch-and-bound algorithm described above introduces the concept of branching on variables, i.e. eliminating fractional solutions by restricting the variable bounds. However, this is not an effective method of branching for branch-and-price. With variable branching, any bound restrictions are potentially made redundant by the generation of new variables with identical decisions at subsequent nodes in the tree. This is particularly evident in airline planning and recovery problems, formulated using flight strings in a set partitioning framework, which are classically modelled using binary variables. For a fractional binary variable $\bar{z}$, the branching decisions involve setting $\bar{z} = 0$ on the left branch and $\bar{z} = 1$ on the right. Since the binary variables in airline problems represent flight strings, setting a fractional variable to zero does not preclude the related flight string from being regenerated in subsequent nodes and entering the problem. As such, the left branch would be redundant and results in an ineffective enumeration of the feasible region. A method used to address this situation is to form branches based upon the information derived from the problem constraints. This branching technique is introduced by Ryan and Foster [76] for transportation problems and is termed *constraint branching*.

In the airline context, Barnhart *et al.* [12] introduces the constraint branching technique of *follow-on* branching. Follow-on branching involves identifying two flight coverage constraints that are satisfied by more than one basic variable. For example, in the PTAP this occurs if there exists two variables with non-zero solutions representing flight strings p, q where $a^r_{ip} = a^r_{iq} = a^r_{jp} = 1, a^r_{jq} = 0$ and $a^r_{ip}y^r_p + a^r_{iq}y^r_q = 1$. The flight pair to branch on is identified by searching over the set of fractional variables to find the most fractional flight connection (i^*, j^*). The two branches are formed by imposing restrictions on the permissible flight strings in the each of the subproblems. For the left branch, all flight strings p, current or generated, must include the flight connection $(i^*, j^*) \in p$ if the flights i^* or j^* are contained in p. On the right branch, all flight strings p must not use the identified connection, $(i^*, j^*) \notin p$. However on the right branch it is not a requirement that the flights i^* and j^* do not exist on the same string for that aircraft, so $(i^*, k) \in p, k \in N$ and $(l, j^*) \in p, l \in N$ where $k \neq j^*$ and $l \neq i^*$ is permissible. This form of branching is demonstrated to produce a more balanced branch-and-bound tree while preserving the structure of the column generation subproblems [15].

4.4 Summary

Three fundamental solution methods were presented in this chapter, column generation, Benders' decomposition and branch-and-price. These three solution methods are employed throughout this thesis, with various enhancement techniques identified in each chapter. Column generation is applied to solve each of the problems developed in this thesis. This solution approach is extended in Chapter 7, with the development of a general framework for column-and-row generation. Both column generation and column-and-row generation are used to solve the integrated airline recovery problems in Chapters 8 and 9. Benders' decomposition is employed to improve the tractability of the recoverable robustness framework employed in Chapters 5 and 6. A number of enhancement techniques for the Benders' decomposition solution process are identified in each application of this approach. In addition, the general framework developed for column-and-row generation presented in Chapter 7 is a direct alternative to Benders' decomposition. Finally, branch-and-price, implemented with the follow-on branching rule presented in Section 4.3.1, is used to solve each problem considered in this thesis to integral optimality. To improve the convergence of the branch-and-price algorithm, a number of problem specific branching rules are developed.

Chapter 5

Recoverable Robust Tail Assignment Problem

The high cost associated with airline recovery has driven an interest in the development of planned solutions that are less susceptible to disruptions. Chapter 2 outlines a number of approaches that have been used to achieve this objective, under the concepts of *robust* and *recoverable* airline planning. These approaches are grouped into two broad categories, proxy and feedback robustness, which describe the model formulation and solution methods. In this chapter the recoverable robust tail assignment problem (RRTAP) is presented as an example of a feedback robust approach for improving recoverability.

As stated in Chapter 1, *recoverability* is a measure of the amount of intervention required by an airline to return operations back to plan following a disruptive event. A characteristic of airline planning problems attempting to improve recoverability is the explicit consideration of the possible recovery actions available to an airline. This is demonstrated in the proxy robust approaches of Ageeva [4], Eggenberg [34] and Kang [51] and the feedback robust approach of Yen and Birge [99]. The proxy robust approaches focus on specific characteristics of the planned solution that are expected to improve recoverability, such as increasing the prevalence of aircraft swapping opportunities. Unfortunately, by the nature of proxy robustness there is no evaluation of the improved recoverability of the planned solution during the optimisation process. By contrast, the feedback robust approach of Yen and Birge [99] is developed as a stochastic program to explicitly evaluate the recoverability of the planned crew pairing solution. However, the evaluation of the crew pairing solution is performed with only a limited set of recovery policies, greatly inflating the impact of the simulated disruptions. Each of these approaches are limited in their ability to improve the recoverability of the planned solution.

The RRTAP is a feedback robust approach that evaluates the recoverability of the planned stage solution against a set of disruption scenarios during the solution process. The planning and recovery tail assignment problems presented in Chapters 2 and 3 respectively, are integrated in the formulation of the RRTAP. The contributions of this chapter are:

1. the use of the full set of recovery policies in the subproblems that evaluate the recoverability of the planned solution,

2. to the best of the author's knowledge, this is the first application of recoverable robustness to airline planning problems.

The use of a full set of recovery policies extends [99] by considering flight cancellations and aircraft rerouting. This feature also extends the proxy robust approaches by providing an evaluation stage during the optimisation process. In addition, the recoverable robustness framework presented by Liebchen *et al.* [57] is applied to timetable recovery for railways, hence the recovery policies are limited to delay and cancellation decisions. The nature of the tail assignment problem permits a larger set of recovery policies than [57], as such the RRTAP provides a contribution to the recoverable robustness framework. Modelling the evaluation stage with a full set of recovery policies accurately simulates the actions taken by the operations control centre following a disruption to produce the highest quality feedback.

The recoverable robustness framework presented by Liebchen *et al.* [57] attempts to identify a planned solution that is recoverable in *limited effort*. The definition of limited effort is problem specific, however it is closely related to the concept of recoverability. For the RRTAP, *limited effort* is defined by two different features of the recovery process i) the number of changes from the planned solution that are required during recovery and ii) the total recovery cost. While the lowest recovery cost is ideal, the number of changes required by an airline to achieve this cost must also be small. By assigning a cost to each change from the planned solution, the weighted sum of the recovery and change costs effectively models the trade-off between these two features in the RRTAP.

The mathematical model of the RRTAP will be presented in Section 5.1. This section will describe the model notation and formulation of the planning and recovery tail assignment problems in the RRTAP. The RRTAP describes a very large and complex mixed integer program, requiring the use of sophisticated techniques to develop an efficient solution approach. The relevant solution methods employed for the RRTAP are detailed in Section 5.2. The recoverability of the RRTAP is evaluated against a set of disruption scenarios and a comparison against a representative proxy robust model is provided in Section 5.3. To provide a greater understanding of the solution approach used for the RRTAP, the effect various enhancement techniques and different parameter values has on the recoverability and runtimes is presented in this section. A summary of the results and the conclusions drawn are presented in Section 5.4. The work presented in this chapter appears in the publication of Froyland, Maher and Wu [40].

5.1 Recoverable Robust Tail Assignment Problem

We define the tail assignment problem as the task of assigning routes to individual aircraft, ensuring that all flights in the schedule are serviced 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 accurately model the recovery problem, the RRTAP employs a full set of recovery options along with estimations of the actual costs. In addition, the planning tail assignment problem is generally modelled as a feasibility problem, which results in a large number of feasible solutions. Therefore, the use of the RRTAP improves the expected recovery costs of the planned solution without any additional planning costs.

The tail assignment problem for the RRTAP has been developed using a flight string formulation introduced by Barnhart *et al.* [12]. The notation presented in Tables 5.1 and 5.2 is used to describe the planning and recovery stages in the RRTAP. The notation that is consistent between the RRTAP and the planning (PTAP) and recovery (RTAP) tail assignment problems is also presented here to provide a clear description of the model. A superscript s denotes the components of the model that relate to the evaluation stage and the disruption scenario s which they belong to, where $s \in S$.

The flight schedule N used in the RRTAP is defined in the same way as for the PTAP in Section 2.1.1. In addition, the definition of a feasible connection presented in Section 2.1.1 is used to construct the set C. In the RRTAP, 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

S	is the set of all scenarios s
R	is the set of all aircraft r
B	is the set of airports b where aircraft flight strings can originate and terminate
P^r	is the set of all flight strings p for aircraft r, the optimal planning variables
P^{sr}	is the set of all flight strings p for aircraft r in scenario s, the recovery variables
N	is the set of all flights j
N^{s-pre}	is the set of flights j that depart before the first disrupted flight in scenario s
N^{s-post}	is the set of flights j that depart after the first disrupted flight in scenario s
C	is the set of all feasible connections in the network, $C = \{(i,j) \| i,j \in N \cup B\}$

Table 5.1: Sets used in the RRTAP.

y_p^{sr} equal 1 when flight string p is operated by aircraft r in the planning and evaluation stages respectively. The cost of using flight route p for aircraft r is given by c_p^r and c_p^{sr} in the planning and evaluation stages respectively. The cost of a flight route in the planning stage is dependent on the length of the connections contained in that route. In the evaluation stage, the cost of a flight route is defined by the amount of delay on flights contained in the string. 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 in the planning and evaluation stages respectively. The RRTAP is solved for a single day schedule, 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 by a flight string represents a column in the constraint matrix.

As explained in Section 2.1.1, the tail assignment problem is the task of assigning flight routes or strings to each 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 in the planning and evaluation stages. We generate individual strings for each aircraft r, and the strings generated for the planning and evaluation stages are contained in the sets P^r and P^{sr} respectively. As a result we treat

y_p^r	= 1 if aircraft r uses flight string p, 0 otherwise
c_p^r	= the cost of aircraft r using flight string p
a_{jp}	= 1 if flight j is 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 flight string p in scenario s, 0 otherwise
c_p^{sr}	= the cost of aircraft r using string p in scenario s, this includes the cost of any delayed flights on that string
a_{jp}^s	= 1 if flight j is in string p in scenario s, 0 otherwise
o_{bp}^s	= 1 if string p terminates at airport b in scenario s, 0 otherwise
z_j^s	= 1 if the flight j is cancelled in scenario s, 0 otherwise
d_j	= the cost of cancelling flight j
$\epsilon_{jr}^{s+}, \epsilon_{jr}^{s-}$	$\begin{cases} \epsilon_{jr}^{s+} = 1, \epsilon_{jr}^{s-} = 0 & \text{if flight } j \text{ is assigned to aircraft } r \text{ for the planning stage but not for recovery in scenario } s \\ \epsilon_{jr}^{s+} = 0, \epsilon_{jr}^{s-} = 1 & \text{if flight } j \text{ is assigned to aircraft } r \text{ for for recovery in scenario } s \text{ but not for the planning stage} \\ \epsilon_{jr}^{s+} = 0, \epsilon_{jr}^{s-} = 0 & \text{if flight } j \text{ is not assigned to aircraft } r \text{ in both the planning stage and recovery in scenario } s \end{cases}$
g^{SW}	weight applied to ϵ_{jr}^{s-} in the objective function, the swap cost
w^s	weight for each scenario in the objective function
M_b	is the minimum number of aircraft required to start the following days flying from base b

Table 5.2: Variables used in the RRTAP.

the sets P^r and P^{sr} for all aircraft r as disjoint.

In the formulation of the recovery tail assignment problem we implement the recovery techniques of flight delays and cancellations while also allowing aircraft rerouting. The technique of modelling flight delays with flight copies is presented in Section 3.1.1, which is implemented for the RRTAP. Flight cancellations are included in the model through the additional variables z_j^s, that equal 1 if flight j is cancelled in scenario s, contributing a cost of d_j to the objective. The addition of the cancellation variables allows the decision of an aircraft either operating or cancelling a flight in the recovery problem.

Since 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-pre} and N^{s-post}, $N = N^{s-pre} \cup N^{s-post}$. The sets N^{s-pre} and N^{s-post} include all of the flights that depart before and after the first disrupted flight in scenario s respectively.

A key feature of the recoverable robustness framework is the objective to find a planned solution that is recoverable in *limited effort*. In the RRTAP, *limited effort* is defined in two parts, minimising the number of deviations from the planned solution during recovery and the lowest cost recovery solution. To reduce the number of deviations, any difference between the planned and recovered flight routes assigned to an individual aircraft is penalised using the variables ϵ_{jr}^{s+} and ϵ_{jr}^{s-}. In the objective function, the swap cost g^{SW} 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 second part of the *limited effort* definition in the RRTAP is the attempt to find a planned solution with the lowest expected recovery cost. As stated at the start of this section, we will be using actual costs for the delay and cancellation of flights in the recovery scenarios.

The objective of minimal deviation is similar to that presented by Thengvall *et al.* [87]. 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 *et al.* [87] where the deviation from the planned solution is restricted through arc constraints in the column generation subproblem. The objective of Thengvall *et al.* [87] is to maintain the use of the planned connections in recovery, whereas our model attempts to maintain the same flights for each aircraft.

The recoverable robust tail assignment problem is formulated to simultaneously solve the planning and recovery problems in the one model. We describe the RRTAP as follows,

$$
\text{(RRTAP)}\quad
\begin{aligned}
\min \quad & \sum_{r\in R}\sum_{p\in P^r} c_p^r y_p^r + \sum_{s\in S} w^s \left\{ \sum_{r\in R}\sum_{p\in P^{sr}} c_p^{sr} y_p^{sr} + \sum_{j\in N} d_j z_j^s + \sum_{r\in R}\sum_{j\in N} g^{SW} \epsilon_{jr}^{s-} \right\}, && (5.1)\\
\text{s.t.} \quad & \sum_{r\in R}\sum_{p\in P^r} a_{jp} y_p^r = 1 \quad \forall j \in N, && (5.2)\\
& \sum_{p\in P^r} y_p^r \le 1 \quad \forall r \in R, && (5.3)\\
& \sum_{r\in R}\sum_{p\in P^r} o_{bp} y_p^r \ge M_b \quad \forall b \in B, && (5.4)\\
& \sum_{r\in R}\sum_{p\in P^{sr}} a_{jp}^s y_p^{sr} + z_j^s = 1 \quad \forall s \in S, \forall j \in N, && (5.5)\\
& \sum_{p\in P^{sr}} y_p^{sr} \le 1 \quad \forall s \in S, \forall r \in R, && (5.6)\\
& \sum_{r\in R}\sum_{p\in P^{sr}} o_{bp}^s y_p^{sr} \ge M_b \quad \forall s \in S, \forall b \in B, && (5.7)\\
& \sum_{p\in P^r} a_{jp} y_p^r - \sum_{p\in P^{sr}} a_{jp}^s y_p^{sr} = 0 \quad \forall s \in S, \forall r \in R, \forall j \in N^{s-pre}, && (5.8)\\
& \sum_{p\in P^r} a_{jp} y_p^r - \sum_{p\in P^{sr}} a_{jp}^s y_p^{sr} = \epsilon_{jr}^{s+} - \epsilon_{jr}^{s-} \quad \forall s \in S, \forall r \in R, \forall j \in N^{s-post}, && (5.9)\\
& y_p^r \in \{0,1\} \quad \forall r \in R, \forall p \in P^r, && (5.10)\\
& y_p^{sr} \in \{0,1\} \quad \forall s \in S, \forall r \in R, \forall p \in P^{sr}, && (5.11)\\
& z_j^s \in \{0,1\} \quad \forall s \in S, \forall j \in N, && (5.12)\\
& \epsilon_{jr}^{s+} \ge 0, \epsilon_{jr}^{s-} \ge 0 \quad \forall s \in S, \forall r \in R, \forall j \in N. && (5.13)
\end{aligned}
$$

The planning tail assignment problem is described by constraints (5.2)-(5.4) and (5.10). Similarly the recovery tail assignment problem is described by constraints (5.5)-(5.7) and (5.11)-(5.12) for each scenario s. The constraints (5.8)-(5.9) are used to track any deviation between the planning and recovery tail assignment problem variables.

The objective function (5.1) minimises the cost of the planning tail assignment and the expected cost of recovery from all scenarios, weighted by w^s, with a penalty for each flight change, g^{SW}. Flight coverage in the planning stage is enforced through constraints (5.2) and in the evaluation stage through constraints (5.5) with an additional variable, z_j^s, to allow for flight cancellations. The restriction on the number of available aircraft is described in constraints (5.3) and (5.6) for the planning and evaluation stages. Also, constraints (5.4) and (5.7) ensure the required number of aircraft are positioned at each airport at the end of the day to begin the next days flying in the planning and evaluation stages. The set of constraints (5.8) are the non-anticipativity constraints ensuring that each aircraft r is assigned to the same flights in both the planning and evaluation stages up to the first disrupted flight for each scenario s. Since 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, constraints (5.9) 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 only include the variable ϵ_{jr}^{s-}, which represents whether flight j is added to the recovered flight route of aircraft r. Since the RRTAP is a minimisation problem, the optimal solution requires ϵ_{jr}^{s+} and ϵ_{jr}^{s-} to be tightly constrained at the lower bound, which is defined by constraints (5.9) or (5.13). Now the lower bound of (5.13) is dominated by (5.9) when the left hand side is greater than zero, which is at most 1. Therefore, the values of ϵ_{jr}^{s+} and ϵ_{jr}^{s-} will be at most 1 in the optimal solution of the RRTAP.

5.2 Solution Methodology

The RRTAP is a large scale optimisation problem that simultaneously solves the planning and recovery tail assignment problems. 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 [27, 63, 69].

Since the RRTAP is similar in structure to a stochastic program, when decomposed by Benders' decomposition, each of the recovery scenarios form an individual subproblem. This technique moves the difficult constraints, equations (5.8)-(5.9), 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 since 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 defining individual 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 [60], the independent Magnanti-Wong method [68] and local branching [71]; here we employ the Magnanti-Wong method [60]. 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 various enhancement techniques developed.

5.2.1 Benders' decomposition

The RRTAP displays the necessary characteristics to apply Benders' decomposition. The general form of such problems is presented by P in Section 4.2, which contain first and second-stage variables that are linked through a set of *complicating* constraints. Comparing the RRTAP to P, the variables y_p^r are equivalent to variables x, and the scenario variables y_p^{sr} can be likened to the variables y^s. As such, the general application of Benders' decomposition provided in Section 4.2 can be employed for the RRTAP. This section will discuss in detail the implementation of the Benders' decomposition solution process using the notation presented in Table 5.3.

The decomposition for this problem is clear given the distinct separation of variables, $y_p^{sr}, z_j^s, \epsilon_{jr}^{s+}$ and ϵ_{jr}^{s-}, between each of the recovery scenarios, $s \in S$. The primal Benders'

Φ	is the objective function value of the Benders' master problem (BMP)
φ^s	the decision variable added to the BMP. This variable is the lower bound on the optimal objective value for the PBSP-s, $\forall s \in S$
$\bar{\mathbf{y}}$	is the optimal solution for the planning variables in the BMP for the current iteration
$\mu^s(\bar{\mathbf{y}})$	is the objective function value for the PBSP-s for a fixed planning stage solution $\bar{\mathbf{y}}$, $\forall s \in S$
Ω^s	the set of all Benders' optimality cuts ω for scenario s added to the BMP

Table 5.3: Additional notation for the Benders' decomposition model.

subproblems (PBSP-s) created for each scenario s include these variables and the constraints (5.5)-(5.9). Only of the planning variables, $y_p^r, \forall r \in R, \forall p \in P^r$ and the constraints (5.2)-(5.4) are included in the Benders' decomposition master problem (BMP).

The solution to the BMP defines the best possible planning solution given the current realised evaluation information, which is given by $\bar{\mathbf{y}} = \{\bar{y}_p^r, \forall r \in R, \forall p \in P^r | \bar{y}_p^r = 1\}$. 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,

$$
\text{(PBSP-}s\text{)}\quad
\begin{aligned}
\mu^s(\bar{\mathbf{y}}) = \min \quad & \sum_{r\in R}\sum_{p\in P^{sr}} c_p^{sr} y_p^{sr} + \sum_{j\in N} d_j z_j^s + \sum_{r\in R}\sum_{j\in N} g^{SW} \epsilon_{jr}^{s-}, && (5.14)\\
\text{s.t.} \quad & \sum_{r\in R}\sum_{p\in P^{sr}} a_{jp}^s y_p^{sr} + z_j^s = 1 \quad \forall j \in N, && (5.15)\\
& \sum_{p\in P^{sr}} y_p^{sr} \le 1 \quad \forall r \in R, && (5.16)\\
& \sum_{r\in R}\sum_{p\in P^{sr}} o_{bp}^s y_p^{sr} \ge M_b \quad \forall b \in B, && (5.17)\\
& \sum_{p\in P^{sr}} a_{jp}^s y_p^{sr} = \sum_{p\in P^r} a_{jp} \bar{y}_p^r \quad \forall r \in R, \forall j \in N^{s-pre}, && (5.18)\\
& \sum_{p\in P^{sr}} a_{jp}^s y_p^{sr} + \epsilon_{jr}^{s+} - \epsilon_{jr}^{s-} = \sum_{p\in P^r} a_{jp} \bar{y}_p^r \quad \forall r \in R, \forall j \in N^{s-post}, && (5.19)\\
& y_p^{sr} \ge 0 \quad \forall r \in R, \forall p \in P^{sr}, && (5.20)\\
& z_j^s \ge 0 \quad \forall j \in N, && (5.21)\\
& \epsilon_{jr}^{s+} \ge 0, \epsilon_{jr}^{s-} \ge 0 \quad \forall r \in R, \forall j \in N. && (5.22)
\end{aligned}
$$

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, $\bar{\mathbf{y}}$. The initial set of strings is constructed by setting $a_{jp'}\bar{y}_{p'}^r = a_{jp'}^s y_{p'}^{sr}, \forall j \in N^{s-pre}, \forall r \in R, \forall p' \in P^r$. This satisfies the cover constraints (5.15), since we are able to set $z_j^s = 1, \forall j \in N^{s-post}$, and the non-anticipativity constraints (5.18).

We define the dual variables as $\boldsymbol{\beta}^s = \{\beta_j^s | \forall j \in N\}$, $\boldsymbol{\gamma}^s = \{\gamma^{sr} | \forall r \in R\}$, $\boldsymbol{\lambda}^s = \{\lambda_b^s | \forall b \in B\}$, and $\boldsymbol{\delta}^s = \{\delta_j^{sr} | \forall r \in R, \forall j \in N\}$ for the constraints (5.15), (5.16), (5.17), and (5.18)-(5.19) respectively. For each scenario s, after solving PBSP-s a Benders' cut is generated from the dual solutions of (5.15)-(5.19). The resulting Benders' optimality cut generated from a single iteration of the Benders' decomposition algorithm is defined as,

$$\varphi^s \geq \sum_{j \in N} \beta_j^s + \sum_{r \in R} \gamma^{sr} + \sum_{b \in B} \lambda_b^s M_b + \sum_{r \in R} \sum_{j \in N} \sum_{p \in P^r} \delta_j^{sr} a_{jp} y_p^r. \tag{5.23}$$

For a fixed $\bar{\mathbf{y}}$, the right hand side of the Benders' optimality cut, equation (5.23), is the objective function value for the dual of PBSP-s. The dual solutions of (5.15)-(5.19) 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 (5.23), 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 by the added cuts. In the solution process of the Benders' decomposition scheme we introduce the set Ω^s, which contains all Benders' cuts ω for 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 given by,

$$\Phi = \min \quad \sum_{r \in R} \sum_{p \in P^r} c_p^r y_p^r + \sum_{s \in S} w^s \varphi^s, \tag{5.24}$$

(BMP)

$$\text{s.t.} \quad \sum_{r \in R} \sum_{p \in P^r} a_{jp} y_p^r = 1 \quad \forall j \in N, \tag{5.25}$$

$$\sum_{p \in P^r} y_p^r \leq 1 \quad \forall r \in R, \tag{5.26}$$

$$\sum_{r \in R} \sum_{p \in P^r} o_{bp} y_p^r \geq M_b \quad \forall b \in B, \tag{5.27}$$

$$\varphi^s - \sum_{r \in R} \sum_{j \in N} \sum_{p \in P^r} \delta_j^{\omega sr} a_{jp} y_p^r \geq \sum_{j \in N} \beta_j^{\omega s} + \sum_{r \in R} \gamma^{\omega sr} + \sum_{b \in B} \lambda_b^{\omega s} M_b \quad \forall s \in S, \forall \omega \in \Omega^s, \tag{5.28}$$

$$y_p^r \in \mathbb{Z}^+ \quad \forall r \in R, \forall p \in P^r, \tag{5.29}$$

$$\varphi^s \geq 0 \quad \forall s \in S. \tag{5.30}$$

The Benders' decomposition solution process is performed by solving the BMP and then with the optimal planning solution, $\bar{\mathbf{y}}$, checking PBSP-s for each scenario s for any improvement cuts. For a given iteration n, the lower bound of PBSP-s for each scenario s is provided by the value of φ_n^s in the solution to BMP and the upper bound is given by $\mu^s(\bar{\mathbf{y}}_n)$. In Section 4.2, the condition for adding a cut from scenario s in iteration n is given by $\mu(\bar{\mathbf{y}}_n) > \varphi_n^s$, which is a very strict stopping condition. To relax this condition, the addition of cuts to the BMP from scenario s in iteration n is given by the gap between the upper and lower bounds of the PBSP-s, $\mu(\bar{\mathbf{y}}_n)$ and φ_n^s respectively, relative to the Benders' master problem objective value, Φ_n. This optimal subproblem criteria is similar to the condition proposed in Papadakos [69] as the stopping criteria. In iteration n of the Benders' decomposition solution process, a cut will be added if the following condition is violated,

$$\frac{\mu^s(\bar{\mathbf{y}}_n) - \varphi_n^s}{\Phi_n} \leq \varepsilon \quad \forall s \in S, \tag{5.31}$$

where ε is the tolerance that we have used in our model, set at $\varepsilon = 10^{-4}$. The solution to the Benders' master problem is the optimal solution to the original problem when no improvement can be made with the addition of cuts. This is equivalent to condition (5.31) being satisfied for all $s \in S$.

The Magnanti-Wong method

In each iteration of the solution process the generated cuts provide an incremental improvement to the master problem. The efficiency of the solution process is highly dependent on the quality of these cuts. In the PBSP-s it is common for a degenerate primal solution to be found, indicating 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 [60]. 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

$$\begin{aligned} &\sum_{j\in N} \beta_{1j}^s + \sum_{r\in R} \gamma_1^{sr} + \sum_{b\in B} \lambda_{1b}^s M_b + \sum_{r\in R}\sum_{j\in N}\sum_{p\in P^r} \delta_{1j}^{sr} a_{jp} y_p^r \\ &\qquad \geq \sum_{j\in N} \beta_{2j}^s + \sum_{r\in R} \gamma_2^{sr} + \sum_{b\in B} \lambda_{2b}^s M_b + \sum_{r\in R}\sum_{j\in N}\sum_{p\in P^r} \delta_{2j}^{sr} a_{jp} y_p^r, \end{aligned} \tag{5.32}$$

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 which 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 (5.25) - (5.30), $\mathbf{y}^0 \in ri(\mathbf{y}^{LP})$; the method by which this point is obtained is explained later in this section. Since the core point is selected to be within $ri(\mathbf{y}^{LP})$, by satisfying condition (5.32) 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,

$$\text{(DMWAP-}s\text{)}\quad \begin{aligned} \max \quad & \sum_{j\in N}\beta_j^s + \sum_{r\in R}\gamma^{sr} + \sum_{b\in B}\lambda_b^s M_b + \sum_{r\in R}\sum_{j\in N}\sum_{p\in P^r}\delta_j^{sr} a_{jp} y_p^{0r}, && (5.33)\\ \text{s.t.} \quad & \sum_{j\in N}\beta_j^s + \sum_{r\in R}\gamma^{sr} + \sum_{b\in B}\lambda_b^s M_b + \sum_{r\in R}\sum_{j\in N}\sum_{p\in P^r}\delta_j^{sr} a_{jp} \bar{y}_p^{r} = \mu^s(\bar{\mathbf{y}}), && (5.34)\\ & (\boldsymbol{\beta}^s, \boldsymbol{\gamma}^s, \boldsymbol{\lambda}^s, \boldsymbol{\delta}^s) \in \Delta^s, && (5.35) \end{aligned}$$

where Δ^s is the dual feasible region of the PBSP-s and its objective function value is given by $\mu^s(\bar{\mathbf{y}})$.

The dual Magnanti-Wong auxiliary optimisation problem, DMWAP-s, is identical to the dual problem of PBSP-s with the addition of constraint (5.34) and a change in the objective function. The primal form of DMWAP-s (PMWAP-s), can be derived from the PBSP-s by i) including a primal variable corresponding to the additional dual constraint (5.34); and ii) setting the right hand side of the comparison constraints (5.18)-(5.19) to the value of the core point $\mathbf{y}^0$. As such, the implementation of the PMWAP-s does not require much additional development and the computational time is comparable to the PBSP-s. In the Benders' decomposition solution process 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, is guaranteed to find the dual solution that is Pareto optimal for the chosen core point as defined by the dominance condition (5.32). 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 $\mathbf{y}^0 \in ri(\mathbf{y}^{LP})$. Mercier *et al.* [63] state that using a core point $\mathbf{y}^0 \notin ri(\mathbf{y}^{LP})$ does not preclude PMWAP-s from finding a valid Benders' cut. However, the further that $\mathbf{y}^0$ is from $ri(\mathbf{y}^{LP})$ 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 $ri(\mathbf{y}^{LP})$ 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

(5.32). 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 $ri(\mathbf{y}^{LP})$ 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 the best possible Benders' cut to improve the BMP solution.

Papadakos [68] developed various enhancements for the Magnanti-Wong method, which we will discuss in Section 5.2.3, 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 *et al.* [63]. However, this particular scheme is not useful for our problem given the complexity of the comparison constraints (5.18)-(5.19) and generally causes the PMWAP-s to be infeasible. Another scheme, presented by Papadakos [68], is to set the core point to the initial solution of the BMP, $\mathbf{y}^0 \rightarrow \bar{\mathbf{y}}_0$, then after each iteration n of the master problem update the core point by $\mathbf{y}^0 \rightarrow \frac{1}{2}\mathbf{y}^0 + \frac{1}{2}\bar{\mathbf{y}}_n$. The benefit of this particular scheme is that at each iteration n of the Benders' decomposition solution process the core point is moving closer towards the $ri(\mathbf{y}^{LP})$. So even if the initial core point is not within the relative interior, $\mathbf{y}^0 \notin ri(\mathbf{y}^{LP})$, it is possible to incrementally improve the potential strength of the Benders' cuts with more iterations of the Benders' decomposition solution process. Through experimental experience this latter scheme from [68] has been shown to be useful in our problem to find a representative core point. We demonstrate in Section 5.3.3 that the Magnanti-Wong method greatly improves the efficiency of the Benders' decomposition solution process.

5.2.2 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. 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 solution process improves the master problem with the addition of negative reduced cost columns. These columns are generated from a column generation subproblem using the current LP dual solutions to the PBSP-s. In Section 5.2.1 we defined the dual variables for each scenario subproblem, s, as $\boldsymbol{\beta}^{\boldsymbol{s}} = \{\beta^s_j | \forall j \in N\}$, $\boldsymbol{\gamma}^{\boldsymbol{s}} =$

$\{\gamma^{sr}|\forall r \in R\}$, $\boldsymbol{\lambda^s} = \{\lambda_b^s|\forall b \in B\}$, and $\boldsymbol{\delta^s} = \{\delta_j^{sr}|\forall r \in R, \forall j \in N\}$ for the constraints (5.15), (5.16), (5.17), and (5.18)-(5.19) respectively.

The PBSP-s describes a recovery tail assignment problem, as such the definitions of the flight schedule and connection network presented in Section 3.1.1 for the RTAP are used. In Section 3.1.1, the technique of flight copies is used to implement delays, greatly affecting the construction of the flight schedule and connection networks. Since flight copies are also used to model flight delays in the PBSP-s, a brief review of the implementation will be provided. 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 $U_j = \{0\}, \forall j \in N^{s-pre}$ indicating the set of copies only contains the copy representing the original scheduled departure. Also, for all flights $j \in N^{s-post}$, U_j contains $v = 0$ and at least one other copy representing some delay on flight j. So, we define the set $\hat{N}^s = \{j_v|j \in N^{s-pre} \cup N^{s-post}, v \in U_j\}$ as all nodes in the connection network for the PBSP-s. The rules presented in Section 3.1.1 that describe a feasible connection using the flight-copy notation are used to define the set of connections for the PBSP-s given by $\hat{C}^s = \{(i_u, j_v)|i_u, j_v \in \hat{N}^s \cup B\}$.

The column generation subproblem for the PBSP-s is formulated as a network flow problem for each aircraft r. 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,

$$\hat{c}_p^{sr} = \min \sum_{(i_u,j_v)\in\hat{C}^s} c_{i_u j_v}^{sr} x_{i_u j_v}^{sr} - \sum_{j_v \in \hat{N}^s} \sum_{\substack{i_u \in \hat{N}^s \\ |(i_u,j_v)\in\hat{C}^s}} (\beta_j^s + \delta_j^{sr}) x_{i_u j_v}^{sr}$$

$$- \sum_{b\in B} \sum_{\substack{i_u \in \hat{N}^s \\ |(i_u,b)\in\hat{C}^s}} \lambda_b^s x_{i_u b}^{sr} - \gamma^{sr}, \tag{5.36}$$

$$\text{s.t.} \quad \sum_{\substack{i_u \in \hat{N}^s \\ |(i_u,j_v)\in\hat{C}^s}} x_{i_u j_v}^{sr} - \sum_{\substack{k_w \in \hat{N}^s \\ |(j_v,k_w)\in\hat{C}^s}} x_{j_v k_w}^{sr} = 0 \quad \forall j_v \in \hat{N}^s, \tag{5.37}$$

$$\sum_{v\in U_j} \sum_{\substack{i_u \in \hat{N}^s \\ |(i_u,j_v)\in\hat{C}^s}} x_{i_u j_v}^{sr} \leq 1 \quad \forall j \in N^{s-pre} \cup N^{s-post}, \tag{5.38}$$

$$\sum_{\substack{i_u \in \hat{N}^s \\ |(b_r, i_u) \in \hat{C}^s}} x^{sr}_{b_r i_u} = 1, \tag{5.39}$$

$$\sum_{b \in B} \sum_{\substack{i_u \in \hat{N}^s \\ |(i_u, b) \in \hat{C}^s}} x^{sr}_{i_u b} = 1, \tag{5.40}$$

$$x^{sr}_{i_u j_v} \in \{0, 1\}, \quad \forall (i_u, j_v) \in \hat{C}^s. \tag{5.41}$$

The problem described by (5.36)-(5.41) is a network flow problem with one source and multiple sink nodes. Constraints (5.37) describe the flow balance and constraints (5.38) 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, which is achieved through the coverage constraints (5.38). Given that (5.36)-(5.41) is formulated as a network flow problem, there are a number of classical algorithms available to efficiently solve this problem. Since the connection network is an acyclic directed graph, Algorithm 4.1 is implemented to solve the column generation subproblem for the PBSP-s.

It must be noted that there is a subtle difference in the flight schedule used between the formulations of the PBSP-s and the RTAP. In the RTAP, the flight schedule is defined using a recovery window, fixing the activities preceding and succeeding this period. The concept of carry-in and carry-out activities is introduced to ensure that the recovered solution continues operations following the recovery period. The recovery window is not employed in the solution of the PBSP-s, and as such all flights in the schedule are considered. A set of non-anticipativity constraints are introduced to fix the activities preceding the disruption, and recovery actions are permitted on all flights that depart after the disruption until the end of the day. This difference in the flight schedule used does not greatly affect the solution approaches, the only effect is seen in construction of the connection networks.

There is little difference in the column generation subproblem for the BMP and the PBSP-s, since both are formulated as network flow problems. The column generation subproblem of the BMP is formulated with the constraints (5.37)-(5.41), however a different connection network is required. Since 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 above 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 $\boldsymbol{u} = \{u_j | \forall j \in N\}$, $\boldsymbol{v} = \{v^r | \forall r \in R\}$, $\boldsymbol{w} = \{w_b | \forall b \in B\}$, and $\boldsymbol{\rho} = \{\rho^{s\omega} | \forall s \in S, \forall \omega \in \Omega^s\}$ for the constraints (5.25)-(5.28) respectively. As a result the objective function for the column generation subproblem of the BMP is given by,

$$\begin{aligned} \bar{c}_p^r = \sum_{(i,j)\in C} c_{ij}^r x_{ij}^r - \sum_{j\in N} \sum_{\substack{i\in N \\ |(i,j)\in C}} u_j x_{ij}^r - v^r - \sum_{b\in B} \sum_{\substack{i\in N \\ |(i,b)\in C}} w_b x_{ib}^r \\ - \sum_{s\in S} \sum_{\omega\in\Omega^s} \left\{ \sum_{j\in N} \sum_{\substack{i\in N \\ |(i,j)\in C}} \delta_j^{\omega s r} x_{ij}^r \right\} \rho^{s\omega}, \quad \forall r \in R. \end{aligned} \tag{5.42}$$

Since the structure of the BMP is modified after each iteration through the addition of Benders' cuts, the objective function (5.42) must be updated with the respective additional dual variables. The addition of Benders' cuts only affects connection costs while maintaining the network structure. Similar to the PBSP-s, the connection networks for the BMP is also an acyclic directed graph, therefore it is possible to implement Algorithm 4.1 to efficiently solve the network flow problem.

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 (5.25). For the aircraft assignment constraints (5.26) and the end of day constraints (5.27) only the first n variables are included, where $n = |R|$. To satisfy all of the constraints in the BMP we reintroduce the concept of ferry flights, which is the repositioning of aircraft by flying without passengers. Constraints (5.27) 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. This is consistent with the statement made in Section 3.1.1, since the ferry flights are only included in the BMP as a modelling approach to guarantee an initial feasible 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 since 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 Section 5.2.1.

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 since 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 (5.27) 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 Section 5.2.4.

5.2.3 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 which is based off the three-phase algorithm developed to solve the integrated crew scheduling and aircraft routing problem with Benders' decomposition [27,63,69]. The two-phase algorithm, described by Algorithm 5.1, is a heuristic that initially solves the linear relaxation of the RRTAP, and re-introduces the integrality requirements to the BMP after the first phase is completed. In Cordeau *et al.* [27], Mercier *et al.* [63] and Papadakos [69] 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 since for all scenarios s, the PBSP-s is always feasible for any solution to the master problem, as explained in Section 5.2.1. Thus, it is only necessary to implement the two-phase algorithm for our model.

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 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 which are added in Phase 2 tighten this upper bound, however in Section 5.3 we demonstrate the

Algorithm 5.1 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, (5.24)-(5.30).
  for all scenarios $s \in S$ **do**
    Solve the PBSP-s using column generation, (5.14)-(5.22).
    if condition (5.31) 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 of type (5.23) to the BMP.
    end if
  end for
until condition (5.31) is satisfied, $\forall s \in S$.

PHASE 2

Re-introduce the integrality requirements for the BMP.
Retain all cuts that have been added in **PHASE 1**.
repeat
  Solve the BMP using column generation, (5.24)-(5.30).
  for all scenarios $s \in S$ **do**
    Solve the PBSP-s using column generation, (5.14)-(5.22).
    if condition (5.31) 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 of type (5.23) to the BMP.
    end if
  end for
until condition (5.31) is satisfied, $\forall s \in S$.

Phase 1 bound is very close to the optimal solution.

It is not necessary to include the Magnanti-Wong method [60] in Algorithm 5.1, however we have found great computational benefit from its use each time the PBSP-s is solved as demonstrated in Section 5.3.3. Another possible technique to improve the computational performance of the Benders' decomposition algorithm is to implement the independent Magnanti-Wong method [68], which is solved to find cuts independent of the subproblem solution. In Papadakos [68], the benefit of using the independent Magnanti-Wong method is to generate initial cuts that provide a good lower bound for the master problem without having to solve the Benders' subproblems. Since 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 [68] and only implement the original Magnanti-Wong method [60].

5.2.4 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. In Section 4.3.1, the concept of constraint branching, developed by Ryan and Foster [76], is introduced. Constraint branching is demonstrated to be very effective within the branch-and-price framework, as such the branching rules developed for the RRTAP are based upon this idea.

The first of the branching rules implemented for the RRTAP is modelled off the follow-on branching presented by Barnhart *et al.* [12]. A description of this rule is presented in Section 4.3.1. This branching rule is implemented in the RRTAP using the same method that is described in Section 4.3.1, so no further details regarding the implementation are required.

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 *et al.* [14] for multi-commodity flow problems. In their problem, the commodity to branch on is determined by the largest amount of flow; in our case each of our commodities have the same amount of flow, hence the method requires some 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 since there is only one origin for each aircraft. 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 the flights in these two strings respectively.

Algorithm 5.2 Aircraft/Arc branching

Let i be the position indicator for the current flight.
Given a pair of flight strings p_1 and p_2 for aircraft r^*,
set i to the position of the first flight in both strings, $i \leftarrow 0$,
set the divergence point d to the starting position, $d \leftarrow 0$, this assumes that the two strings only share the source node.
Initialise arc_1 and arc_2 as empty flight strings.
while $i <$ length of p_1 and $i <$ length of p_2 **do**
    if flight at position i in string $p_1 \neq$ flight at position i in string p_2 **then**
        Set the divergence point $d \leftarrow i$.
        Exit loop.
    end if
    Increment i by 1.
end while
for all flights from position d to the length of p_1 **do**
    append flight to arc_1.
end for
for all flights from position d to the length of p_2 **do**
    append flight to arc_2.
end for
if there exists a connected pair of flights that is found in both arc_1 and arc_2 **then**
    eliminate this pair and all succeeding flights from arc_1 and arc_2.
end if
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 branching is performed by excluding aircraft r^* from using the flight connections contained in arc_1 on the left branch, and arc_2 on the right branch. A detailed description on identifying arc_1 and arc_2 to branch on is presented in Algorithm 5.2.

With the two branching rules implemented there must be a priority assigned to each specifying when they are 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 since 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 Section 5.2.2 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 which is used by two different aircraft. We define $V_j^r = \{v \in V | r^v = r, s^v = j\}$ 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 $V_j^{r_1} \neq \emptyset$. 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.

5.3 Computational Experiments

To evaluate the effectiveness of using the recoverable robustness framework 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 [45] in defining the cost of a flight string c_p^r. The new proxy robust model consisting of the objective function $\sum_{r \in R} \sum_{p \in P^r} c_p^r y_p^r$ and the constraints (5.2)-(5.4) and (5.10) is solved to find an optimal tail assignment. Since 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

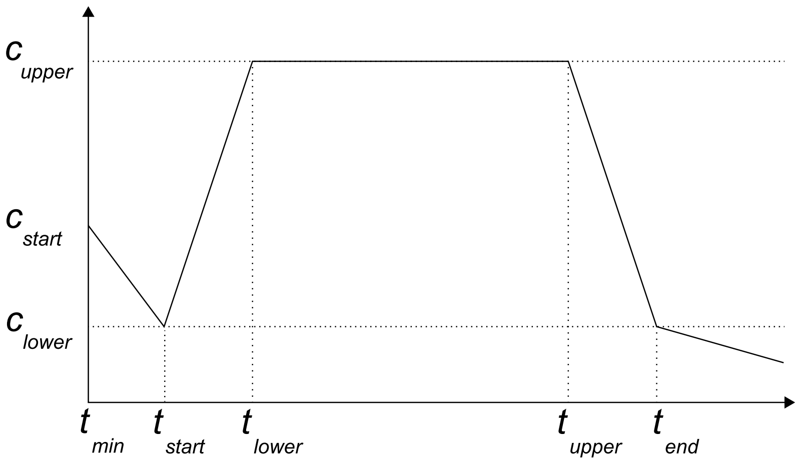

Figure 5.1: Connection cost function presented in Grönkvist [45]. Time parameters (minutes): $t_{min} = 40, t_{start} = 120, t_{lower} = 180, t_{upper} = 300, t_{end} = 360$. Cost Parameters (\$): $c_{start} = 500, c_{lower} = 100, c_{upper} = 5000$.

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 the PhD thesis of Grönkvist [45] 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 robust model in Figure 5.1. 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, $t_{min} \leq t \leq t_{start}$, while ideal in regards to aircraft utilisation, are more prone to propagating delay, and suggests that a compromise ideal connection length of $t = 120$. The medium length connections, $t_{lower} \leq t \leq t_{upper}$, are penalised heavily since they do not provide high enough utilisation and are too short to service extra flights in a recovery situation. The long connections, $t \geq t_{end}$, are also favoured since 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 changes, cancellations and delay minutes in our experiments.

We mention in Section 5.2 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 in i) 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 a potential benefit in applying the recoverable robustness framework to any model.

We implemented this model in C++ and called SCIP 2.0.1 [3] to solve the integer program using CPLEX 12.2 as the linear programming solver.

5.3.1 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 3 major airports, serviced by 10 aircraft. Two different types of disruption scenarios have been implemented, which are airport closures and aircraft grounding, resulting in 102 scenarios. The specifics of the disruption scenarios are presented in Table 5.4.

We estimate the relative probability of each of the above scenarios occurring in a single day and encode these probabilities as the weights, w^s, in equations (5.1) and (5.24). 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 summer, severe storms characterised by high winds and regular lightning

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

Table 5.4: Disruption scenarios used.

strikes generally occur in the afternoon. We estimate that in a season a single airport may experience an afternoon closure approximately 3-4 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 of 300 minutes will occur with a probability of 2% $\times$ 30% = 0.6%.

An aircraft grounding could be attributed to a number of different factors, which include 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 US airlines published by the Bureau of Transportation Statistics [72], in August 2011 approximately 18% of all flights were delayed, and out of all flight delays approximately 26% were caused by factors in the airlines control, so 18% $\times$ 26% = 4.68% of all flights were delayed due to airline factors. We assume that an aircraft grounding causes flight delays due to 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 flight delayed due to airline factors, and this difference occurs since 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 Section 5.1, 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 since 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 *et al.* [26] which 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 multiplied by a lost revenue parameter or *loss rate*. The *loss rate* is an airline specific parameter indicating the expected amount of passenger recapture after a cancellation. A loss rate less than 1 indicates that a percentage of passengers are recaptured by rebooking themselves on another flight provided by the airline, whereas a value greater than 1 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. Since 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. Since 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 are not 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. Since 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 rate. Also, the delay cost per flight given in the EUROCONTROL report [26] is based on a full aircraft. Given that we are assuming a 75%

capacity, the delay cost of an aircraft per minute in our model is \$75.

As mentioned in Section 5.1 we handle flight delays using flight copies in the recovery network. For our experiments we have used a maximum allowable delay of 180 minutes, with a flight copy increment of 30 minutes. Given that the delay increment is 30 minutes, the results will over estimate the true delay costs, since many shorter feasible connections may 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.

5.3.2 Comparison of recoverable robust solutions and Grönkvist solution

The RRTAP attempts to find a planned solution that requires minimal changes during the recovery from disruptions. To represent the difficulty faced by operations controllers to reroute aircraft we use a swap cost parameter g^{SW}. This 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 costs 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 costs in the range $10 \leq g^{SW} \leq 10000$ (10, 500, 1000, 2500, 5000, 10000). A swap cost of 10 illustrates the case where there is virtually no penalty for changes. It is also trivial to implement this parameter with different values for each flight or aircraft in the model.

The BMP of the recoverable robust model is solved using the Grönkvist connection cost function. This allows us to 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 evaluated using the weighted sum of the recovery costs over all scenarios. This is defined as

$$WeightedCost = \sum_{s \in S} w^s \left\{ \sum_{r \in R} \sum_{p \in P^{sr}} c_p^{sr} y_p^{sr} + \sum_{j \in N} d_j z_j^s + \sum_{r \in R} \sum_{j \in N} g^{SW} \epsilon_{jr}^{s-} \right\}, \tag{5.43}$$

which is the second term in the objective function (5.1). We use equation (5.43) to calculate the weighted simulated recovery cost using the solution from the proxy robust model and the solution to the RRTAP at the completion of Phase 1 and 2 of the two-phase algorithm. We compare the weighted recovery costs calculated with different cancellation loss rates and swap costs. 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 which is demonstrated in Table 5.5.

Table 5.5 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 is 14.83%, which occurs with $g^{SW} = 1000$ and a loss rate equal to 0.25. There is an apparent nonlinear relationship between recoverable robustness improvement and the swap costs, with the greatest improvement occurring with swap costs in the range $1000 < g^{SW} < 5000$. In that range and with a loss rate ≥ 0.25, the RRTAP achieves an average improvement of 8.77% over the proxy robust solution, 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. Since the tail assignment problem can be formulated as a feasibility problem, any improvement in the recoverability of the planned solution is made at no extra cost to the airline.

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, attempting to avoid the propagation of delays and create recovery opportunities. To demonstrate the relative performance of alternative connection cost functions against the Grönkvist function,

		Swap Cost				
		500	1000	2500	5000	10000
Loss Rate	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)

Table 5.5: Relative difference between the Grönkvist solution (x) and the recoverable robust solution, Phase 2 (y) and Phase 1 (z), $((y-x)/x\ (\%),\ (z-x)/x\ (\%))$. For conciseness the results using a swap cost of 10 are omitted. Relative difference greater than 5% is highlighted.

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 5.6 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 5.5, including the experiments with a swap cost equal to 10. Each of the functions presented in Table 5.6 construct a tail assignment with different characteristics, for example the linear, quadratic and square root functions favour shorter connections compared to the hyperbola function which favours long connections. While the results presented in Table 5.6 show that the linear and square root functions dominate Grönkvist in most test cases, Grönkvist outperforms the quadratic and hyperbola functions. Table 5.6 also shows that the range of mean relative differences is not particularly large, even over this collection of disparate 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%

Table 5.6: Analysis of connection cost functions. The constructed tail assignment is evaluated against 54 test cases using different penalties and loss rates.

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 [45, 46] we choose to solve RRTAP relative to this function.

The results in Table 5.5 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 4 representative cases; i) Swap Cost = 500, Loss Rate = 0.5; ii) Swap Cost = 1000, Loss Rate = 1.5; iii) Swap Cost = 2500, Loss Rate = 1; and iv) Swap Cost = 2500, Loss Rate = 2.5. Figure 5.2 presents the individual recovery costs in each scenario for the proxy robust, Phase 1 and Phase 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 cost for scenario s is calculated by,

$$IndividualCost(s) = \sum_{r\in R}\sum_{p\in P^{sr}} c_p^{sr} y_p^{sr} + \sum_{j\in N} d_j z_j^s + \sum_{r\in R}\sum_{j\in N} g^{SW}\epsilon_{jr}^{s-}, \tag{5.44}$$

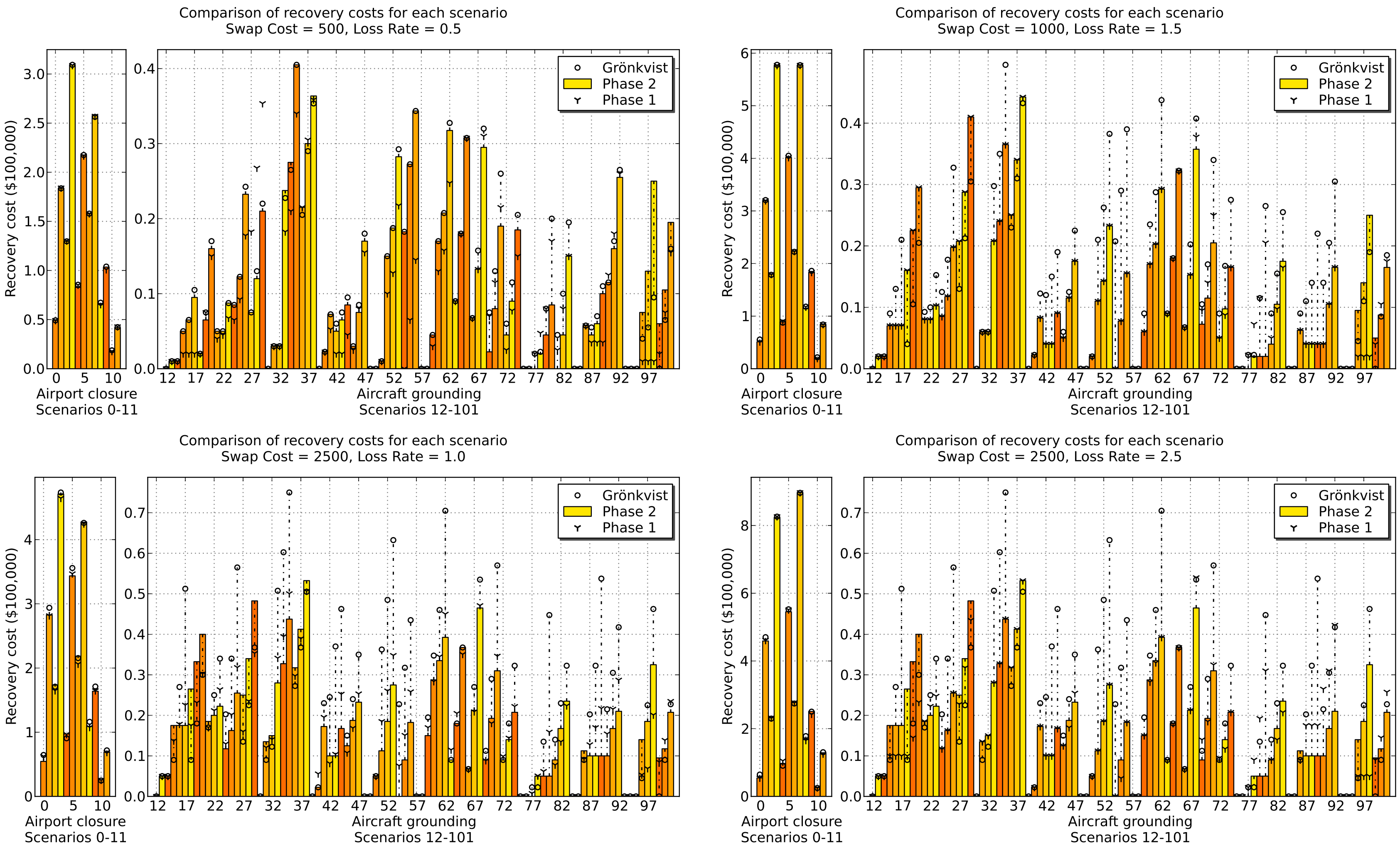

Figure 5.2: Individual recovery costs for all 102 scenarios - comparison between Grönkvist, Phase 1, and Phase 2 results. The scenario numbers are presented in Table 5.4

which is presented for all cases in Figure 5.2. We have selected two types of scenarios for use in this model, airport closure and aircraft grounding, as explained in Table 5.4. An airport closure has a much greater impact on an airlines operations than an aircraft grounding since a large number of flights and aircraft are potentially affected. In Table 5.7 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 5.7 with Figure 5.2 it is clear that the weighted recovery cost is dependent on the number of affected flights. Since 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 5.2. It is difficult to determine the affected flights for the aircraft grounding scenarios since this is dependent on the planned routing, which we are optimising.

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

Table 5.7: Number of flights affected in the airport closure scenarios. Scenarios 0-11.

While 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 scenarios. 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 5.2 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 an important result since we are attempting to improve the recoverability of the tail assignment for a broad range of disruptions. Therefore, if the improvements were restricted only to a few scenarios the efficacy of this technique would be reduced.

The RRTAP is a robust planning model that is closely aligned with stochastic programming. Since 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 (5.1)-(5.13) 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 μ^{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 μ^{max} to be greater than or equal to the recovery solution cost for all scenarios. We define the PRTAP as,

$$\text{(PRTAP)} \quad \begin{aligned} \min \quad & \sum_{r\in R}\sum_{p\in P^r} c_p^r y_p^r + \mu^{max}, && (5.45)\\ \text{s.t.} \quad & \text{Constraints (5.2)-(5.13)}, && (5.46)\\ \mu^{max} \geq & \sum_{r\in R}\sum_{p\in P^{sr}} c_p^{sr} y_p^{sr} + \sum_{j\in N} d_j z_j^s + \sum_{r\in R}\sum_{j\in N} g^{SW}\epsilon_{jr}^{s-} \quad \forall s\in S, && (5.47)\\ & \mu^{max} \geq 0. && (5.48)\end{aligned}$$

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, μ^{max}, to the objective function and the inclusion of additional constraints (5.47) 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 (5.43), and the 90th percentile recovery cost over all scenarios for the RRTAP and the PRTAP is presented in Figure 5.3. Across all parameter sets, the difference in the maximum recovery costs between the two models is within the range of -3.73% to 6.05%, with an average of 1.04% indicating only a small improvement in the PRTAP over the RRTAP. Since there is little variation in the maximum recovery costs, the 90th percentile has been selected for Figure 5.3 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 since 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 cost of 10000. From these results it is clear that in

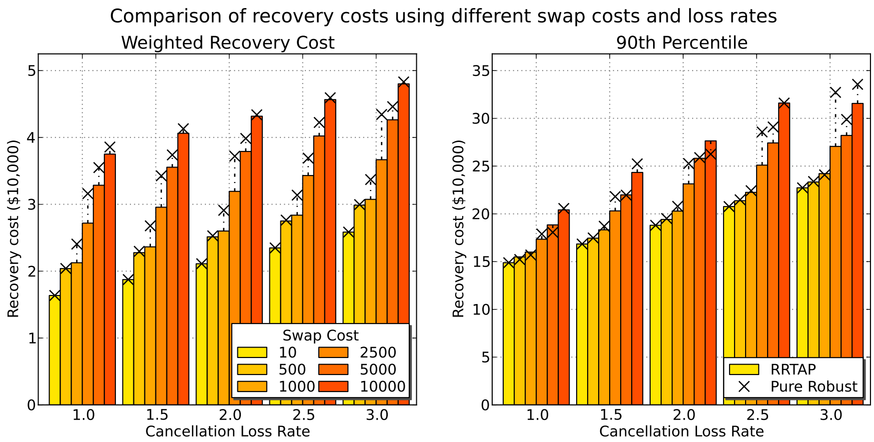

Figure 5.3: Weighted and 90th percentile recovery costs - comparison between the RRTAP and a Pure Robust formulation.

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 provides interesting results when compared to the RRTAP. The experiments for the RRTAP had a maximum runtime of 3 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 7371.18 and 3093.93 seconds for the PRTAP and RRTAP respectively. We provide a greater analysis of the runtime for the RRTAP in Section 5.3.3.

5.3.3 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 Section 5.2 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 to solve large-scale optimisation problems. To further improve the solution runtimes we introduced the Magnanti-Wong method [60], as illustrated in Section 5.2.1, to generate Pareto optimal Benders' cuts. We also implemented a number of branching rules described in Section 5.2.4 for use in Phase 2 of the two-phase algorithm. In Figure 5.4 we present the runtimes for the RRTAP using different algorithmic enhancements on the four cases selected for Figure 5.2. The different results presented in Figure 5.4 are defined as i) including all enhancements presented in Section 5.2; ii) including only the Magnanti-Wong method [60] with the default branching rules in the

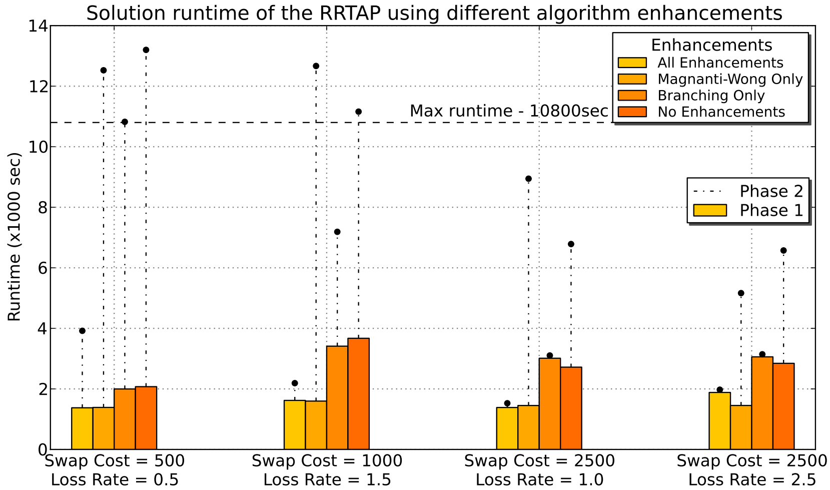

Figure 5.4: Solution runtimes of the RRTAP using different algorithm enhancements. The dots represent the total runtime.

SCIP 2.0.1 distribution; iii) using the branching rules described in Section 5.2.4 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 since the Magnanti-Wong method generates different cuts to the standard Benders' decomposition approach. The addition of Magnanti-Wong cuts generates a different problem by the end of Phase 1, which affects the complexity 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, since 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 5.4 demonstrates that while each enhancement can improve the

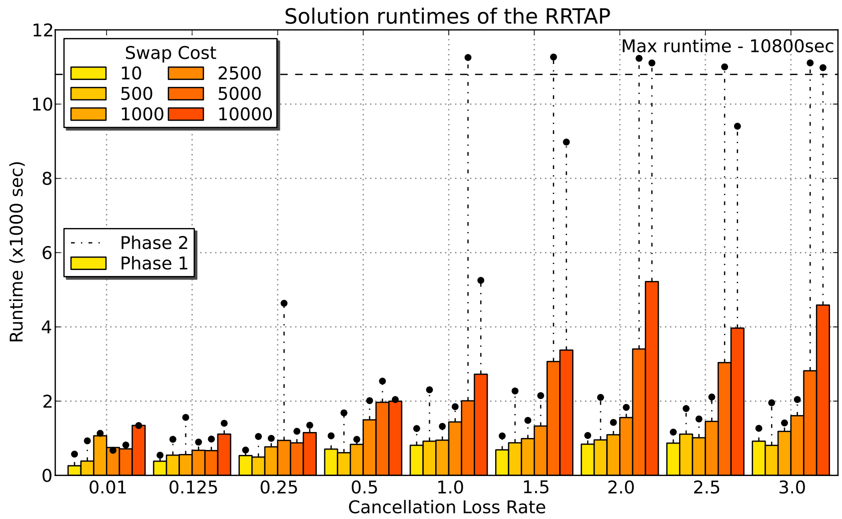

Figure 5.5: Solution runtimes of the RRTAP. The dots represent the total runtime for each parameter set.

runtimes for the algorithm, it is necessary to include all enhancements to achieve the greatest runtime improvement.

While 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 5.5 illustrates the runtimes required to calculate the results presented in Table 5.5 for the RRTAP. In our experiments we limited the runtimes to 3 hours. In the cases where the runtime exceeds 3 hours, the model was terminated after the first run of the Benders' decomposition algorithm completed following the 3 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 5.5. In Table 5.5 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 5.5 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. In order to achieve a solution quickly that is close to optimal, one could simply complete just Phase 1 of the two-phase algorithm.

5.3.4 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 changes, delays and cancellations. Figure 5.6

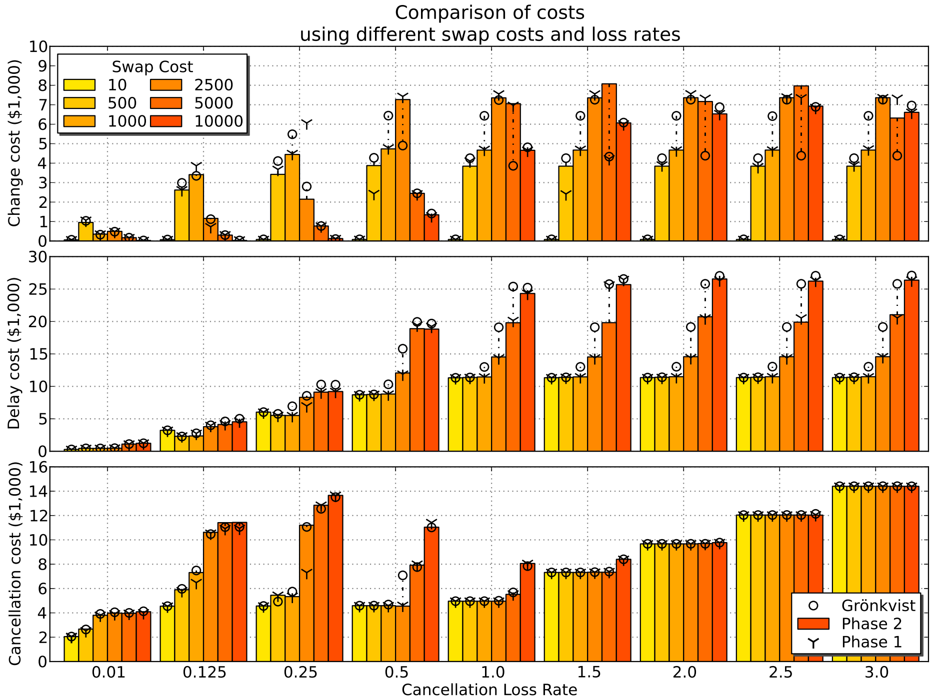

Figure 5.6: Change, cancellation and delay costs - comparison between the Grönkvist, Phase 1 and Phase 2.

presents this breakdown, demonstrating the trade-offs that can occur when setting the model parameters. In Section 5.3.2 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 of $g^{SW} = 5000$, where the number of changes in the proxy robust solution is *lower* than the recoverable robust solution. However, for the same swap cost of $g^{SW} = 5000$, the recoverable robust solution significantly outperforms the proxy robust result for the *delay* costs. For other swap costs, such as $g^{SW} = 1000$ and $g^{SW} = 2500$, the results are quite varied with the improvement being attributed to a decrease in changes and delays respectively. This is attributable to the minimum length of delay for an individual flight and its associated cost. Since the flight delays are discretised to be every 30 minutes from the original departure, the minimum cost of delay is \$75 $\times$ 30 minutes = \$2250. For the proxy robust solution, the tendency is to allow more changes, rather than to delay flights for swap costs less than 2250, and for swap costs greater than 2250 to allow more flight delays than changes.

Thus, using a swap cost of $g^{SW} = 5000$ 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 5.5 it is possible to see that the greatest weighted improvement in the recovery costs occurs when $g^{SW} = 1000$ and $g^{SW} = 2500$. Above we have presented the case where $g^{SW} = 5000$, with the benefit attributable to a decrease in the total *delay* costs. This is combined with an increase in the total number of *changes* performed during the recovery process. The largest weighted recovery cost improvement occurs when $g^{SW} = 2500$ and we see that this benefit is largely due to a reduction in the *delay* costs. While the greatest improvement is in the *delay* costs, there is still an improvement in the *change* costs. As mentioned above, we find that the improvement in both costs is due to the minimum cost of delay being close to the swap costs of $g^{SW} = 2500$. The result of decreased flight delays is particularly important since this directly reduces the number of delay minutes experienced by passengers, which has a real effect on the on-time performance of an airline.

5.3.5 Effect of flight copy increments

In our model we have used discrete flight copies to handle flight delays for the recovery subproblems. As mentioned above we have used a delay copy increment of 30 minutes for each flight copy, with a maximum delay of 180 minutes. In Figure 5.7 we present the weighted recovery costs and the runtime for the RRTAP using different delay increments with a swap cost of $g^{SW} = 2500$ and loss rate of 1.5. As expected, the figure shows that as the delay copy increment decreases, resulting in a larger recovery flight network due to more flight copies, the solution runtime 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 5.7, 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 5.7 we have omitted the breakdown of the recovery costs for brevity, however we will provide a concise description of our findings. By decreasing the delay copy increment we see different results in recovery components of changes, delays and cancellations: the number of delayed

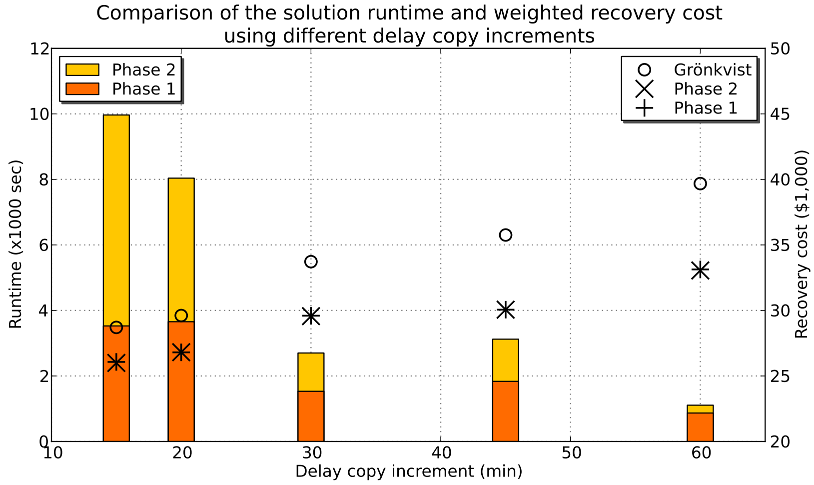

Figure 5.7: Comparison of the solution runtimes and weighted recovery costs using different delay copy increments. The bars and the points represent the solution runtimes and weighted recovery costs respectively. These results have been calculated using a swap cost of 2500 and loss rate of 1.5. Maximum possible delay is 180 minutes.

flights and number of changes increases, while the number of cancellations and delay minutes decreases. Since 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.

5.4 Conclusions

In this chapter we presented a novel recoverable robustness model for the tail assignment problem. We show that the solutions to this model guarantee reduced recovery costs with no increase in the planning costs. Further, the recovery tail assignment is modelled with 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 various 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 and solve individually the planning and recovery tail assignment problems. This structure allows for the use of different recovery algorithms and a wide range of planning algorithms and methods which can be improved upon using the RRTAP.

Through the use of Benders' decomposition and column generation we have demonstrated an efficient solution approach for the RRTAP. The acceleration technique of the Magnanti-Wong method [60] has been applied to the Benders' decomposition formulation, which significantly improves the computational performance. Further, we 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 compared the results of the RRTAP against a proxy robust solution developed using a connection cost function presented by Grönkvist [45]. 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 evaluated the recoverability of a planning tail assignment constructed using this connection cost function. Using this connection cost function in the planning stage of our recoverable robustness algorithm we developed a planned tail assignment with a better recoverability compared to the Grönkvist solution.

The two-phase method was employed to efficiently solve the RRTAP following the application of Benders' decomposition. It was identified 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. This is critical in reducing solution runtimes since a large proportion of the runtimes are spent in Phase 2 adding cuts to improve the lower bound towards the Phase 1 upper bound. Terminating the algorithm at the completion of Phase 1 results in near optimal solutions with very fast computational times.

The results were presented with a range of values for airline specific parameters, namely the swap costs and lost rates. The method used to assign costs to changes, delays and cancellations can have varied effects on the cost benefits from the RRTAP. We demonstrated that due to the value of the minimum delay cost and the swap costs there can be a tendency towards greater changes or delays in the final solution. By presenting a range of values for the parameters we demonstrated that a trade-off between different recovery cost components can be achieved. This allows each airline to value delays, cancellations and changes differently depending on their individual business practices.

A common method to reduce the complexity of a recovery problem is to use discrete flight copies to handle flight delays. Our analysis compared the results from using different numbers of flight copies, which has shown a potentially large overestimation of recovery costs. Further, changing the number of flight copies has an effect on the minimum possible delay cost and hence affects the composition of delays and changes in the final solution. With the discovery of a good upper bound at the end of Phase 1, richer results for the RRTAP can be obtained by using a greater number of flight copies without substantial increases in solution runtimes.

The tail assignment problem used in the planning stage of the RRTAP does not explicitly consider maintenance routing for aircraft. Aircraft maintenance routing is a critical aspect of the airline planning process ensuring the safe operation of the airline fleet. There are a number of different approaches described in Section 2.2.1 used to model the maintenance routing problem, which can be incorporated into the recoverable robustness framework. The one-day routes modelling approach is of particular interest since maintenance planning robustness is inherit in its formulation. While the one-day routes provides a maintenance plan that is not affected by disruptions in the long term, daily schedule perturbations still affect the maintenance routings. In this chapter recoverable robustness was demonstrated to improve the recoverability of the single day routes for the tail assignment problem. It is expected that similar improvements can be achieved with the recoverable robust single day maintenance routing problem. The use of recoverable robustness in conjunction with a one-day maintenance routing formulation is investigated in the following chapter.

The improvement in recoverability for the RRTAP is promising in regards to the wider application of the recoverable robustness framework. The resources of aircraft and crew are highly connected, as such a true analysis on the effect of disruptions must consider both. Thus, a natural extension to the RRTAP is to integrate the planning problems for the aircraft and crew. The integrated aircraft routing and crew scheduling problem presented by Cordeau *et al.* [27] demonstrates the potential benefits from this integration in regards to improved solution quality and robustness. By integrating aircraft and crew, the complexity of the planning problem is significantly increased, and similarly for the recovery problem. One of the greatest hindrances of the recoverable robust solution process is the requirement to solve a large number of recovery problems to provide feedback to the planning problem. As such, it is only possible to develop a recoverable robust integrated aircraft routing and crew scheduling problem if fast solution runtimes can be achieved for the recovery problem. Potential solution approaches for the integrated airline recovery problem are investigated in Chapters 8 and 9 of this thesis.

Chapter 6

Recoverable Robust Single Day Aircraft Maintenance Routing Problem

The application of recoverable robustness to the tail assignment problem is investigated in Chapter 5. The results presented for the recoverable robust tail assignment (RRTAP) demonstrate the potential improvements in recoverability achieved by this technique. While the tail assignment problem is representative of airline planning problems solved in practice, there are many features omitted that greatly affect the solution quality. One such feature is aircraft maintenance planning, which is a necessary consideration to ensure the safe operation of the airline fleet. A novel maintenance planning problem is considered in this chapter to further investigate the application of recoverable robustness to airline planning problems.

The aircraft maintenance routing problem is solved to ensure maintenance checks are performed on aircraft at regular intervals. Various approaches employed to solve this problem are presented in Section 2.2.1, each with a different business practice focus. Three categories of modelling approaches are described in Section 2.2.1, the *big cycle*, *strings* and *one-day routes* approaches. The *big cycle* approach is solved to identify a single flight route that can be operated by all aircraft within the fleet, satisfying maintenance requirements and achieving equal aircraft utilisation. The *strings* approach constructs maintenance feasible flight routes that span between maintenance stations. Finally, the *one-day routes* approach constructs flight strings that span from the start to the end of the day, providing an adequate number of routes departing from each overnight base that terminate at a maintenance station. A key feature of the *one-day routes* approach is the ability to reduce the effects of schedule perturbations from previous days on the maintenance plan for the current day. Since the improvement in recoverability is related to improved operational performance, the *one-day routes* approach is investigated in this chapter.

The contributions of this chapter are:

1. the introduction of a novel modelling approach to integrate the aircraft routing and maintenance planning problems,

2. the further investigation of the recoverable robustness framework by employing alternative planning stage problems,

3. evaluating the recoverable robustness problem using various data sets and a large number of evaluations scenarios.

Firstly, the maintenance planning problem introduced in this chapter is developed using the concept of *one-day routes*. Previous studies employing this modelling approach [47, 54] require the solution of an aircraft routing problem as input. This reduces the efficacy of the approach since only small modifications to the aircraft routing solution can be made. The maintenance planning problem presented in this chapter addresses this by integrating the aircraft routing and maintenance planning problems.

Secondly, the potential of the recoverable robustness framework to improve the recoverability of airline planning problems is presented in Chapter 5. This chapter continues the investigation from Chapter 5 by considering an alternative, more complex, planning-stage problem that more closely represents airline business practices. The application of recoverable robustness to the maintenance planning problem attempts to improve recoverability while ensuring the maintenance plan is satisfied. The problem formulation is a contribution of this chapter, demonstrating the possible simultaneous improvement in maintenance planning and recoverability.

Finally, the evaluation of the recoverable robustness framework in this chapter is performed on large airline schedules to demonstrate the applicability of the approach. The data set used in Chapter 5 is not representative of real-world airline schedules; hence, larger data sets are considered in this chapter to demonstrate the challenges in developing an efficient solution approach. In addition, the effect that increasing the number of evaluation scenarios has on the solution process and improvement in recoverability is assessed.

This chapter is presented in two parts, the formulation of the single day aircraft maintenance routing problem and the application of recoverable robustness. This structure is used to highlight the individual strengths of the maintenance planning approach and the recoverable robustness framework. Section 6.1 discusses the model for the single day aircraft maintenance routing problem. This will provide a full description of the techniques used to formulate this problem, in particular the novel maintenance planning approach. This is followed by a description of the recoverable robust single day aircraft maintenance routing problem in Section 6.2. The recoverable robustness framework in this chapter involves the evaluation of the planning aircraft routing solution using a recovery subproblem. The description of the recovery problem used in the evaluation stage will be presented in Section 6.2. The solution techniques of Benders' decomposition and column generation are employed to solve the problems presented in this chapter. A description of their implementation will be provided in Section 6.3. The results from our experiments will be presented in Section 6.4, demonstrating the improved recoverability achieved for the maintenance planning problem solved for large airline schedules. The conclusions and potential research directions will be discussed in Section 6.5. This work presented in this chapter has been completed in collaboration with Guy Desaulniers and François Soumis.

6.1 Single Day Aircraft Maintenance Routing Problem

The single day aircraft maintenance routing problem (SDAMRP) is solved to find a set of aircraft routes that ensure maintenance feasibility for each individual day. This problem is motivated by the unfortunately common situation where maintenance plans spanning multiple days become infeasible due to schedule perturbations. To avoid the effects of schedule perturbations occurring on previous days, the SDAMRP is solved to generate an adequate number of single day aircraft routes originating from each overnight airport that terminate at maintenance bases. The solution to the SDAMRP provides a maintenance plan at the start of the day that satisfies the maintenance requirements for each aircraft.

There are a number of maintenance checks that must be performed on aircraft to satisfy aviation regulatory requirements. The maintenance check performed most frequently is a type A check, which is modelled in the SDAMRP to be required once every six days. This assumption ensures that a sufficient amount of time is given to achieve high utilisation of aircraft while not exceeding any maintenance requirements. While performing a type A check once every six days may be appropriate for most airlines, it is trivial to alter this to match individual business practices.

The maintenance routing problem developed in this chapter is based upon the concept of maintenance misalignments. *A* **maintenance misalignment** *occurs when an aircraft that requires maintenance at the end of the day is assigned a fight route that does not terminate at a maintenance station.* If there exists any maintenance misalignments in an aircraft routing solution, costly aircraft swaps must be performed during the day to satisfy maintenance requirements. As a contribution of the SDAMRP, each maintenance misalignment is penalised in an attempt to provide a sufficient number of routes departing from each overnight airport that terminate at maintenance stations.

The SDAMRP is an extension upon the aircraft routing problem (ARP) presented in Section 2.1.1. Primarily, the SDAMRP is formulated as the ARP with an additional set of constraints to count the number of maintenance routes departing from each overnight airport. As such, the notation presented in Table 2.1 is used for the SDAMRP with some modifications required to model the additional maintenance constraints. For completeness, all notation used to describe the SDAMRP, including any required modifications, is presented in Table 6.1.

6.1.1 Aircraft routing flight strings

The SDAMRP is developed using the flight string formulation introduced by Barnhart *et al.* [12]. A flight string p is defined as a sequence of flights that is performed by an aircraft during a single day. It is required that each flight string originates and terminates at an overnight airport b, which are contained in the set B. The set of flights used to construct the aircraft flight strings is given by N and all feasible connections are contained in C. The definitions for the sets N and C are provided in Section 2.1.1.

A major difference between the formulation of the ARP and SDAMRP is the partitioning of the set of feasible flight strings by origination airport into disjoint sets. All strings p that are assigned to aircraft originating from overnight airport b are contained in the set P_b. The decision variables y_p equal 1 if flight string p is operated by an aircraft and 0 otherwise. Each flight string is described by the parameters a_{jp} that equal 1 to indicate that flight j is contained in string p and 0 otherwise. The cost of flight string p, c_p, is dependent on the length of the connections contained in that string, which can be weighted as a form of proxy robustness for

B	is the set of all overnight airports b where aircraft flight strings originate and terminate
P_b	is the set of all flight strings p that originate from airport b, the planned flight strings
N	is the set of all flights j
C	is the set of all feasible connections in the network, $C = \{(i,j) \mid i,j \in N \cup B\}$
y_p	$= 1$ if flight string p is used in the planning stage, 0 otherwise
c_p	$=$ the cost of using flight string p in the planning stage
a_{jp}	$= 1$ if flight j is in string p in the planning stage, 0 otherwise
t_p	$= 1$ if string p terminates at a maintenance base at the end of day in the planning stage, 0 otherwise
R_b	is the number of aircraft that overnight at base b
ψ_b	the number of flight strings originating from airport b that terminate at a maintenance base
$F_b(\psi_b)$	is the function in the objective to penalise the number of maintenance misalignments at airport b

Table 6.1: Sets and variables used in the SDAMRP.

the planned solution. An example of a proxy robust approach based upon connection lengths in presented by Grönkvist [45].

Aircraft are a finite resource for airlines that are strategically positioned across airports throughout the network. The number of aircraft that are positioned at overnight airport b at the beginning of each day is given by R_b. Therefore, this parameter provides an upper bound on the number of flight routes that can originate from b in a feasible solution. This model can be solved for any flight schedule since flow balance ensures that a sufficient number of aircraft terminate at each overnight airport b to begin the following days flying.

6.1.2 Maintenance misalignment

The SDAMRP attempts to satisfy the maintenance requirements for each aircraft by penalising the expected number of maintenance misalignments at each overnight base. This is achieved by introducing the variables ψ_b and parameters t_p to count the number of routes departing from airport b that terminate at a maintenance base. The parameters t_p equal 1 to indicate that string p terminates at a maintenance base and 0 otherwise. Using the parameters t_p, the model constraints force the value of ψ_b to equal the number of maintenance routes that depart from airport b.

The maintenance requirements of each aircraft at the start of the day can be described by one of two states, either requiring maintenance that evening or not. Therefore, the expected number of aircraft that require maintenance departing from each overnight airport is conveniently modelled by a binomial distribution. Using the assumption that maintenance is required once every six days, the probability of an aircraft requiring maintenance is 1/6. To minimise the number of maintenance misalignments the penalty function $F_b(\psi_b)$ is introduced. This function penalises the difference between the expected required maintenance routes, given by the binomial distribution, and the actual number of maintenance routes, given by the variables ψ_b. While the penalty function is conveniently described by $F_b(\psi_b)$, it is necessary to define this function in terms of integer variables in order to formulate the SDAMRP as a mixed-integer program. The specific details regarding this penalty function will be provided in Section 6.1.4.

6.1.3 Mathematical model

The parameters and variables described above are used to formulate an aircraft maintenance routing problem that is solved for a single day flight schedule. The complete formulation of the SDAMRP is presented below,

$$\text{(SDAMRP)} \quad \begin{aligned} \min \quad & \sum_{b \in B} \sum_{p \in P_b} c_p y_p + \sum_{b \in B} F_b(\psi_b), && (6.1)\\ \text{s.t.} \quad & \sum_{b \in B} \sum_{p \in P_b} a_{jp} y_p = 1 \quad \forall j \in N, && (6.2)\\ & \sum_{p \in P_b} y_p \leq R_b \quad \forall b \in B, && (6.3)\\ & \psi_b = \sum_{p \in P_b} t_p y_p \quad \forall b \in B, && (6.4)\\ & y_p \in \{0,1\} \quad \forall b \in B, \forall p \in P_b, && (6.5)\\ & \psi_b \geq 0 \quad \forall b \in B. && (6.6) \end{aligned}$$

The objective of the SDAMRP minimises the cost of aircraft routing and any penalties resulting from maintenance misalignments. Constraints (6.2) ensure that every flight j contained in the original schedule N is included on a single flight route. An upper bound on the number of aircraft departing from each overnight airport is given by constraints (6.3). A contribution of this chapter is the inclusion of constraints (6.4) to count the number of routes departing from overnight airport b that terminate at a maintenance station. By counting the number of aircraft routes terminating at a maintenance station, the number of maintenance misalignments at each overnight airport b can be penalised by the function $F_b(\psi_b)$.

6.1.4 Penalty function for maintenance misalignments

As explained in Section 6.1.2, the function $F_b(\psi_b)$ penalises the difference between the expected required and actual number of maintenance routes departing from airport b. The actual number of maintenance routes is given by ψ_b and the expected number is obtained from a binomial distribution, $B(R_b, \phi)$ where $\phi = 1/6$. Given this distribution, we define the variables p_{bi} as the probability that exactly i aircraft at overnight airport b require maintenance at the end of the day. The value of p_{bi} is defined as,

$$p_{bi} = \binom{R_b}{i} \phi^i (1-\phi)^{R_b - i}. \tag{6.7}$$

Using the values calculated for these variables and the actual number of maintenance routes given by ψ_b, it is possible to quantify the expected number of maintenance misalignments at each overnight airport b. This expectation, $E_b(\psi_b)$, is given by,

$$E_b(\psi_b) = \begin{cases} \displaystyle\sum_{i=\psi_b+1}^{R_b} (i - \psi_b) p_{bi} & \text{if } 0 \leq \psi_b < R_b, \\ 0 & \text{if } \psi_b = R_b. \end{cases} \tag{6.8}$$

The penalty function included in the objective of the SDAMRP is described by $F_b(\psi_b) = \tau E_b(\psi_b)$, where τ is an arbitrary positive weight. In the case that $c_p > 0, \forall b \in B, \forall p \in P_b$ in equation (6.1) it may be necessary to use a relative large value for τ in the penalty function to increase the efficacy of this approach.

It is clear from equation (6.8) that the function $F_b(\psi_b)$ is not linear, hence a reformulation is required to construct the SDAMRP as a mixed-integer program. The reformulation of $F_b(\psi_b)$ and the SDAMRP involves modelling the penalty function using a set of integer variables. To describe this, we first define Δ_b^i as a sequence of step sizes,

$$\Delta_b^i = F_b(i-1) - F_b(i) \quad \forall i = 1, \ldots, R_b, \tag{6.9}$$

which is a decreasing sequence in i. Thus, the penalty function can be defined as the sum of this sequence,

$$F_b(\psi_b) = \begin{cases} \displaystyle\sum_{i=\psi_b+1}^{R_b} \Delta_b^i & \text{if } 0 \leq \psi_b < R_b, \\ 0 & \text{if } \psi_b = R_b. \end{cases} \tag{6.10}$$

Now, in the current form it is not possible to use equation (6.10) in the mixed-integer programming formulation of the SDAMRP. The difficulty with equation (6.10) is the lower bound in the summation, $\psi_b + 1$. By introducing the set of binary variables $Z_b^i, i = 1, \ldots, R_b$, it is possible to define $\psi_b = \sum_{i=1}^{R_b} Z_b^i$. The reformulation of the SDAMRP is performed by replacing ψ_b by $\sum_{i=1}^{R_b} Z_b^i$ in constraints (6.4) and $F_b(\psi_b)$ by $F_b(0) - \sum_{i=1}^{R_b} \Delta_b^i Z_b^i$. This results in an linear objective function and a mixed-integer programming formulation of the SDAMRP.

6.2 Recoverable Robust Single Day Aircraft Maintenance Routing Problem

The application of recoverable robustness to the tail assignment problem is presented in Chapter 5. As explained in the previous chapter, recoverable robustness is a framework that attempts to identify a planned solution that is recoverable with *limited effort*. This chapter further investigates the application of recoverable robustness to airline planning problems, using the SDAMRP as an alternative problem for the planning stage. Modelling the planning stage of the recoverable robustness problem using the SDAMRP attempts to minimise the effect of disruptions on the maintenance planning solution.

The recoverable robust single day aircraft maintenance routing problem (SDAMRP-RR) investigates the application of the recoverable robustness framework developed in Chapter 5 to the maintenance planning problem presented in Section 6.1. As such the notation presented in Tables 5.1, 5.2 and 6.1 is used in this section. For conciseness, the notation relevant to the SDAMRP-RR will be presented in Tables 6.2 and 6.3. Only a brief description of the features presented in previous chapters will be given in this section.

L	is the set of all airports l
S	is the set of all scenarios s
P_b^s	is the set of all flight strings p in scenario s that originate from airport b, the recovery variables,
N^s	is the set of flights j included in the flight schedule for scenario s, $N^s \subseteq N$
C^s	is the set of feasible connections between flights in N^s, $C^s = \{(i,j) \| i,j \in N^s\}$, $C^s \subset C$
U_j	is the set of all delay copies v for flight $j \in N^s$,
$\hat{N}^s$	is the set of flight-copy pairs j_v included in scenario s, $j \in N^s, v \in U_j$
$\hat{C}^s$	is the set of feasible connections between flight-copy pairs in $\hat{N}^s$, $\hat{C}^s = \{(i_u, j_v) \| i_u, j_v \in \hat{N}^s\}$
$\bar{C}^s$	is the set of feasible connections in C related to the connections in $\hat{C}^s$, $\bar{C}^s = \{(i,j) \| (i,j) \in C \wedge (i_u, j_v) \in \hat{C}^s, \exists u \in U_i, \exists v \in U_j\}$

Table 6.2: Additional sets used in the SDAMRP-RR.

The SDAMRP-RR simultaneously solves the planning and recovery aircraft routing problems, where the latter is used to evaluate the recoverability of the former. The evaluation stage is a fundamental aspect of the recoverable robustness framework, which is used to explicitly evaluate the recoverability of the planning stage solution. In the SDAMRP-RR, the evaluation stage solves a recovery problem for a set of disruption scenarios S. Improved recoverability of the planned aircraft routing solution is assessed by the expected recovery costs and the effort required to return operations back to plan for each scenario $s \in S$. The planning problem is given by the SDAMRP and the recovery aircraft routing problem developed in this section ensures the satisfaction of the maintenance plan. In this section a description of the recovery problem is provided along with the details regarding the integration of planning and evaluation stages.

6.2.1 Aircraft recovery flight strings

Similar to the SDAMRP, the feasible flight strings for aircraft are partitioned by overnight airport b into disjoint sets. All flight strings p departing from overnight airport b in scenario s are contained in the set P_b^s. The flight string formulation is used to define the aircraft routes in both the planning and recovery problems. However, there are significant differences in each problem affecting the string construction. Similar to the SDAMRP, the decision variables y_p^s equal 1 if string p is operated by an aircraft in scenario s and 0 otherwise. Each flight string originates from an overnight base, however the connections used prior to the disruption in scenario s are fixed. Since these connection are fixed, it is possible to omit these flights from the schedule and the connection network. As such, an alternative set of flights, N^s, is used to define the recovery schedule for each scenario s and all feasible connections between the flights in N^s are contained in C^s. It is important to note that in describing features from the recovery tail assignment problem (RTAP) that are relevant to the SDAMRP-RR the sets N^D and C^D are replaced by the sets N^s and C^s respectively.

Since the recovery flight schedule defined by the set N^s omits all flights preceding the disruption described by scenario s, the location of each aircraft at the start of the disruption must be specified. The possible locations of an aircraft at the start of a disruption are contained in L, which describes all airports l within the flight network. The parameter v_{lp}^s indicates that in scenario s the aircraft operating flight string p in the planning stage is positioned at airport l at the start of the disruption.

y_p^s	$= 1$ if flight string p is used for recovery in scenario s, 0 otherwise
c_p^s	$=$ the cost of using flight string p for recovery in scenario s
a_{jp}^s	$= 1$ if flight j is in string p in scenario s, 0 otherwise
e_{ijp}, e_{ijp}^s	$= 1$ if connection (i, j) is in string p in the planning stage or scenario s respectively, 0 otherwise
t_p^s	$= 1$ if string p terminates at a maintenance base at the end of day for recovery in scenario s, 0 otherwise
o_{bp}^s	$= 1$ if string p terminates at airport b at the end of the day for recovery in scenario s, 0 otherwise
v_{lp}^s	$= 1$ if the aircraft operating the planned flight string p is positioned at airport l at the start of scenario s, 0 otherwise
c_j^s	$=$ the lost revenue and passenger delay cost resulting from cancelling flight j in scenario s
g^{LR}	the loss rate parameter, a quantitative measure of passenger dissatisfaction resulting from a flight cancellation
z_j^s	$= 1$ if the flight j is cancelled in scenario s, 0 otherwise
$\epsilon_{ij}^{s+}, \epsilon_{ij}^{s-}$	$\begin{cases} \epsilon_{ij}^{s+} = 1, \epsilon_{ij}^{s-} = 0 & \text{if connection } (i,j) \text{ is used for the planning stage but not for recovery in scenario } s \\ \epsilon_{ij}^{s+} = 0, \epsilon_{ij}^{s-} = 1 & \text{if connection } (i,j) \text{ is used for recovery in scenario } s \text{ but not for the planning stage} \\ \epsilon_{ij}^{s+} = 0, \epsilon_{ij}^{s-} = 0 & \text{if connection } (i,j) \text{ is not used in both the planning stage and recovery in scenario } s \end{cases}$
g^{SW}	the swap cost, weight applied to ϵ_{ij}^{s-} in the objective function
w^s	weight for each scenario s in the objective function
Z_b^i	$= 1$ if i flight strings originating from airport b terminate at a maintenance base at the end of the day in the planning stage
Δ_b^i	weight applied to Z_b^i to penalise the ith maintenance misalignment

Table 6.3: Additional variables used in the SDAMRP-RR.

6.2.2 Recovery policies

The recovery aircraft routing problem used in the evaluation stage of the SDAMRP-RR is a more general form of the RTAP presented in Section 3.1.1. The major difference between the aircraft recovery problem and the RTAP is the partitioning of the feasible flight routes. As explained above, the flight routes for each scenario in the evaluation stage are partitioned by origination airport, similar to the SDAMRP. In addition, it is not necessary to construct individual routes for each aircraft, however it is necessary to ensure the same number of routes depart from each origination airport as set in the planning stage. Since these are only subtle differences between the recovery problem of the evaluation stage and the RTAP, the definitions given in Section 3.1.1 can be used.

The recovery problem is solved with a full set of recovery options, including aircraft rerouting and flight delays and cancellations. In the event of a disruption, the aircraft routes in the planning stage solution may not be feasible for the disrupted flight schedule N^s, hence the construction of new routes is required. The new aircraft routes are generated in the recovery process to respect the origination locations and maintenance requirements. A necessary feature of aircraft recovery problems is that maintenance critical aircraft are assigned flight routes terminating at maintenance stations. This is enforced in the generation of recovery flight routes in the SDAMRP-RR.

The recovery policies of flight delays and cancellations are implemented using the same techniques presented in Section 3.1.1 for the RTAP. Flight delays are implemented using flight copies, with all copies v for flight j contained in the set U_j. Flight cancellation in the SDAMRP-RR are modelled with the variables z_j^s, that equal 1 to indicate flight j is cancelled in scenario s at a cost of c_j^s. A recovery window, as described in Section 3.1.1, is employed to reduce the complexity of the evaluation stage in the SDAMRP-RR. However, the recovery window is only used to restrict the flights which can be delayed. Aircraft rerouting and flight cancellations are permitted on all flights contained in the set N^s.

There is a subtle difference in the definition of flight cancellation costs c_j^s for the SDAMRP-RR compared to the RRTAP in Chapter 5. In this chapter, the cost of a flight cancellation is given by the cost of rebooking passengers onto alternative flights and the amount of lost revenue from each passenger that is not rebooked. This definition does not quantitatively describe any loss of good will as a result of passenger dissatisfaction. In the SDAMRP-RR, this is modelled with the introduction of a *loss rate* parameter g^{LR} to indicate the willingness of passengers to rebook with the airline. There are three important ranges for the *loss rate* parameter i) $g^{LR} < 1$ indicates that passengers are not discouraged to travel with the airline in the future and some of the lost revenue is recaptured by passengers rebooking with the airline, ii) $g^{LR} = 1$ does not account for any recapture or loss of future bookings, and iii) $g^{LR} > 1$ describes the situation where passengers are less likely to rebook with the airline in the future. The results will demonstrate the effect that different values of g^{LR} has on the efficacy of the recoverable robustness framework.

Flight schedules are designed to ensure that an adequate number of aircraft terminate at each overnight base to continue the following days flying. This is the result of flow balance, and hence model constraints are not required to enforce this coverage in planning aircraft routing problems. Unfortunately in recovery problems, flight cancellations commonly break the flow balance of the original schedule. In addition, the recovery schedule N^s may not contain the required flights to guarantee adequate end of day coverage at each overnight base. To address this aspect of recovery problems for the cyclic schedules used in this chapter, the parameter R_b specifies the minimum number of aircraft required to terminate at each overnight base b. In the model constraints for the aircraft recovery problem, the parameters o^s_{bp} are introduced, which equal 1 if string p in scenario s terminates at overnight airport b and 0 otherwise.

6.2.3 Objective of minimal deviation

An important feature of the recoverable robustness framework is the objective of minimal deviation from the planned solution. Since the SDAMRP-RR is solved as an aircraft routing problem, each aircraft is not individually identified. Therefore, the objective of minimal deviation is implemented by attempting to construct aircraft routes in the recovery problem with the same connections used by aircraft in the solution to the planning stage. However, the set of connections in the planning and evaluations stages is different as a result of flights omitted from the recovery schedule and the alternative departure times representing flight delays. Therefore, the set $\bar{C}^s$ is defined to contain the connections $(i, j) \in C$ where there exists copies $u \in U_i$ and $v \in U_j$ such that $(i_u, j_v) \in \hat{C}^s$. As stated previously, a flight string p is defined by a sequence of connected flights. Thus, the parameters e_{ijp} and e^s_{ijp} are defined to equal 1 if the connection $(i, j) \in \bar{C}^s$ is included in string p for the planning stage and recovery scenario s respectively.

To minimise the number of connection changes between the planning and recovery solutions we introduce the variables ϵ^{s+}_{ij} and ϵ^{s-}_{ij}. If a connection (i, j) is used in a planned route and not in a recovery route for scenario s, ϵ_{ij}^{s+} equals 1 and ϵ_{ij}^{s-} equals 0. Alternatively, if connection (i,j) is used in a recovery route for scenario s and not in a planned route, ϵ_{ij}^{s+} equals 0 and ϵ_{ij}^{s-} equals 1. Finally, if the connection is not used in either a planned or recovery route for scenario s then both ϵ_{ij}^{s+} and ϵ_{ij}^{s-} equal 0. In the objective function we include only the variables ϵ_{ij}^{s-} to count the total number of connections used in recovery that are not used in the planned aircraft routes. This is done to avoid double counting the total number of changes resulting from the inclusion of both sets of variables.

The mathematical model for the SDAMRP-RR is given by,

(SDAMRP-RR)

$$\min \quad \sum_{p\in P} c_p y_p + \sum_{b\in B}\left\{F_b(0) - \sum_{i=1}^{R_b} \Delta_b^i Z_b^i\right\}$$

$$+\sum_{s\in S} w^s \left\{\sum_{p\in P^s} c_p^s y_p^s + \sum_{j\in N^s} g^{LR} c_j^s z_j^s + \sum_{(i,j)\in C^s} g^{SW} \epsilon_{ij}^{s-}\right\}, \tag{6.11}$$

$$\text{s.t.} \quad \sum_{b\in B}\sum_{p\in P_b} a_{jp} y_p = 1 \quad \forall j \in N, \tag{6.12}$$

$$\sum_{p\in P_b} y_p \le R_b \quad \forall b \in B, \tag{6.13}$$

$$\sum_{i=1}^{R_b} Z_b^i - \sum_{p\in P_b} t_p y_p = 0 \quad \forall b \in B, \tag{6.14}$$

$$\sum_{b\in B}\sum_{p\in P_b^s} a_{jp}^s y_p^s + z_j^s = 1 \quad \forall s \in S, \forall j \in N^s, \tag{6.15}$$

$$\sum_{p\in P_b^s} y_p^s \le R_b \quad \forall s \in S, \forall b \in B, \tag{6.16}$$

$$\sum_{b_1\in B}\sum_{p\in P_{b_1}^s} o_{b_2 p}^s y_p^s \ge R_{b_2} \quad \forall s \in S, b_2 \in B, \tag{6.17}$$

$$\sum_{b\in B}\sum_{p\in P_b^s} v_{lp}^s y_p^s - \sum_{b\in B}\sum_{p\in P_b} v_{lp}^s y_p = 0 \quad \forall s \in S, \forall l \in L, \tag{6.18}$$

$$\sum_{b\in B}\sum_{p\in P_b^s} t_p^s v_{lp}^s y_p^s - \sum_{b\in B}\sum_{p\in P_b} t_p v_{lp}^s y_p \ge 0 \quad \forall s \in S, \forall l \in L, \tag{6.19}$$

$$\sum_{b\in B}\sum_{p\in P_b} e_{ijp} y_p - \sum_{b\in B}\sum_{p\in P_b^s} e_{ijp}^s y_p^s = \epsilon_{ij}^{s+} - \epsilon_{ij}^{s-} \quad \forall s \in S, \forall (i,j) \in \bar{C}^s, \tag{6.20}$$

$$y_p \in \{0,1\} \ \forall b \in B, \forall p \in P_b, \quad y_p^s \in \{0,1\} \ \forall s \in S, \forall b \in B, \forall p \in P_b^s, \tag{6.21}$$

$$z_j^s \in \{0,1\} \ \forall s \in S, \forall i \in N^s \quad \epsilon_{ij}^{s+}, \epsilon_{ij}^{s-} \ge 0 \ \forall s \in S, \forall (i,j) \in \bar{C}^s, \tag{6.22}$$

$$Z_b^i \in \{0,1\} \quad \forall b \in B, \forall i \in \{1,\ldots,R_b\}. \tag{6.23}$$

The SDAMRP-RR combines the planning stage of the SDAMRP with an evaluation stage provided by aircraft recovery subproblems. The objective of the SDAMRP-RR is the sum of the objective from the SDAMRP and the weighted cost of recovery and penalties resulting from connection changes in each scenario. The modifications described in Section 6.1.4 to formulate the SDAMRP as a mixed-integer program have been applied in the formulation of the SDAMRP-RR, given by equations (6.11) and (6.14). Constraints (6.12)-(6.14) describe the SDAMRP as the planning stage for the recoverable robust problem.

The constraints related to the evaluation stage of the recoverable robust problem are given by equations (6.15)-(6.20). Constraints (6.15) are the flight coverage constraints including the decision variables z_j to describe flight cancellations. An upper bound on the number of aircraft flight strings departing from each overnight base in a feasible recovery solution is given by constraints (6.16). Since flow balance is not guaranteed in the recovery schedule, the number of aircraft required to terminate at each overnight base is provided by constraints (6.17).

The link between the planning and recovery stages is provided by constraints (6.18)-(6.20). Constraints (6.18) are included to identify the number of aircraft located at each airport at the start of a disruption described by scenario s. As a key feature of the SDAMRP-RR, the aircraft expected to receive maintenance at the end of the day are still assigned a flight route that terminates at a maintenance station in the recovered solution. Constraints (6.19) ensure that the number of routes originating from airport l that terminate at maintenance stations is the same in the planning and evaluation stages. Finally, a measure of *limited effort* in the recovery of the SDAMRP is given by the number of different connections used between the planning and evaluation stages. Therefore, constraints (6.20) are included in the SDAMRP-RR to count the number of connections used in the recovered solution that are not set in the planning stage.

6.3 Solution Methodology

The SDAMRP and SDAMRP-RR are large-scale optimisation problems that require the use of decomposition techniques to develop efficient solution approaches. The decomposition techniques employed in this chapter are column generation and Benders' decomposition. Both of these solution techniques are introduced in Chapter 4 and applied to the RRTAP in Chapter 5. For conciseness, only the key features of each solution approach will be presented in this section.

6.3.1 Benders' decomposition

The SDAMRP-RR displays a similar structure to problem P presented in Section 4.2. In particular, the first-stage (deterministic) decision variables x of problem P are related to the planning aircraft routing variables y_p of the SDAMRP. Also, it can be observed that the second-stage (probabilistic) decision variables y^s are related to the aircraft recovery variables y_p^s for each scenario s. Given the similarity between the SDAMRP-RR and problem P, the standard implementation of Benders' decomposition described in Section 4.2 can be applied to achieve an efficient solution approach.

In the SDAMRP-RR, there is a clear separation of variables between the planning and recovery problems to define the master and subproblems respectively. The subproblems given by this decomposition, which we will label as the primal Benders' subproblem (PBSP-s), are defined for each scenario s to consist of the recovery string variables y_p^s, the cancellation variables z_j^s, and the comparison count variables ϵ_{ij}^{s+} and ϵ_{ij}^{s-}. Each of the subproblems represent a recovery problem that is solved for a given disruption scenario to evaluate the effect of the planning decisions on the cost of recovery.

The Benders' master problem (BMP) for the SDAMRP-RR is defined as a planning aircraft routing problem given by the SDAMRP. The BMP is formulated to contain the planning string variables y_p^s, the constraints (6.12)-(6.14) and a set of additional cuts to reflect the decisions made in each PBSP-s. The decision variables φ^s are introduced in the added cuts to provide a lower bound on the objective value of the PBSP-s for each scenario s. The BMP formulated without any added cuts is identical to the SDAMRP. Since the added cuts progressively restrict the feasible region, all feasible solutions to the BMP are feasible for the SDAMRP. It should also be stated that due to this property the solution to the SDAMRP is a lower bound on the optimal solution to the BMP.

Evaluating the recoverability of the planning solution is performed by solving the PBSP-s with fixed solution values from the BMP. The fixed solution to the BMP, given by $\bar{\mathbf{y}}_n = \{\bar{y}_{np}, \forall b \in B, \forall p \in P_b | \bar{y}_{np} = 1\}$, represents the best possible planning solution in iteration n achieved with the previously realised evaluation information. The benefits of Benders' decomposition is observed in this stage since solving each individual subproblem with a fixed master problem solution is much simpler than solving the original problem as a whole. Fixing the solution to the planning stage significant reduces the complexity of the original problem by permitting the evaluation stage to be solved purely as a series recovery problems.

The PBSP-s is defined as an aircraft recovery problem given by,

$$\text{(PBSP-}s\text{)}\quad \begin{aligned} \mu_s(\bar{\mathbf{y}}_n) = \min \quad & \sum_{p\in P^s} c_p^s y_p^s + \sum_{j\in N^s} g^{LR} c_j^s z_j^s + \sum_{(i,j)\in C^s} g^{SW} \epsilon_{ij}^{s-}, && (6.24)\\ \text{s.t.} \quad & \sum_{b\in B}\sum_{p\in P_b^s} a_{jp}^s y_p^s + z_j^s = 1 \quad \forall j \in N^s, && (6.25)\\ & \sum_{p\in P_b^s} y_p^s \leq R_b \quad \forall b \in B, && (6.26)\\ & \sum_{b_1\in B}\sum_{p\in P_{b_1}^s} o_{b_2 p}^s y_p^s \geq R_{b_2} \quad b_2 \in B, && (6.27)\\ & \sum_{b\in B}\sum_{p\in P_b^s} v_{lp}^s y_p^s = \sum_{b\in B}\sum_{p\in P_b} v_{lp}^s \bar{y}_{np} \quad \forall l \in L, && (6.28)\\ & \sum_{b\in B}\sum_{p\in P_b^s} t_p^s v_{lp}^s y_p^s \geq \sum_{b\in B}\sum_{p\in P_b} t_p v_{lp}^s \bar{y}_{np} \quad \forall l \in L, && (6.29)\\ & \sum_{b\in B}\sum_{p\in P_b^s} e_{ijp}^s y_p^s + \epsilon_{ij}^{s+} - \epsilon_{ij}^{s-} = \sum_{b\in B}\sum_{p\in P_b} e_{ijp} \bar{y}_{np} \quad \forall (i,j) \in \bar{C}^s, && (6.30)\\ & y_p^s \in \{0,1\} \;\; \forall p \in P^s, \quad z_j^s \in \{0,1\} \;\; \forall j \in N^s, && (6.31)\\ & \epsilon_{ij}^{s+}, \epsilon_{ij}^{s-} \geq 0 \quad \forall (i,j) \in \bar{C}^s. && (6.32) \end{aligned}$$

The objective of the PBSP-s minimises the cost of recovery, in particular flight delay and cancellation costs, and penalise any changes from the connections set in the planning stage. The dual variables for this problem are defined as $\boldsymbol{\alpha}^s = \{\alpha_j^s, \forall j \in N^s\}$, $\boldsymbol{\beta}^s = \{\beta_b^s, \forall b \in B\}$, $\boldsymbol{\delta}^s = \{\delta_b^s, \forall b \in B\}$, $\boldsymbol{\gamma}^s = \{\gamma_l^s, \forall l \in L\}$, $\boldsymbol{\lambda}^s = \{\lambda_l^s, \forall l \in L\}$ and $\boldsymbol{\rho}^s = \{\rho_{ij}^s, \forall (i,j) \in C^s\}$ for the constraints (6.25)-(6.30) respectively.

The general application of Benders' decomposition solves the PBSP-s to identify either a feasibility or optimality cut in each iteration. A feasibility cut is added to the BMP in the event that the PBSP-s is proved to be infeasible, conversely an optimality cut is added if a feasible solution exists. Because of the similar structure of the variables in the PBSP-s and BMP, an initial feasible solution can be constructed for the PBSP-s using $\bar{\mathbf{y}}_n$. As a result, only optimality cuts are generated from the PBSP-s and added to the BMP. The Benders' optimality cut is given by,

$$\varphi^s \geq \sum_{j\in N^s} \alpha_j^s + \sum_{b\in B} \left\{ R_b \left(\beta_b^s + \delta_b^s\right) + \sum_{l\in L}\sum_{p\in P_b} v_{lp}^s y_p \left(\gamma_l^s + t_p \lambda_l^s\right) + \sum_{(i,j)\in C^s}\sum_{p\in P_b} e_{ijp} y_p \rho_{ij}^s \right\}. \quad (6.33)$$

Given an optimal solution to the PBSP-s, the dual solutions used to construct the cut given by equation (6.33) represents an extreme point of the dual feasible region. Since the PBSP-s

is a highly degenerate problem, there are potentially many extreme points in the dual problem that provide the same objective value. The efficacy of the cuts constructed from each of these extreme points when added to the BMP is not identical. Therefore, the convergence of the solution process can be improved by selecting the most dominate. There are many approaches that can be applied to identify the most dominate optimality cut.

The rate of convergence of the Benders' decomposition solution process is highly dependent on the strength the cuts added to the master problem. A common approach used to identify the extreme point in the dual problem providing the most dominate optimality cut, called the Pareto-optimal cut, is presented by Magnanti and Wong [60]. The use of the Magnanti-Wong method to identify Pareto-optimal cuts is demonstrated in the solution process of various airline optimisation problems [27, 63, 69] and in Chapter 5. The improvements in solution runtimes achieved by implementing this method is presented in Section 5.3.3. The Magnanti-Wong method has been applied to find Pareto-optimal cuts from the solution of the PBSP-s in this chapter in an attempt to achieve similar runtime improvements.

The BMP is solved to find the best possible planning solution relative to the cuts added from the PBSP-s. All Benders' cuts ω added to the BMP from the PBSP-s for scenario s are contained in the set Ω^s. The BMP is given by,

$$
\text{(BMP)}\quad
\begin{aligned}
\Phi_n = \min \quad & \sum_{p \in P} c_p y_p + \sum_{b \in B} \left\{ F_b(0) - \sum_{i=1}^{R_b} \Delta_b^i Z_b^i \right\} + \sum_{s \in S} w^s \varphi^s, && (6.34)\\
\text{s.t.} \quad & \sum_{b \in B} \sum_{p \in P_b} a_{jp} y_p = 1 \quad \forall j \in N, && (6.35)\\
& \sum_{p \in P_b} y_p \leq R_b \quad \forall b \in B, && (6.36)\\
& \sum_{i=1}^{R_b} Z_b^i - \sum_{p \in P_b} t_p y_p = 0 \quad \forall b \in B, && (6.37)\\
& \varphi^s - \sum_{b \in B} \left\{ \sum_{l \in L} \sum_{p \in P_b} v_{lp}^s y_p \left(\gamma_l^{\omega s} + t_p \lambda_l^{\omega s} \right) + \sum_{(i,j) \in C^s} \sum_{p \in P_b} e_{ijp} y_p \rho_{ij}^{\omega s} \right\} \\
& \qquad \geq \sum_{j \in N^s} \alpha_j^{\omega s} + \sum_{b \in B} R_b \left(\beta_b^{\omega s} + \delta_b^{\omega s} \right) \quad \forall s \in S, \forall \omega \in \Omega^s, && (6.38)\\
& y_p \in \{0, 1\} \quad \forall b \in B, \forall p \in P_b, && (6.39)\\
& Z_b^i \in \{0, 1\} \quad \forall b \in B, \forall i \in \{1, \dots, R_b\}, && (6.40)\\
& \varphi^s \geq 0 \quad \forall s \in S. && (6.41)
\end{aligned}
$$

The objective of the BMP minimises the cost of aircraft routing, the penalties arising from maintenance misalignments and the weighted sum of the lower bounds on the PBSP-s objective values. The constraints (6.35)-(6.37) are identical to constraints (6.2)-(6.4) from the SDAMRP following the modification described in Section 6.1.4. The additional constraints representing Benders' optimality cuts constructed from the dual solutions to the PBSP-s are given by equations (6.38).

The lower and upper bounds on the objective value for the PBSP-s in iteration n are given by the solution value of $\bar{\varphi}_n^s$ and the objective value $\mu_s(\bar{\mathbf{y}}_n)$ respectively. The iterative solution process between the BMP and the PBSP-s progressively restricts the feasible region with the addition of cuts to improve upon the lower bounds given by φ^s. This improvement in the bounds for each scenario results in a strictly non-decreasing lower bound on the objective value of the original problem, given by Φ. Simultaneously, the addition of cuts also has the effect of reducing the upper bound on the objective value of the original problem. However, the sequence of upper bounds from each iteration is not strictly non-increasing. The upper bound on the original problem in iteration n is given by,

$$\Phi_n^{UB} = \sum_{p \in P} c_p \bar{y}_{np} + \sum_{b \in B} \left\{ F_b(0) - \sum_{i=1}^{R_b} \Delta_b^i \bar{Z}_{nb}^i \right\} + \sum_{s \in S} \mu_s(\bar{\mathbf{y}}_n), \tag{6.42}$$

where $\bar{\mathbf{Z}}_n = \{\bar{Z}_{nb}^i, \forall b \in B, \forall i \in \{1, \ldots, R_b\} | \bar{Z}_{nb}^i = 1\}$ are the fixed solution values for the maintenance count variables in iteration n. The best upper bound identified during the solution process is labelled as $\hat{\Phi}^{UB} = \max_n \{\Phi_n^{UB}\}$. Using the lower and upper bounds on the original problem, the optimality gap in iteration n is calculated by,

$$Gap_n = \frac{\hat{\Phi}^{UB} - \Phi_n}{\Phi_n}. \tag{6.43}$$

Two different stopping conditions are used in the SDAMRP-RR, the first is given by the gap between the upper and lower bounds on the original problem and the other is related to the addition of cuts. By defining the optimality gap using equation (6.43), the solution quality can be measured at the end of each iteration. If the optimality gap lies within a desired range, then the solution process can be terminated.

Alternatively, optimality cuts are added to the BMP in iteration n if the difference between the lower and upper bounds, φ_n^s and $\mu_s(\bar{\mathbf{y}}_n)$ respectively, for scenario s violate an optimality condition. The optimality condition that is implemented for this problem is adopted from a condition presented by Papadakos [69], which is given by,

$$\frac{\mu_s(\bar{\mathbf{y}}_n) - \varphi_n^s}{\Phi_n} \leq \varepsilon. \tag{6.44}$$

The tolerance level, ε, that is used for this problem is 10^{-4}. Using this optimality condition, the optimal solution to the original problem is identified when (6.44) is satisfied for all scenarios s in a single iteration. The optimality condition given by equation (6.44) is identical to that used in Chapter 5.

While it is expected that the optimality gap stopping condition will be reached prior to the cut addition condition, both stopping conditions are implemented for this problem. The primary purpose of the latter stopping condition is to identify when cuts must be added from each subproblem. Using this condition stops redundant cuts being added to the master problem from scenarios that are already solved close to optimality. The optimality gap condition takes a global view of the problem, terminating the solution process when a desired gap is reached.

The two-phase algorithm

The heuristic two-phase algorithm described by Algorithm 5.1 is applied in the Benders' decomposition solution process for the SDAMRP-RR. The implementation of Algorithm 5.1 in this chapter is identical to that presented in Chapter 5, as such no further explanation is required.

A feature of the two-phase algorithm that is not discussed in the previous chapter is the calculation of the optimality gap at the completion of Phase 2. Since the integrality requirements are not reintroduced for the PBSP-s, the solution to the BMP at the end of Phase 2 may not be the integer optimal solution to the SDAMRP-RR. The optimality gap between the current implementable solution and the optimal solution can be found by solving the PBSP-s for all scenarios s to integral optimality. Using equation (6.42) to calculate the upper bound on the optimal solution and setting $\hat{\Phi}^{UB}$ to this, the optimality gap of the implementable solution is calculated by equation (6.43).

Parallel computing

The Benders' decomposition framework lends itself to parallel computing methods as demonstrated by Linderoth and Wright [58]. Since each of the recovery subproblems in the SDAMRP-RR are completely separable and solved independently, it is possible to solve each of these concurrently. The concurrent approach for solving the Benders' decomposition subproblems does not affect the solution methods for each individual problem, as such the generation and addition of optimality cuts remains the same. The only difference between the sequential and concurrent approach is in the implementation of the Benders' decomposition process to include the distribution of subproblems onto a number of compute threads.

The distribution of the subproblems for the SDAMRP-RR attempts to achieve the minimum amount of idle time for all threads. In the first iteration, the subproblems are assigned to threads in numerical order, where the first n subproblems are each solved on one of the n threads. Once a thread finishes executing the solution process for the current subproblem, then next subproblem in the list is solved on that thread. The evaluation stage for the current iteration concludes when all subproblems have been assigned to a thread and the execution of each thread has completed.

For the subsequent iterations, the solution runtime for each subproblem in the previous iteration is used in the assignment process of subproblems to threads. The subproblems are sorted by solution runtime in descending order and the n subproblems with the longest runtime in the previous iteration are each assigned to one of the n threads. As in the previous case, once a thread completes the execution of the solution process the next subproblem in the runtime sorted list is assigned to that thread.

Significant improvements in the convergence of the Benders' decomposition solution process are observed by the implementation in a parallel computing environment. The results in Section 6.4.5 will detail the difference in the optimality gaps achieved with the sequential and concurrent solution approaches.

6.3.2 Column generation

Column generation is applied to optimisation problems formulated with a large number of variables displaying a special combinatorial structure. Each of the three problems presented in this chapter, the SDAMRP, BMP and PBSP-s, display such a structure with the variables describing paths traversed by aircraft through a connection network. Since flight string variables are used in the formulation of each of these problems, the implementations of the column generation solution approach are very similar. A detailed description of column generation implemented to solve the PBSP-s will be provided in this section. For completeness, the differences in the implementation of column generation for the SDAMRP, BMP and PBSP-s will also be highlighted and explained.

As explained in Section 4.1.1, the restricted master problem (RMP) is formulated as a restriction on the original problem by containing only a subset of all possible variables. This restriction is defined by the sets $\bar{P}_b \subseteq P_b$ and $\bar{P}_b^s \subseteq P_b^s$ which replace P_b and P_b^s respectively in the SDAMRP, BMP and PBSP-s. The solution to the RMP will provide an upper bound on the objective value of the original problem, that is improved with the addition of variables to the sets $\bar{P}_b$ and $\bar{P}_b^s$.

Column generation for the PBSP-s

The flight string variables in the PBSP-s define paths through a connection network that originate and terminate at overnight airports. Given this variable structure, the set of all possible variables P_b^s can be described by the feasible region of a network flow problem. Using the dual variables defined in Section 6.3.1, the reduced cost of a flight string p originating from overnight airport b is given by,

$$\bar{c}_p^{sb} = c_p^{sb} - \sum_{j \in N^s} a_{jp}^s \alpha_j^s - \beta_b^s - \sum_{b \in B} o_{bp}^s \delta_b^s - \sum_{l \in L} v_{lp}^s \left\{ \gamma_{lb}^s + t_p^s \lambda_l^s \right\} - \sum_{(i,j) \in C^s} e_{ijp}^s \rho_{ij}. \tag{6.45}$$

In the optimal solution to the RMP, all variables contained in $\bar{P}_b^s$ have a reduced cost of $\bar{c}_p^{sb} \geq 0$ and $\bar{c}_p^{sb} = 0$ for all variables contained in the basis. This indicates that an improvement in the objective value of the RMP is only achieved through the addition of variables from the set $P_b^s \backslash \bar{P}_b^s, \forall b \in B$. By setting the objective function of a shortest path problem to equation (6.45), the solution will identify variables with a reduced cost $\bar{c}_p^{sb} \leq 0$ that are expected to improve the objective function of the RMP.

Since negative reduced cost variables are identified by the solution to a network flow problem, the solution methods presented in Section 4.1.3 can be applied to the PBSP-s. The connection network defined for the PBSP-s is an acyclic directed graph, as such the nodes can be sorted in a topological order. By sorting the nodes in a topological order the reaching algorithm described by Algorithm 4.1 can be employed to solve the shortest path problem in $O(m)$ time, where m is the number of arcs in the network.

Column generation for the BMP and SDAMRP

Since the variables in the BMP describe the flow of aircraft through a connection network similar to the PBSP-s, a shortest path problem can also be solved to identify the minimum reduced cost variables. The major difference between the shortest path problems solved to identify negative reduced cost columns for the BMP and the PBSP-s is the form of the objective function that is used. Since the RMP formulated for the BMP has a different structure to the PBSP-s, a different set of dual variables must be defined. The dual variables for the BMP are given by $\mathbf{u} = \{u_j, \forall j \in N\}$, $\mathbf{v} = \{v_b, \forall b \in B\}$, $\mathbf{w} = \{w_b, \forall b \in B\}$ and $\boldsymbol{\kappa} = \{\kappa^{s\omega}, \forall \omega \in \Omega^s\}$ for constraints (6.35)-(6.38) respectively. Therefore, the reduced cost function of a variable in the BMP originating from overnight airport b is given by,

$$\bar{c}_p^b = c_p^b - \sum_{j \in N^s} a_{jp} u_j^s - v_b + t_p w_b + \sum_{s \in S} \sum_{\omega \in \Omega^s} \left\{ \sum_{l \in L} v_{lp}^s \left(\gamma_l^{\omega s} + t_p \lambda_l^{\omega s} \right) + \sum_{(i,j) \in C} e_{ijp} \rho_{ij}^{\omega s} \right\} \kappa^{s\omega}. \quad (6.46)$$

Solving a shortest path problem using equation (6.46) as the objective function identifies negative reduced cost variables for the BMP. Since the connection network defined for the BMP can be sorted in a topological order, this shortest path problem can be solved using Algorithm 4.1.

The connection networks defined for the SDAMRP and BMP are identical, as such the same set of feasible columns is defined for both problems. This implies that the shortest path problem employed to identify negative reduced cost columns for the BMP can also be used for the SDAMRP. The only modification required is the elimination of the final term in equation (6.46) to define the objective function of the shortest path problem for the SDAMRP.

6.3.3 Trust region

The standard implementation of the Benders' decomposition solution process is commonly affected by slow convergence to the optimal solution. One aspect affecting the convergence of the algorithm is the efficacy of the cuts generated in the subproblem. A number of approaches have been proposed to address this difficulty by improving the strength of each added cut to tighten the feasible region of the master problem quickly and efficiently [38, 60, 68]. As stated in Section 6.3.1, the Magnanti-Wong method [60] has been implemented in the solution process for the SDAMRP-RR to identify Pareto-optimal cuts.

The quality of the solution to the Benders' master problem is also identified as a factor affecting the convergence of the solution approach. Rei *et al.* [71] present an example applying local branching to the Benders' master problem to improve the bounds on the problem while generating multiple optimality cuts. This approach also reduces the number of times that the master problem is solved, which is described as a time consuming feature of the Benders' decomposition approach.

An undesired effect of the Benders' decomposition solution approach is the potentially large distances between master problem solutions from consecutive iterations. The movement of the master problem solution to different parts of the feasible region between iterations may render many of the added cuts ineffective. This increases the number of iterations executed in the Benders' decomposition solution process, requiring the generation of more optimality and feasibility cuts. An approach proposed to address this involves defining a trust region in the formulation of the master problem. A trust region enforces the solution in the current iteration to be found within a neighbourhood around the master problem solution from the previous iteration. This is implemented with either a regularisation term in the master problem objective function [75] or an additional set of constraints [58, 78].

The trust region in the SDAMRP-RR introduces a set of constraints to *measure* the distance between solutions of consecutive iterations. Since the BMP is solved using column generation, it is not possible to define the trust region based upon the solution values alone. As explained in Section 6.1, a flight string describes a set of connected flights to be performed by a single aircraft. Therefore, the optimal solution to the BMP is described by the set of connections that are used in the aircraft flight routes. Thus, a measure of the distance between solutions is the number of different connections that are used.

To implement the trust region in the SDAMRP-RR the following set of constraints are added to the BMP,

$$\sum_{b\in B}\sum_{p\in P_b} e_{ijp}y_p + \epsilon^+_{ij} - \epsilon^-_{ij} = \sum_{b\in B}\sum_{p\in P_b} e_{ijp}\bar{y}_{(n-1)p} \quad \forall (i,j) \in C, \tag{6.47}$$

This set of constraints is similar to the linking constraints (6.32) in the PBSP-s, as such the counting variables ϵ^+_{ij} and ϵ^-_{ij} are defined in a very similar manner. To minimise the number of connection changes between solutions of consecutive iterations, the term $\sum_{(i,j)\in C} g^{TR}\epsilon^-_{ij}$ is added to the objective function of the BMP. The value of the weight g^{TR} in the objective function affects the number of connection changes which are made.

While the trust region is used to restrict the feasible region of the master problem, it is important to ensure that the resulting objective function value is less than the best found upper bound. This is achieved by including a constraint to impose an upper bound on the objective function value. Such a constraint for the BMP is given by,

$$\sum_{p\in P} c_p y_p + \sum_{b\in B}\left\{F_b(0) - \sum_{i=1}^{R_b}\Delta^i_b Z^i_b\right\} + \sum_{s\in S} w^s\varphi^s \le \hat{\Phi}^{UB}. \tag{6.48}$$

The addition of the trust region and the upper bound on the objective value introduces a two-stage solution process for the BMP. The first stage solves the BMP as described in Section 6.3.1 to identify the current lower bound on the objective of the original problem. The second stage adds the constraints (6.47) and (6.48) to the BMP, which is then solved to identify a solution within a neighbourhood of the solution from the previous iteration. While it is only necessary to solve the second-stage of this process to generate optimality cuts, the lower bound identified in the first stage is used in the stopping condition described by (6.43). In addition, this lower bound multiplied by a tolerance $\xi \geq 1$ can be used interchangeably with $\hat{\Phi}^{UB}$ in constraint (6.48) provided $\xi\Phi < \hat{\Phi}^{UB}$. By replacing the right-hand side of constraint (6.48) with $\min\{\xi\Phi, \hat{\Phi}^{UB}\}$, the solution to the second stage will remain close to the current lower bound and the solution from the previous iteration solution.

This two-stage solution process for the BMP is implemented to solve the SDAMRP-RR. Our implementation involves the process described above, however it is possible that the solution to the original problem achieved by this approach overestimates the optimal solution. Once the optimal solution is found by this process, the Benders' decomposition solution algorithm continues, however only the first stage is used when solving the BMP. The solution process then terminates when either stopping condition presented in Section 6.3.1 is satisfied.

Solving the BMP using a trust region identifies a suboptimal solution at each iteration of the Benders' decomposition solution process. While the optimality of the BMP affects the cuts generated in the evaluation stage, it is demonstrated by Geoffrion and Graves [42], Côté and Laughton [28] and Rei *et al.* [71] that suboptimal solutions to the BMP does not affect the finite convergence of the solution process. In particular, Geoffrion and Graves [42] describes a solution process that only identifies feasible suboptimal solutions to the BMP at each iteration. The termination of the solution process in [42] occurs when no further feasible solutions to the BMP exist with an objective value less than the best found upper bound. The algorithm described by Geoffrion and Graves [42] is similar to our implementation, however the stopping conditions presented in Section 6.3.1 significantly alters the solution approach. Our implementation of Benders' decomposition terminates with the optimal solution prior to all feasible solutions being identified, greatly improving the solution runtimes.

The results presented in the following section will demonstrate that the trust region method is a necessary enhancement to improve the convergence of the Benders' decomposition solution process. By contrast, the solution approach for the RRTAP did not require a trust region, with acceptable solution runtimes achieved through the implementation of the Magnanti-Wong method alone. The major differences between the RRTAP and the SDAMRP-RR affecting runtimes are the size of the data sets and number of scenarios used in the evaluation stage. As such, the SDAMRP-RR describes a much larger and more complex problem requiring the use of additional enhancement techniques to reduce the solution runtimes.

6.4 Computational Results

The SDAMRP attempts to provide an adequate number of maintenance routes departing from each overnight airport to satisfy the maintenance requirements of all aircraft. The ability of the SDAMRP to achieve this desired goal is evaluated by comparing the resulting solution with that of an aircraft routing problem solved without any maintenance considerations. To provide this set of benchmark results, an aircraft routing problem is formulated from the SDAMRP with the maintenance counting constraints (6.4) eliminated and the term $\sum_{b \in B} F_b(\psi_b)$ removed from the objective. Both the SDAMRP and the benchmark problem is solved for four different flight schedules and the number of maintenance misalignments from each is compared.

The SDAMRP-RR is presented as an extension to the SDAMRP, attempting to improve recoverability with no impact to the planned maintenance solution. The improved recoverability is evaluated by simulating the recovery of disruption scenarios for the SDAMRP and SDAMRP-RR solutions and comparing the resulting recovery costs. A selection of the flight schedules used to benchmark the SDAMRP are used in this analysis of the improved recoverability.

6.4.1 Description of schedule data and parameters

Four different flight schedules are used in this chapter to demonstrate the versatility of the maintenance scheduling approach of the SDAMRP. Table 6.4 provides the details of each of the schedules which vary in the number of flights, airports and aircraft. The key features of each flight schedule is the set of airports where aircraft flight routes terminate, called overnight bases, and the subset of these where maintenance can be performed. Since all aircraft flight routes originate and terminate at overnight bases, the SDAMRP attempts to provide a sufficient number of routes from each of these bases that terminate at maintenance stations.

Traditionally the aircraft routing problem is solved following the fleet assignment, which partitions the schedule into sets of flights, one for each fleet type. Since the aircraft assigned

	F267_A49	F578_A153	F1165_A289	F3370_A526
Flights	267	578	1165	3370
Aircraft	49	153	289	526
Airports	20	50	97	73
Overnight(Maintenance) bases	12(1)	41(2)	67(5)	73(10)

Table 6.4: Flight schedule details.

to a single partition of the schedule are all of the same type, the traditional formulation of the aircraft routing problem describes a feasibility problem. Solving the SDAMRP, and similarly the SDAMRP-RR, as a feasibility problem is accomplished by setting the cost of each flight route to zero, i.e. $c_p = 0, \forall b \in B, \forall p \in P_b$. Setting the costs in this way is equivalent to having a zero cost on each arc in the connection network, hence this formulation will be described as solving the SDAMRP with a zero connection cost function (ZCF).

Parameters for the SDAMRP-RR

To thoroughly review the SDAMRP-RR, experiments are performed by also solving the planning stage using a proxy robust connection cost function introduced by Grönkvist [45] (GCF). This proxy robust connection cost function is implemented in the RRTAP presented in Chapter 5. To provide a direct comparison with the results presented in Section 5.3, the identical parameter settings for the GCF are used to solve the SDAMRP-RR. A contribution of this chapter is a more detailed review of the application of recoverable robustness to the aircraft routing problem, discussing the impact of the connection cost functions on the recoverability improvements and solution runtimes.

The definition of parameters used for the SDAMRP-RR are identical to that given in Section 5.3.1 for the RRTAP. In particular, the swap cost g^{SW} is used to penalise any different connections between the planning and recovery solutions. The experiments presented in this section solve the SDAMRP-RR with a range of swap costs to broadly analyse the recoverability improvement, i.e. $g^{SW} \in \{2000, 5000, 7500\}$. The magnitude of this parameter directly affects the flexibility of the recovery solution by minimising the number of connection changes that are made. The cost of a flight delay is based upon the EUROCONTROL report by Cook and Tanner [25] and is set at \$100AUD. Similar to the RRTAP, it is assumed that the load factor for the entire schedule is 75%, so the cost of delaying a flight by one minute is equal to

\$100AUD $\times$ 0.75 = \$75AUD. Finally a *loss rate* g^{LR} is used to quantitatively describe passenger dissatisfaction. An indepth analysis of the solution to the SDAMRP-RR achieved using different loss rates is performed by presenting the results with $g^{LR} \in \{0.5, 1.0, 1.5, 2.0\}$. Similar to the swap cost parameter, the flexibility of the recovery solution decreases as g^{LR} increases.

A selection of the flight schedules presented in Table 6.4 are used to evaluate the SDAMRP-RR, namely the F267_A49 and F578_A153 schedules. These schedules are selected to provide an example of the potential recoverability improvements that can be achieved by the application of recoverable robustness. In Chapter 5, the RRTAP is evaluated on a schedule that contains 53 flights operated by 10 aircraft. The evaluation of the SDAMRP-RR extends upon the analysis presented in the previous chapter by assessing the applicability of the recoverable robustness framework to larger flight schedules.

Evaluation scenarios for the SDAMRP-RR

The solution process of the SDAMRP-RR is separated into planning and evaluation stages. The evaluation stage involves assessing the recoverability of the current planning solution by solving recovery problems for a set of disruption scenarios. The disruption type selected for each scenario in the evaluation stage is an airport closure. This disruption type is selected for the SDAMRP-RR since airport closures generally require significant intervention by the airline during recovery. While there are many different types of disruptions that can occur, it is believed that airport closures provide a good example of the benefits that can be achieved with the recoverable robustness framework. In practice, it is trivial to implement different disruption types in the evaluation stage to match the desired outcomes of each airline.

The airport closure scenarios are generated from the original flight schedule to represent closures at different airports with a range of starting times and durations. The airports where at least 5% of all arrivals or departures occur are selected as the affected airports for the closure scenarios. For the F267_A49 data set there are six airports that satisfy this criteria and for the F578_A153 there are four. To assess the impact of disruptions on maintenance planning, the airports where aircraft maintenance can be performed are included in the set of affected airports for both data sets.

To provide a broad range of scenarios at each airport the earliest and latest start times for the closures are 6:30am and 5:00pm local time respectively, and the durations range between 1 and 5 hours. The set of scenarios is constructed such that there are an equal number of closure start times and durations. Using the F267_A49 data set as an example, if the evaluation stage is solved with 24 scenarios, 4 are generated for each airport by using 2 different starting times and durations. While each of the scenarios are not equally likely, we wish to improve the recoverability across the complete set of scenarios, as such a uniform distribution is used.

6.4.2 Analysis of maintenance misalignment

As part of regular maintenance planning, a fraction of all aircraft require maintenance at the end of each day. In this chapter it is assumed that aircraft require maintenance once every six days, therefore 1/6 of aircraft at each overnight airport will require maintenance the following day. By this assumption, adequate maintenance planning requires at least 1/6 of all aircraft routes originating from each airport to terminate at a maintenance station at the end of the day.

To demonstrate the benefits of the SDAMRP, the solution is compared against an aircraft routing problem that is solved without considering maintenance planning. The solution approach used as a benchmark for this analysis assumes that the aircraft routing and maintenance planning are performed separately. Given the aircraft routing solution, modifications are made *a posteriori* to ensure all aircraft receive maintenance as required. This is the airline planning approach presented by Feo and Bard [37], Gopalan and Talluri [44], Sriram and Haghani [80], and Lapp and Cohn [54]. The solution to the SDAMRP attempts to identify an aircraft routing solution that requires little manual modification to satisfy all maintenance requirements. As such, we believe that a comparison against the standard aircraft routing solution is acceptable for this analysis.

The improved maintenance planning by the SDAMRP is assessed by calculating the total number of maintenance misalignments across all airports. The number of maintenance misalignments at overnight airport b is calculated by

$$MM_b = \max\left\{0, \frac{R_b}{6} - \sum_{i=1}^{R_b} Z_b^i\right\}, \tag{6.49}$$

and the total number of maintenance misalignments is given by $MM_{Tot} = \sum_{b\in B} MM_b$. This value is calculated for both the standard aircraft routing problem and the SDAMRP. The maintenance misalignment results are presented in Table 6.5.

The results in Table 6.5 demonstrate that the SDAMRP significantly reduces the number of maintenance misalignments compared to the standard aircraft routing solution. The improve-

Misalignment	F267_A49	F578_A153	F1165_A289	F3370_A526
Aircraft Routing	2.67	3.5	6.67	3.0
SDAMRP	1.17	0.5	0.0	0.0
Improvement (%)	56.25	85.71	100.0	100.0
ARP - Runtime (sec)	1.61	6.4	155.82	67675.0
SDAMRP - Runtime (sec)	7.27	6.97	168.84	65087.0

Table 6.5: Maintenance misalignments and solution runtimes for different flight schedules.

ment in the number of maintenance misalignments across all data sets is at least 56.25%, and for two of the flight schedules the number of misalignments is reduced to zero. This result is very important since the SDAMRP is solved at the equivalent stage as the standard aircraft routing problem, so any maintenance misalignments that exist requires an additional maintenance planning problem to be solved. The results demonstrate that is possible to completely eliminate the need for any further modification of the aircraft routing solution by solving the SDAMRP.

6.4.3 Analysis of planning recoverability

The application of recoverable robustness to the SDAMRP is an attempt to reduce the impact of disruptions on daily maintenance plans. The optimisation process of the SDAMRP-RR involves the evaluation of the SDAMRP by solving a recovery problem for a set of disruption scenarios, generated using the method described in Section 6.4.1. A total of 150 disruption scenarios are used in the evaluation stage of the SDAMRP-RR for both data sets. The feedback process in the recoverable robustness framework reduces the expected recovery costs of the SDAMRP while still providing a reduced number of maintenance misalignments in the planned solution. Most importantly, the recovery problem solved in the evaluation stage ensures that maintenance is still performed on maintenance critical aircraft at the end of the day.

To assess the recoverability improvement given by the SDAMRP-RR, the solutions to both the SDAMRP and SDAMRP-RR are evaluated against a set of disruption scenarios. To provide a fair comparison between the solutions of the SDAMRP and SDAMRP-RR, a set of 300 disruptions scenarios are generated for the recoverability evaluation. Since this alternative set of scenarios is different to the 150 used in the evaluation stage of the SDAMRP-RR, any advantage to the recoverable robust solution in this analysis is removed.

The recoverability of the solutions to the SDAMRP and SDAMRP-RR is given by the average cost of the recovery solutions over the set of 300 evaluation scenarios. The solution to each model, given by $\bar{\mathbf{y}}$, is used as an input to a recovery problem defined by the PBSP-s, and the cost of recovery for scenario s is given by the objective function value $\mu_s(\bar{\mathbf{y}})$. Therefore, we define the recoverability of the aircraft routing solution from the SDAMRP as $Rec = \sum_{s \in S} w^s \mu_s(\bar{\mathbf{y}})$. We define the recoverability achieved using the solution to the SDAMRP-RR in the same manner, which is labelled as Rec_{RR}.

There are many features and parameters of the SDAMRP-RR that affect the efficacy of the recoverable robustness framework and solution runtimes. The key features affecting the possible improvement in the recoverability are i) the value of the swap cost parameter g^{SW}, ii) the value of the loss rate parameter g^{LR}, and iii) the number of scenarios used in the evaluation stage. Both the swap cost g^{SW} and the loss rate g^{LR} are closely related with a trade-off between the number of aircraft swaps and flight cancellations observed in the recovery solutions. As such, the results for the SDAMRP-RR using different values for g^{SW} and g^{LR} will be presented in the following section. Solving the recovery problem is a time consuming feature of the recoverable robustness framework, consequently the number of scenarios that are used in the evaluation stage of the SDAMRP-RR has a great effect on the solution runtimes. The result of increasing the number of scenarios will be presented in following sections. The results analyse the effects that each of these features has on the number of maintenance misalignments, recoverability improvement and solution runtimes.

Effect of swap cost and loss rate parameters

The swap cost and loss rates parameters are airline specific values that greatly affect recoverability improvement and solution runtimes of the SDAMRP-RR. Since the SDAMRP-RR is solved as a planning problem the runtime required to find the optimal solution is not of critical importance. However, it is possible to reduce the solution runtimes for the SDAMRP-RR through the intelligent selection of these parameters. The values of the swap cost and loss rate parameters also affect the flexibility of the recovery solution, which directly impacts the potential recoverability improvement.

The improvement in the recoverability of the SDAMRP from applying recoverable robustness is given by the relative difference between Rec and Rec_{RR}. This is calculated by $Improve = |Rec - Rec_{RR}| / \min\{Rec, Rec_{RR}\}$ and is given as a percentage in Figure 6.1. The results presented in this figure demonstrate the recoverability improvement achieved using dif-

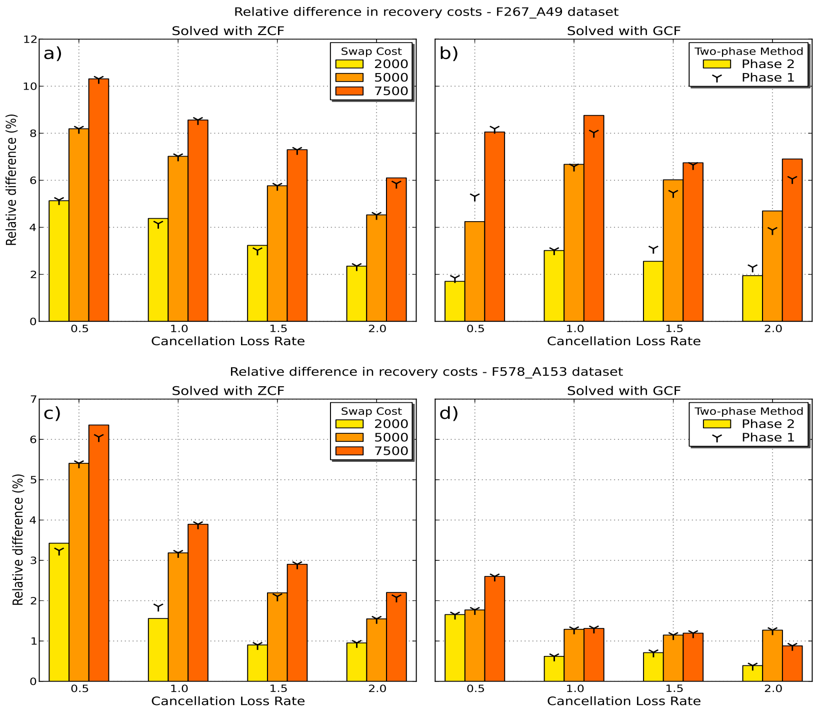

Figure 6.1: Relative difference in the recoverability of the SDAMRP by applying recoverable robustness. 150 scenarios are used in the evaluation stage of the SDAMRP-RR.

ferent swap costs, loss rates and connection cost functions. It is clear from this figure that the greatest improvement in recoverability is achieved with a swap cost $g^{SW} = 7500$ regardless of the value of g^{LR} with only one exception. This is a feature of the recoverable robustness framework since the solution to the SDAMRP-RR attempts to minimise the number of changes from the planned solution. As the value of g^{SW} increases, the effect of this parameter forces the solution to the SDAMRP-RR to require fewer changes during recovery compared to the SDAMRP. Therefore, total number of swaps performed in the evaluation of the SDAMRP-RR will be much less than that required for the SDAMRP. Consequently, the weighted recovery cost from the evaluation of the SDAMRP increases as the swap cost increases, where the converse is true for the SDAMRP-RR. This explains the behaviour presented in Figure 6.1 where the recoverability improvement increases as the swap cost increases.

The flexibility in the recovery problem has a significant effect on the efficacy of the recoverable robustness framework. The values of g^{LR} have a direct correlation with the flexibility of the recovery problem, whereby larger values reduce the number of flight cancellations that are made. Figure 6.1 demonstrates that as the value of g^{LR} increases, the reduced flexibility in the recovery problem negatively affects the improvement in recoverability achieved by the SDAMRP-RR. This effect of reduced flexibility is observed for both of the data sets and connection cost functions.

Flexibility in the planning stage is also observed to affect the potential recoverability improvement of the SDAMRP-RR. In Figure 6.1, a greater recoverability improvement is achieved for both data sets using the ZCF. Since the ZCF does not impose a cost on connections in the network, the construction of aircraft flight routes is only affected by the maintenance misalignment constraints and the added Benders' cuts. By contrast, the GCF favours the use of specific connections in the construction of flight routes, reducing the efficacy of the Benders' cuts. Therefore, the Benders' cuts have a greater impact on the flight route construction when the ZCF is used compared to the GCF, hence permitting a greater recoverability improvement for the SDAMRP-RR.

In Section 6.3.1, the implementation of the two-phase method for solving the SDAMRP-RR is described. The first phase of this method solves the LP relaxation of the SDAMRP-RR to optimality to provide a lower bound on the optimal integer solution. As demonstrated in Section 5.3.2, using all the cuts generated in Phase 1 of the two-phase method and solving the BMP to integer optimality, the resulting solution is a very good approximation to the optimal solution found at the end of Phase 2. This is also demonstrated in Figure 6.1 with the stars describing the improvement in recoverability given by the solution to the SDAMRP-RR at the end of Phase 1. It is important to note that the improvement in recoverability reported in Figure 6.1 does not include the cost of maintenance misalignments. Consequently, it is possible for the solution found at the end of Phase 1 to outperform the solution found at the end of Phase 2 in terms of recoverability. It is clear that the Phase 1 solution is a good approximation of the solution that is achieved at the end of Phase 2.

The runtimes for each of the cases presented in Figure 6.1 is given in Figure 6.2. A maximum runtime for the SDAMRP-RR is set at 10 hours (36000 seconds), which is exceeded for most of the cases presented. When the maximum runtime is exceeded, the method used to find the best integer solution to the SDAMRP-RR depends on the current phase of the solution process. If the algorithm terminates during Phase 1, the solution to the SDAMRP-RR is found by solving the BMP to integral optimality with the current set of generated cuts. Otherwise, if the algorithm terminates during Phase 2, the solution to the SDAMRP-RR is given by the best integer solution to the BMP found during the solution process.

Since the runtimes for the SDAMRP-RR are very high, identifying a good upper bound at

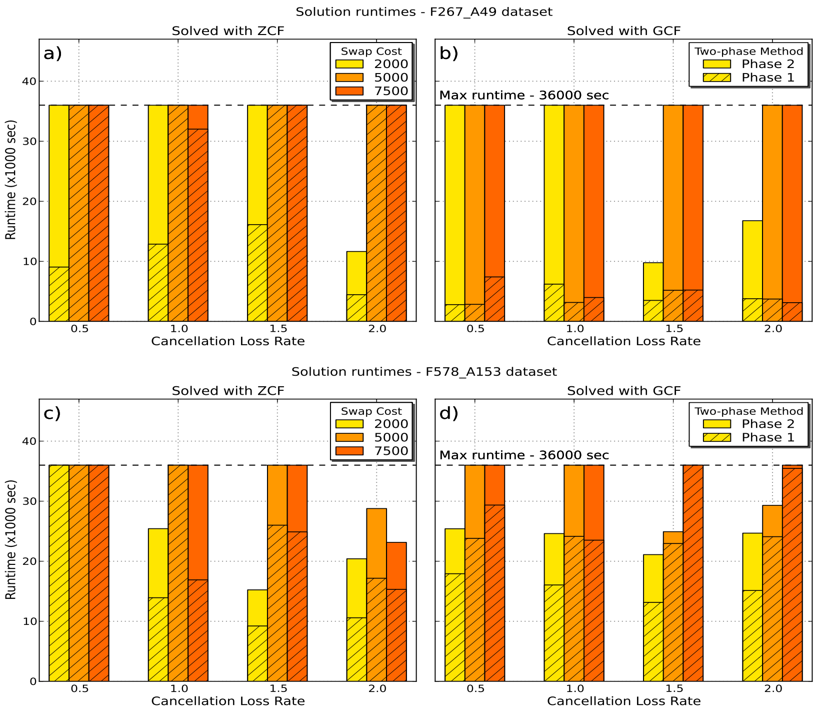

Figure 6.2: Runtimes of the SDAMRP-RR. 150 scenarios in the evaluation stage of the SDAMRP-RR.

the end of Phase 1 for the SDAMRP-RR is of great importance. Figure 6.2b is a prime example of this, where the runtimes for Phase 1 are significantly less than the runtimes for Phase 2. By comparing the results in Figure 6.1b with the runtimes in Figure 6.2b, it is clear that the additional time required to identify the integer optimal solution does not have a significant effect on the improvement in recoverability.

As a final note, the runtimes for the SDAMRP-RR solved with the ZCF, Figures 6.2a and 6.2c, are generally greater than when the GCF is used, Figures 6.2b and 6.2d. The use of the ZCF in the planning stage of the SDAMRP-RR increases the degeneracy of the problem which greatly affects the convergence of the Benders' decomposition solution process. One cause of degeneracy is symmetry in the BMP, which requires a greater number of cuts from the PBSP-s to eliminate identical solutions. The GCF helps to eliminate the symmetry in the BMP through the cost of the aircraft flight routes, also reducing the number of optimal solutions. While the runtimes required for the SDAMRP-RR using the ZCF are much greater than when the GCF is used, Figure 6.1 demonstrates that the former achieves a better recoverability improvement. This presents a trade-off between solution runtime and quality, arising from the flexibility of the BMP.

Number of evaluation scenarios

This section examines the effect that increasing the number of scenarios has on the improvement in recoverability. In the previous section, the solution to the SDAMRP-RR is solved with 150 scenarios in the evaluation stage. To further analyse the recoverable robustness framework, experiments are performed with 250, 350 and 500 scenarios in the evaluation stage of the SDAMRP-RR. The set of scenarios in each of these experiments are generated using the process described in Section 6.4.1. Similar to the experiments in the previous section, the solutions to the SDAMRP and SDAMRP-RR are evaluated against a set of 300 disruption scenarios.

Two different results are observed from the different data sets as the number of evaluation scenarios increases. Firstly, Figure 6.3a demonstrates that the number of scenarios used in the evaluation stage of the SDAMRP-RR does not greatly affect the recoverability improvement for the F267_A49 data set. In particular, the variation in the recoverability improvement caused by an increase in the number of scenarios is much less than that achieved using different swap penalty values. This indicates that it is more effective to intelligently select the values for the parameters g^{SW} and g^{LR} than to increase the number of scenarios.

For the F578_A153 data set, as the number of scenarios increases a much greater variation in the recoverability improvement is observed in Figure 6.3b. In particular, the increase in the number of evaluation scenarios causes the recoverability improvement using the F578_A153 data set to decrease to very low levels. For example, using 500 scenarios in the evaluation stage with $g^{SW} = 2000$, the SDAMRP-RR is solved to find an aircraft routing solution that has a 0.36%

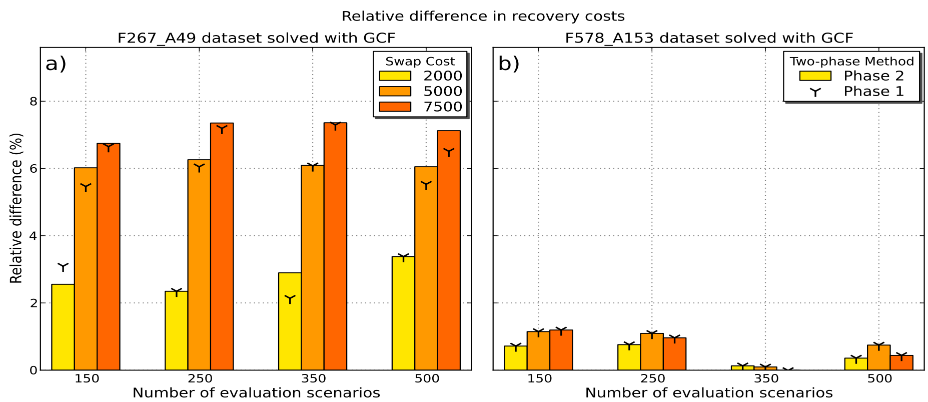

Figure 6.3: Relative difference in the recoverability of the SDAMRP by applying recoverable robustness. The SDAMRP-RR is solved with different sets of scenarios in the evaluation stage. The cancellation loss rate is set at $g^{LR} = 1.5$.

	Master iterations				Optimality gap			
	Evaluation scenarios				Evaluation scenarios			
Swap cost	150	250	350	500	150	250	350	500
2000	7	6	2	5	0.96%	0.47%	26.92%	1.71%
5000	9	9	2	4	0.92%	0.81%	30.3%	6.64%
7500	9	7	2	2	1.25%	1.42%	25.32%	35.27%

Table 6.6: Number of master iterations and the optimality gap of the SDAMRP-RR solved for the F578_A153 data set using different sets of evaluation scenarios.

improvement in recoverability. In addition, using 350 evaluation scenarios the recoverability improvement is almost negligible. Given that the runtimes required to achieve this improvement have been capped at 10 hours, this result is not encouraging for practical application. It is clear for the F578_A153 data set that the selection of the g^{SW} and g^{LR} parameter values is much more effective in improving the solution quality and runtimes compared to increasing the number of evaluation scenarios.

The poor recoverability improvement for the F578_A153 data set is the result of the large runtimes required in the evaluation stage of the SDAMRP-RR. The maximum runtime is exceeded for all but four of the experiments presented in Table 6.6, which is indicated by an optimality gap greater than 1%. It is important to note that the optimality gap reported in Table 6.6 is calculated using equation (6.43) with an integer first-stage and continuous second-stage solutions.

In Figure 6.3b, the worst recoverability improvement is observed when 350 scenarios are used in the evaluation stage. By comparing Figure 6.3b with Table 6.6, it is clear that these experiments also display the worst runtime performance. Only 2 master problem iterations are performed for the experiments using each of the swap cost values examined. This demonstrates that the runtime spent solving the evaluation scenarios has a significant impact on the recoverability improvement achieved by the SDAMRP-RR. Also, large optimality gaps are observed for all experiments using 350 scenarios in the evaluation stage, explaining the poor recoverability improvement presented in Figure 6.3b.

6.4.4 Maintenance misalignment in the SDAMRP-RR

The results in Section 6.4.2 demonstrate that the SDAMRP significantly reduces the number of maintenance misalignments compared to a standard aircraft routing solution. While the application of recoverable robustness to the SDAMRP attempts to improve the recoverability of the planned solution, this should not come at the expense of maintenance planning. In stochastic programming problems, the solution to the first-stage variables without considering any uncertainty information provides a lower bound on the optimal stochastic solution. As such, there is a cost associated with considering the second-stage uncertainty information during the optimisation process. This is observed when comparing the solutions to the SDAMRP and SDAMRP-RR, where a lower bound on the number of maintenance misalignments is given by the solution to the SDAMRP. Figure 6.4 presents a comparison of the number of maintenance misalignments for the SDAMRP, SDAMRP-RR and the standard aircraft routing problem. The *dashed* line represents the SDAMRP solution, which is the best possible solution, and the *dot-dashed* line is the solution to the standard aircraft routing problem.

The results in Figure 6.4 for the SDAMRP-RR are all within the gap between the SDAMRP and aircraft routing solution, with most solutions close to the lower bound. While the SDAMRP-RR is a conservative planning approach, the results demonstrate that this solution does not greatly affect the reduction in maintenance misalignments achieved by the SDAMRP. It is in-

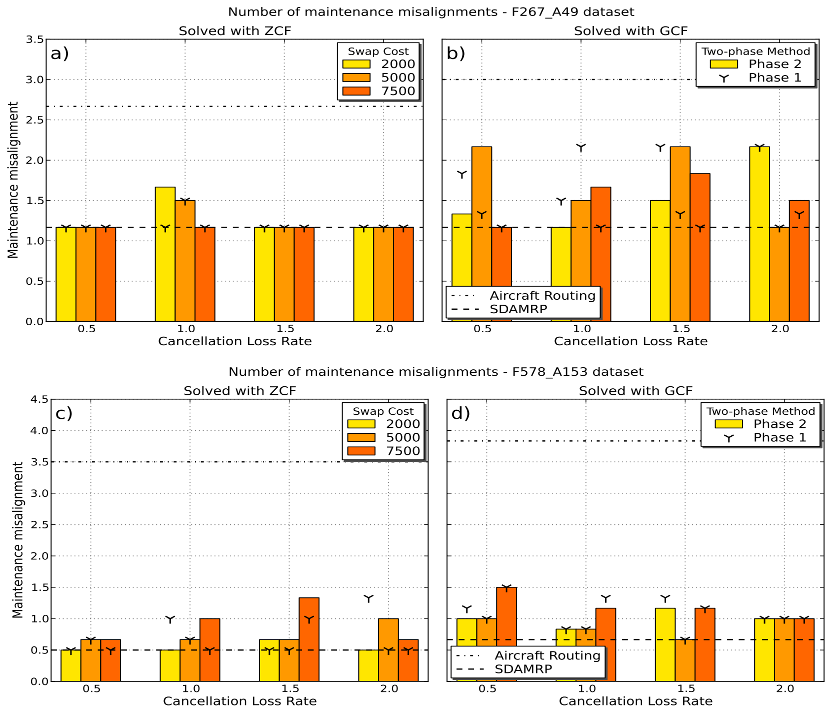

Figure 6.4: Assessing the impact of recoverable robustness on maintenance misalignments. 150 scenarios are used in the evaluation stage of the SDAMRP-RR.

teresting to note that the number of maintenance misalignments for the SDAMRP-RR when solved with the ZCF is equal to the solution of the SDAMRP for a large number of cases. This can be explained by the increased flexibility that is provided by not artificially imposing flight string costs based on connection lengths. Since the aircraft routing problem is highly degenerate, there are large set of solutions that can achieve the same maintenance misalignment result. In this case, the application of recoverable robustness to the SDAMRP helps to select the solution from this set that provides the best recoverability.

It is observed in Figure 6.4 that the greatest reduction in maintenance misalignment is achieved by solving the SDAMRP-RR with the F578_A153 data set. This is expected since the same result is also observed in Table 6.5 for the SDAMRP. The most interesting observation is that solving the SDAMRP-RR for the F578_A153 data set displays the least variation from the SDAMRP solution across all parameter values. This is an important result since it demonstrates that the application of recoverable robustness to larger data sets does not greatly affect the maintenance planning of the SDAMRP. It can be concluded that the solution to the SDAMRP-RR improves recoverability while still achieving a significant reduction in the number of maintenance misalignments.

6.4.5 Enhancement techniques for the SDAMRP-RR

The application of Benders' decomposition to solve the SDAMRP-RR is described in Section 6.3.1. While Benders' decomposition greatly improves the tractability of the SDAMRP-RR, the standard implementation of this solution technique exhibits poor convergence to the optimal solution. A number of techniques are introduced in Sections 6.3.1 and 6.3.3 to improve the convergence of the Benders' decomposition solution process and reduce solution runtimes. In this section the performance of the parallel computing and trust region enhancements are assessed, demonstrating the improved convergence achieved by their implementation. The experiments solve the SDAMRP-RR for the F267_A49 and F578_A153 data sets using the GCF and 50 scenarios in the evaluation stage.

The evaluation of the different enhancement techniques for the F267_A49 and F578_A153 data sets is presented in Table 6.7. The optimality gap reported in these tables is given by equation (6.43) with the lower and upper bound calculated with integer optimal solutions to the BMP and PBSP-s. Since the stopping conditions presented in Section 6.3.1 involves the

Swap Penalty	All Enhancements	Parallel Only	Trust Region Only	No Enhancements
F267_A49				
2000	1.26%	1.64%	1.34%	3.75%
5000	1.29%	6.88%	2.44%	11.92%
7500	2.82%	11.68%	2.84%	19.06%
F578_A153				
2000	**2.75%**	**3.09%**	2.99%	5.14%
5000	**6.22%**	7.29%	11.8%	10.71%
7500	**9.06%**	11.4%	21.23%	17.51%

Table 6.7: Optimality gap of the SDAMRP-RR solved for the F267_A49 and F578_A153 data sets using different enhancement techniques. The SDAMRP-RR is solved with the GCF and 50 evaluation scenarios. Bold entries are the experiments that terminate within the maximum runtime.

calculation of the optimality gap using continuous second stage solutions, it is likely that the results reported in Table 6.7 are greater than 1%.

It is clear that the use of all enhancements is necessary to achieve the best convergence for the Benders' decomposition solution process for both data sets. This is not surprising since the two enhancement techniques focus on two separate stages of the solution process. However, it is interesting to note that for the F267_A49 data set the trust region method achieves a better convergence compared to using a parallel computing environment. This demonstrates the strength of the trust region method to focus the solutions of the Benders' master problem between consecutive iterations and improving the efficacy of the Benders' optimality cuts.

A different conclusion can be drawn from the results for the F578_A153 data set, providing a richer analysis of the enhancement techniques. It is observed that using parallel computing achieves a greater improvement in the convergence of the Benders' decomposition solution approach compared to the trust region method. Since the F578_A153 data set is much larger than the F267_A49 data set, a greater proportion of the solution runtime per iteration is required for the evaluation stage. Therefore, the benefits from distributing the recovery problems to a number of parallel compute threads will be more pronounced for the F578_A153 data set. It is expected that as the problem size and the number of scenarios in the evaluation stage increases, the use of parallel computing will have a greater impact on the convergence of the Benders' decomposition solution process.

6.5 Conclusions

This chapter presents two major contributions to airline planning problems, the development of the SDAMRP and an analysis of the recoverable robustness framework. The SDAMRP describes a single day maintenance routing problem that introduces a modelling technique to effectively provide adequate maintenance coverage for the entire fleet. The modelling approach used for the SDAMRP is a novel approach that penalises the number of maintenance misalignments that occur at each airport. This model is extended with the application of the recoverable robustness framework to improve the recoverability of the maintenance planning solution. The application of recoverable robustness protects the maintenance planning solution from perturbations during the day of operations while reducing the expected operational costs. The analysis of the SDAMRP-RR contributes to the recoverable robustness framework demonstrating the potential recoverability improvements that can be achieved for larger data sets.

The formulation of the SDAMRP in Section 6.1 describes a mathematical programming model with linear constraints and a non-linear objective function. The non-linear objective function arises from the introduction of a penalty for any maintenance misalignments in the planned solution. The non-linearity of the objective function is addressed by a unique reformulation described in Section 6.1.4. This reformulation provides an equivalent mixed integer programming model which can be solved using a standard column generation approach.

A significant reduction in the number of maintenance misalignments is achieved by the solution to the SDAMRP compared to a standard aircraft routing solution. A minimum improvement of 56.25% is achieved in the experiments performed on 4 flight schedules of varying sizes. A conclusion that can be drawn from the results is that the SDAMRP is very effective in reducing the number of maintenance misalignments for all of the examined flight schedules.

The SDAMRP-RR is presented as a large-scale optimisation problem requiring the use of Benders' decomposition and column generation to improve the problem tractability. To address the poor convergence of the Benders' decomposition solution process additional enhancement techniques, including the Magnanti-Wong method [60], the two-phase method, parallel computing and a trust region method, are applied to solve the SDAMRP-RR. This combination of enhancement techniques is demonstrated to significantly improve the problem convergence. In addition, the use of parallel computing is demonstrated to provide a greater improvement in the convergence of the Benders' decomposition solution approach for larger data sets.

The application of recoverable robustness presented by the SDAMRP-RR achieves a considerable improvement in the recoverability of the SDAMRP. Solving the SDAMRP-RR for large flight schedules extends upon the results presented in Chapter 5, further demonstrating the potential of the recoverable robustness framework. A contribution of this chapter is the evaluation of the recoverability improvement achieved using different connection cost functions in the planning stage. The results demonstrated that flexibility in the planning stage is directly proportional to the recoverability improvement. In addition, the SDAMRP-RR is solved with various numbers of scenarios in the evaluation stage, further extending the analysis in Chapter 5. The results from this analysis conclude that an acceptable improvement in recoverability can be achieved with a modest number of scenarios in the evaluation stage.

Solving the single day problem in isolation for each day of flying is a significant advantage of the SDAMRP. However there are potential benefits from extending the single day maintenance routing problem to construct maintenance plans for longer time frames or consider other maintenance check types. Another possible extension of this problem is the integration with a maintenance location planning problem to reduce costs by eliminating unnecessary maintenance bases.

The recoverable robustness framework investigated in this chapter is greatly affected by the problem size and number of evaluation scenarios. A potential extension of the recoverable robustness framework is the development of an integrated recoverable robust aircraft routing and crew duty problem. As demonstrated in Section 6.4.3, a limitation of the framework is the runtime required to solve the evaluation stage. To properly investigate this possible extension, it is important to develop an efficient solution approach for the integrated airline recovery problem. Chapter 7 introduces the column-and-row generation solution approach to achieve this goal with Chapters 8 and 9 demonstrating the application of this solution approach to integrated airline recovery problems.

Chapter 7

Column-and-Row Generation

Column-and-row generation is a solution approach that extends standard column generation to reduce problem complexity and solution runtimes. Column generation, described in Section 4.1, relies on the assumption that only a small subset of all variables are basic in the optimal solution. Using this assumption, a smaller, more compact linear program is formed, which is solved by dynamically generating variables. This assumption is also the basis for column-and-row generation coupled with the understanding that the basic variables have non-zero coefficients in a small subset of all rows. Similar to column generation, a restricted linear program is formed through the elimination of columns, and in addition rows are also eliminated to further reduce the problem size. A row generation procedure updates the set of rows in the master problem with the expectation of improving the objective function value by expanding the set of permissible columns. With the elimination of both columns *and* rows, a small initial master problem is formed by this solution approach. This problem then grows both horizontally and vertically with the addition of columns and rows that are identified as necessary to find the optimal solution.

A key feature of column-and-row generation is the dynamic generation of structural constraints for the master problem. As such, problems must display a decomposable structure that permits the elimination of constraints. There are many practical applications that display this particular problem structure, for example the multi-stage cutting stock problem and integrated airline planning and recovery problems. In this thesis, the application of column-and-row generation to solve the integrated airline recovery problem will be discussed.

This chapter will provide an overview of the column-and-row generation solution approach and a general framework for its implementation. Section 7.1 will discuss the current applications of column-and-row generation, including the various generic frameworks that have been developed. The features of column-and-row generation fundamental to the framework implemented in this thesis will be presented in Section 7.3. This will be followed by a general algorithm for the column-and-row generation solution approach in Section 7.4. The work that is presented in this chapter appears as part of a paper submitted to Transportation Science by Maher [61].

7.1 Related Literature

Constraint generation in the form of cutting planes is widely applied to solve integer programming problems. A modern development of constraint generation is column-and-row generation, involving the dynamic generation of variables and structural constraints. One of the earliest modern applications of column-and-row generation is presented by Zak [100] to solve the multi-stage cutting stock problem. The multi-stage cutting stock problem is an extension of the classical single-stage cutting stock problem, which is successfully solved using column generation [43]. Each stage in the multi-stage cutting stock problem involves a number cutting size decisions which are then passed to subsequent stages. The cutting size decisions made in each intermediate stage are modelled as constraints to restrict the decisions to later stages. As such, this problem is conveniently solved by column-and-row generation where constraints are initially eliminated to reduce the number of intermediate cutting decisions that can be made. The column-and-row generation approach presented in [100] describes three different subproblems, two identify columns to add based on the current set of rows, and the third identifies any additional rows. Column-and-row generation is demonstrated in [100] to be an efficient approach for the multi-stage cutting stock problem, however assumptions made by the author limit the potential improvements.

An interesting application of column-and-row generation is presented by Avella *et al.* [9] for solving the time-constrained routing problem. This problem considers a large number of tourist sites to identify individual itineraries that maximise personal utility. Column-and-row generation is employed to formulate a smaller problem that contains only a subset of all possible sites. A column generation procedure is executed to identify the utility-maximising itineraries given the current set of tourist sites, and the row generation procedure updates the sites that are considered in the model. Avella *et al.* [9] presents a typical example of the types of problems that column-and-row generation can be applied to and the relevant solution methods.

The p-Median problem is presented by Avella *et al.* [10] as an example of column-and-row generation within a branch-cut-and-price framework. Since all variables and constraints are explicitly known *a priori*, variations on the column and row generation procedures are presented in [10]. The generation of columns does not require a pricing subproblem, since the columns are selected for addition to the LP by directly evaluating the reduced costs. The two processes in the column-and-row generation algorithm presented by [10] are strongly linked, with the addition of columns invoking the addition of related linking constraints. An important aspect of column-and-row generation is the calculation of the dual variable solutions for the eliminated constraints. In [10], the column-and-row generation approach is exact since the dual variables for the eliminated rows can be calculated correctly as a result of the problem structure. Avella *et al.* [10] present a special case in regards to the dual variable calculation, which is not applicable to the problems considered in this thesis. The method used to calculate an optimal dual solution in this thesis is presented in Section 7.3.2.

While column generation is applied to large-scale optimisation problems, there have been very few examples of similar applications of column-and-row generation. Muter *et al.* [66] present a robust crew pairing problem on a schedule including extra flights that are potentially employed during recovery. The extra flights considered in this problem are not available during the planning stage, however the crew pairings must be constructed under the expectation that the flights will be used in the event of a disruption. In [66], only a subset of extra flights will be required in the optimal solution. Therefore, all extra flights are eliminated from the original formulation to define a more tractable problem. Column generation is used to solve the current formulation of the master problem. Due to the eliminated constraints, it is possible that the early termination of the solution process may occur. This is addressed by a heuristic iterative procedure that examines previously generated columns to identify favourable rows for addition to the master problem. This is a good example applying column-and-row generation to real-world problems, however the use of a heuristic in the row generation procedure raises the possibility of suboptimal solutions. The column-and-row generation approach presented in this thesis is exact with the row generation procedure accurately calculating the dual values for the eliminated rows.

The split delivery vehicle routing problem and a network design problem for urban rapid transit systems are discussed by Feillet *et al.* [36]. The application of a simultaneous column and cut generation approach is described in relation to these two problems, where rows are added to eliminate infeasiblities in the dual problem. The row generation procedure involves the construction of a dual solution as part of an optimality condition. The approach that is presented in [36] dynamically generates structural constraints, however it is more closely related to cut generation than column-and-row generation.

7.1.1 Generic frameworks

Each of the above papers describe a unique implementation of the column-and-row generation solution approach. It is clear from the various implementations that there is little consensus on the general application of this solution approach. This is compounded further by three recent studies that focus on the development of a generic column-and-row generation framework, namely Frangioni and Gendron [39], Sadykov and Vanderbeck [77] and Muter *et al.* [65]. Each of the generic frameworks presented in these papers involve the dynamic generation of structural constraints, however there are key differences regarding the implementation.

Frangioni and Gendron [39] introduce the Structured Dantzig-Wolfe Decomposition (SDW) as an extension of the standard Dantzig-Wolfe decomposition to include row generation. This column-and-row generation approach involves the formulation of a Lagrangian subproblem to dualise rows not currently required in the master problem. Since a subset of all *complicating* structural constraints are dualised to reduce the problem complexity, the full problem formulation must be known *a priori*. An interesting aspect of this approach is that the duals related to the eliminated structural constraints can be ignored without any loss of exactness to the solution algorithm. Additional rows for the master problem are identified by solving the Lagrangian subproblem to find columns with a positive value in the optimal solution. These columns, along with their associated rows are added to the master problem by using a variable mapping from the subproblem variable space.

Column generation for extended formulations is presented by Sadykov and Vanderbeck [77] as a generic column-and-row generation approach. This approach shares many similarities with SDW, however there are some fundamental differences. The reformulation of the original problem permits a restricted form of the extended formulation to be defined. The restricted form includes only a subset of all possible columns and rows from the original problem, as such a subproblem is required to dynamically generate each. Similar to [39], a Lagrangian subproblem is formulated to identify any rows that are required in the current restricted extended formulation. An interesting aspect of this approach is that columns are generated using a subproblem designed for the compact formulation which are then added via a variable mapping to an extended formulation of the problem. By solving the master problem as an extended formulation, linear combinations of the added columns can be made to construct feasible solutions, which is not possible in a standard column generation approach.

Muter *et al.* [65] present a generic scheme for solving large-scale linear programs with column-dependent-rows. The column-dependent-rows is an interesting feature of optimisation problems which commonly arises in many practical applications. Two different examples are presented in [65], the multi-stage cutting stock problem and the quadratic set covering problem, to demonstrate the implementation of this approach. A key difference between Muter *et al.* [65] and both Frangioni and Gendron [39] and Sadykov and Vanderbeck [77] is that the problem is solved using the compact formulation and the row generation pricing subproblem is explicitly defined to calculate the dual variables for the eliminated constraints. In [65], it is identified that the dual variable solution related to the eliminated constraints are critical in the row generation procedure and defining optimality conditions. Finally, both [39] and [77] rely on the full problem description being know *a priori*, where [65] generates structural constraints without any prior knowledge of the eliminated rows. This difference presents a limitation of the former generic frameworks, while the latter can be applied to a wider range of problems.

An approach that is similar to the generic column-and-row generation frameworks presented above is the row-reduced column generation (RrCG) introduced by Desrosiers *et al.* [31]. RrCG generalises the improved primal simplex and dynamic constraint aggregation methods to improve the efficiency of column generation for highly degenerate master problems. A similarity between [31] and the framework presented in this chapter is the restriction on the set of *compatible* columns as a result of eliminating rows from the master problem. In addition, the set of duals related to the eliminated constraints must be calculated to solve a column generation subproblem that considers the *incompatible* columns.

The RrCG employs dynamic constraint aggregation, as such the implementation of this approach is fundamentally different to the column-and-row generation framework developed in this chapter. Firstly, the set of compatible columns in column-and-row generation is restricted based upon the elimination of variable linking constraints. Comparing this to the RrCG, the aggregated constraints in the master problem defines the set of compatible columns. Secondly, constraint aggregation in the RrCG permits the full problem description to be given by the master problem formulation. This is not the case for column-and-row generation, since the elimination of linking constraints restricts the integration of the decision variables. Thirdly, the method employed to update the set of compatible columns for column-and-row generation only requires the solution to a column generation subproblem, compared to solving an LP using column generation in the RrCG. Finally, updating the set of compatible columns in the RrCG results in an alternative aggregation of constraints in the master problem. This is significantly different to column-and-row generation, where rows are added to the master problem, leaving the current set of constraints unchanged.

7.2 Contributions of the Column-and-Row Generation Framework

The generic frameworks presented above do not directly apply to the problems presented in this thesis, thus an alternative framework is developed using existing features where appropriate. Firstly, the problem formulation of the integrated airline recovery problems presented in Chapters 8 and 9 are fully known *a priori*, which is a feature permitting the use [39] or [77]. However, the eliminated constraints are from the compact problem formulation, therefore the framework presented by Muter *et al.* [65] is more appropriate. The framework presented in this section is an extension of that developed by Muter *et al.* [65].

A contribution of this thesis is the further development of column-and-row generation and the provision of novel applications for this approach. In [65], the column-dependent-rows describe linking constraints between a primary set of variables and a single set of secondary variables. The framework presented here extends [65] to consider multiple sets of secondary variables. Secondly, an algorithmic approach is presented in this chapter to detail the implementation of column-and-row generation within a general framework. Another contribution of this chapter is the application of column-and-row generation to a problem structure that is commonly solved by Benders' decomposition. The column-and-row generation framework is presented as an alternative, efficient solution approach for this problem type. Finally, the generic framework developed by Muter *et al.* [65] does not provide any explicit evaluation of the runtime improvement compared to a standard column generation approach. This thesis will provide such an evaluation in regards to two different formulations of the integrated airline recovery problem.

7.3 Features of the Column-and-Row Generation Framework

The solution approach presented by Muter *et al.* [65] involves defining two restrictions on the original problem, the restricted master problem (RMP) and the short restricted master problem (SRMP). The RMP is constructed to contain all constraints from the original problem but only a subset of all possible variables. The SRMP describes a further restriction on the original problem, containing a subset of all variables *and* constraints that form the RMP. Since the SRMP is formed by eliminating structural constraints from the RMP, variable fixings in the column generation subproblems are used to restrict the set of permissible columns. This technique of variable fixing is presented by Zak [100], Muter *et al.* [65] and Muter *et al.* [66]. In addition, the variable fixing employed in this chapter is similar to defining the set of compatible columns in Desrosiers *et al.* [31]. By applying variable fixings in the column generation subproblem, all feasible solutions to the SRMP are feasible for the RMP and the original problem.

The critical aspects of the column-and-row generation procedure are the formulation of the RMP and SRMP, and the method used to calculate an optimal dual solution to identify favourable rows. These two features will be discussed in Sections 7.3.1 and 7.3.2 respectively. Finally, a general algorithm for the row generation procedure that will be employed throughout this thesis will be presented in Section 7.3.3. Since column generation is described in Section 4.1, it is not necessary to detail the application of this technique to solve the SRMP.

7.3.1 Formulation of the restricted problems

To provide an overview of column-and-row generation, the key features will be discussed with respect to a generic problem. The example problem is formulated with a single set of primary variables and multiple sets of secondary variables. In the problem description x is used to represent a single vector of primary variables, and each vector of secondary variables is given by $y^i, i \in \{1, 2, \ldots, n\}$. The multiple sets of secondary variables considered in this framework extends the generic framework presented by Muter *et al.* [65].

Since this section focuses on problems solved by column generation, it is assumed that vectors x and y^i contain only a subset of all possible variables from the original problem. The structure of the original problem, and by extension the RMP, contains a set of constraints related to each x and y^i, A^x and A^i respectively, with a set of linking constraints A^{iL}_x and A^{iL}_y

between x and y^i. The rows representing the linking constraints are dynamically generated using the row generation procedure developed in this section.

To construct the SRMP, it is necessary to redefine the variable vectors and constraint matrices used to describe the RMP. Initially, a subset of linking constraints are eliminated from the RMP, which involves removing rows from A_x^{iL} and A_y^{iL}, to provide the constraint matrices $\bar{A}_x^{iL}$ and $\bar{A}_y^{iL}$ respectively. As stated previously, the elimination of rows from the RMP is coupled with the fixing of variables in the column generation subproblem. This is required to prohibit the generation of variables with non-zero elements in the eliminated rows. Consequently, the set of all possible variables that can be generated for the SRMP is reduced, therefore the vectors $\bar{x} \subset x$ and $\bar{y}^i \subset y^i$ are defined. While all rows in the matrices A^x and A^i are still present in the formulation of the SRMP, the restriction on the possible set of variables requires the elimination of columns, hence the matrices $\bar{A}^x$ and $\bar{A}^i$ are defined.

The matrix representation of the RMP and SRMP is given by,

RMP

$$\min \quad c^x x + \sum_i c^i y^i, \tag{7.1}$$

$$\text{s.t.} \quad A^x x = b, \tag{7.2}$$

$$A^i y^i = b^i \ \ \forall i, \tag{7.3}$$

$$A_x^{iL} x - A_y^{iL} y^i = 0 \ \ \forall i, \tag{7.4}$$

$$x \geq 0, \ \ y^i \geq 0. \tag{7.5}$$

SRMP

$$\min \quad \bar{c}^x \bar{x} + \sum_i \bar{c}^i \bar{y}^i, \tag{7.6}$$

$$\text{s.t.} \quad \bar{A}^x \bar{x} = b, \tag{7.7}$$

$$\bar{A}^i \bar{y}^i = b^i \ \ \forall i, \tag{7.8}$$

$$\bar{A}_x^{iL} \bar{x} - \bar{A}_y^{iL} \bar{y}^i = 0 \ \ \forall i, \tag{7.9}$$

$$\bar{x} \geq 0, \ \ \bar{y}^i \geq 0. \tag{7.10}$$

It must be reiterated that the elimination of rows also limits the set of feasible variables that can be generated in the column generation subproblems. This is critical aspect of the implementation of column-and-row generation, permitting the property that a feasible solution for the SRMP is feasible for the RMP and the original problem. This property is used in the calculation of an optimal dual solution, which is required to identify favourable rows for addition to the SRMP.

7.3.2 Calculation of dual solutions

To demonstrate the correctness of the column-and-row generation framework described here, an additional problem is introduced, the RMP′. This problem is formulated with all constraints from the RMP, but only a subset of all possible variables. Thus, the formulation of the RMP′

is identical to the RMP at an intermediate stage of the column generation solution process.

To describe the RMP′, the matrices $\tilde{A}_y^{iL}$ are introduced to contain all rows eliminated from A_y^{iL} to construct the SRMP. Additionally, a set of dummy variables $\tilde{y}^i$ are created such that each $y \in \tilde{y}^i$ has a non-zero element in only one row of $\tilde{A}_y^{iL}$. This is an important condition imposed on the construction of $\tilde{y}^i$ that is exploited in Theorem 7.3.1. The dummy variables contained in $\tilde{y}^i$ also have non-zero elements in the rows of (7.3), thus the matrices $\tilde{A}^i$ must be introduced. Therefore, the matrix representation of the RMP′ is given by,

$$(\text{RMP}') \quad \begin{aligned} \min \quad & \bar{c}\bar{x} + \sum_i \left\{ \bar{c}^i \bar{y}^i + \tilde{c}^i \tilde{y}^i \right\}, && (7.11)\\ \text{s.t.} \quad & \bar{A}\bar{x} = b, && (7.12)\\ & \bar{A}^i \bar{y}^i + \tilde{A}^i \tilde{y}^i = b^i \ \ \forall i, && (7.13)\\ & \bar{A}_x^{iL}\bar{x} - \bar{A}_y^{iL}\bar{y}^i = 0 \ \ \forall i, && (7.14)\\ & -\tilde{A}_y^{iL}\tilde{y}^i = 0 \ \ \forall i, && (7.15)\\ & \bar{x} \geq 0, \ \ \bar{y}^i \geq 0, \ \ \tilde{y}^i \geq 0. && (7.16) \end{aligned}$$

For each row in equations (7.12)-(7.15) there is an equivalent row in equations (7.2)-(7.4). It is assumed that in this formulation the optimal solution to the RMP′ is not the optimal solution to the original problem. This implies that additional columns with a negative reduced cost can be found by solving the column generation subproblems for the primary and secondary variables. This is an important property of the RMP′ that is used to ensure the correctness of the dual solution calculation.

By construction, an optimal primal solution to the SRMP is a feasible solution to the RMP′. The following lemma will prove that this feasible primal solution to the RMP′ is an optimal solution.

Lemma 7.3.1. *The optimal primal solution to the SRMP is an optimal primal solution to the RMP′.*

Proof. The constraints (7.15) force the variables $\tilde{y}^i$ to be zero in any feasible solution of the RMP′. As such, the optimal primal solution to the RMP′ can be found by eliminating constraints (7.15) and solving this problem with the variables $\tilde{y}^i$ fixed to zero. Solving this modified form of the RMP′ is equivalent to solving the SRMP. □

It is possible to solve the LP relaxation of the RMP′ to find the optimal dual solution.

However, given the large number of linking constraints in the original problem, this is potentially a very time consuming process. Since the SRMP has been solved to optimality, ideally the dual solution to the RMP′ can be calculated from this solution.

There are two major steps in the procedure to calculate an optimal dual solution for the RMP′. Firstly, the constraints (7.7)-(7.9) in the SRMP are identical to the constraints (7.12)-(7.14). Therefore, the solutions to the related dual variables can be simply equated. The second step involves finding the solutions for the dual variables related to the rows in (7.15), which are the constraints eliminated to form the SRMP. This involves solving the column generation subproblems for the secondary variables to accurately calculate the values of these dual variables.

Additional notation is required to describe the calculation of the dual variables for the rows eliminated to form the SRMP. An index set U_i is defined to reference each row u in the constraint matrix A_y^{iL}. Extending this notation, the index set for the rows included in the SRMP is given by $\bar{U}_i$ and all eliminated rows are contained in $U_i \backslash \bar{U}_i$. Finally, the dual variables for each row in A_y^{iL} is given by $\boldsymbol{\gamma}^i = \{\gamma_u^i | u \in U_i\}$. This notation conveniently describes the rows which are eliminated or contained in the SRMP and the dual values which must be computed.

The value of γ_u^i is calculated from the minimum reduced cost $\hat{c}^i$ for a variable with a non-zero entry in row u of matrix $\tilde{A}_y^{iL}$. This is achieved by executing Algorithm 7.1.

Algorithm 7.1 Computing a feasible dual solution

1: Assume that $\gamma_u^i = 0$ and force all feasible solutions to the column generation subproblem for the secondary variables i to have a 1 in row u of $\tilde{A}_y^{iL}$ and 0 in all rows $v \in U_i \backslash \bar{U}_i, v \neq u$.

2: Solve the column generation subproblem to identify variable $\hat{y}$ that has the minimum reduced cost $\hat{c}^i$.

3: Set $\gamma_u^i = -\hat{c}^i$.

The calculation procedure given by Algorithm 7.1 relies on the structure of the RMP′ and the form of the dummy variables that populate the rows $u \in U_i \backslash \bar{U}_i$. The reasoning provided here draws upon the discussion related to the column-and-row generation framework by Muter *et al.* [65]. For $\hat{y}$, found by Algorithm 7.1, to be eligible to enter the basis of the RMP′ implies that $\hat{c}^i < 0$. Since $\hat{y}$ has a non-zero element in a row of $\tilde{A}_y^{iL}$, the construction of the RMP′ forces $\hat{y} = 0$ upon the addition of this column, resulting in a degenerate simplex iteration. To avoid this situation, it is assumed that the minimum reduced cost for all variables found using Algorithm 7.1 is at least zero. This requirement ensures that the dual solutions that are computed for $\boldsymbol{\gamma}^i$ are feasible for the RMP$'$. The following theorem will prove the feasibility of the computed values for $\boldsymbol{\gamma}^i$ and that the resulting feasible dual solution is also optimal.

Theorem 7.3.1. *The dual solutions computed for $\boldsymbol{\gamma}^i$ forms an optimal dual solution to the RMP$'$.*

Proof. The first step of this proof is to show that the solutions calculated for the dual variables $\boldsymbol{\gamma}^i$ using Algorithm 7.1 are feasible for the RMP$'$. For this proof the variable $\bar{\gamma}_u^i$ is assumed to have a value that satisfies all dual constraints of the RMP$'$. Additionally, the reduced cost of a column $y \in \tilde{y}^i$ is given by $\bar{c}_y^i$.

Algorithm 7.1 solves the column generation subproblem for the secondary variables i to identify $\hat{y}$ that has the minimum reduced cost $\hat{c}^i$, assuming $\gamma_u^i = 0$. Comparing $\hat{y}$ with the variables currently in the RMP$'$, there are two possible outcomes:

i) There exists a column $y \in \tilde{y}^i$ identical to $\hat{y}$. Since y is identical to $\hat{y}$, $\hat{c}^i = \bar{c}_y^i - \bar{\gamma}_u^i$. In Algorithm 7.1, the value of γ_u^i is set to $-\hat{c}^i$, hence $\bar{c}_y^i - \bar{\gamma}_u^i + \gamma_u^i = 0$. Therefore, setting $\gamma_u^i = -\hat{c}^i$ ensures dual feasibility.

ii) There exists a column $y \in \tilde{y}^i$ that has a non-zero element in row u of $\tilde{A}_y^{iL}$ but is not identical to $\hat{y}$. This implies that the variable γ_u^i exists in at least one dual constraint. Assume that setting $\gamma_u^i = -\hat{c}^i$ violates a constraint in the dual of the RMP$'$. This implies that $\bar{c}_y^i$ calculated using $\gamma_u^i = -\hat{c}^i$ in place of $\bar{\gamma}_u^i$ is negative, i.e. $\bar{c}_y^i - \bar{\gamma}_u^i + \gamma_u^i < 0$. Since $\hat{c}^i + \gamma_u^i = 0$, then $\hat{c}^i > \bar{c}_y^i - \bar{\gamma}_u^i$. Now, step 2 of Algorithm 7.1 identifies $\hat{y}$ that has the minimum reduced cost, so $\hat{c}^i > \bar{c}_y^i - \bar{\gamma}_u^i$ is a contradiction. Therefore, $\bar{c}_y^i - \bar{\gamma}_u^i + \gamma_u^i \geq 0$ must be true, hence the computed value for γ_u^i satisfies all dual constraints.

The first step of this proof has established that the dual solutions calculated for $\boldsymbol{\gamma}^i$ using Algorithm 7.1 are feasible for the RMP$'$. Therefore, the calculated values for $\boldsymbol{\gamma}^i$ can be used as the dual solutions for the constraints (7.15). A feasible dual solution for the RMP$'$ is then simply constructed by equating the solutions to the dual variables representing constraints (7.12)-(7.14) to the dual solutions of the SRMP.

The second step proves that the feasible dual solution constructed for the RMP$'$ is also optimal. Firstly, it is stated in Lemma 7.3.1 that the optimal primal solution to the SRMP is also optimal for the RMP$'$. In addition, the solutions to the dual variables representing constraints (7.12)-(7.14) are equated to the dual solutions of the SRMP. Since the right hand side of the constraints represented by equations (7.15) are zero, the value of the respective dual variables do not affect the optimal objective function value. It follows that the dual objective function value for the RMP′ is identical to the dual objective function value of the SRMP. Given that the dual objective function value of the SRMP is equal to the primal objective values of the SRMP and RMP′, then primal and dual objective values for the RMP′ are equal. Therefore, the feasible dual solution constructed for the RMP′ is optimal. □

7.3.3 Row generation algorithm

Using the optimal dual solution to the RMP′, the row generation algorithm is executed to identify rows to update the SRMP. This procedure involves solving the column generation subproblem for the primary variables to find negative reduced cost columns feasible for the RMP′. While all columns identified during this procedure are feasible for the RMP′, it is likely that, due to the eliminated constraints, they are infeasible for the SRMP. Such columns are identified by displaying at least one non-zero element in the rows $u \in U_i \backslash \bar{U}_i$. If columns infeasible for the SRMP are found, then u must be added to $\bar{U}_i$ and the related row to the SRMP. Consequently, the SRMP grows vertically and horizontally with the addition of rows and columns respectively. The row generation algorithm is detailed in Algorithm 7.2.

The column-and-row generation solution approach terminates when no favourable rows are

---

Algorithm 7.2 Row generation algorithm

Input: An optimal solution to the SRMP.

Set the dual values for rows (7.12)-(7.14) to the dual solutions for the rows (7.7)-(7.9).
for all secondary variable sets i **do**
  for all rows u contained in $\tilde{A}_y^{iL}$ **do**
    Execute Algorithm 7.1 to compute the value of γ_u^i.
  end for
end for
By Theorem 7.3.1 an optimal dual solution for the RMP′ has been computed.
Solve the column generation subproblem for the primary variables to identify variables feasible for the RMP′.
if a negative reduced cost column has at least one non-zero entry in the rows of $\tilde{A}_y^{iL}$ **then**
  add the rows with non-zero entries in $\tilde{A}_y^{iL}$ to $\bar{A}_y^{iL}$.
end if

---

identified by Algorithm 7.2. This is consistent with the termination condition of the standard column generation approach, terminating when no columns with a negative reduced cost for the RMP are found. Since an optimal dual solution is calculated for the RMP′, the column generation subproblem for the primary variables accurately evaluates the minimum reduced cost. Therefore, if no negative reduced cost columns are found for the RMP′, then the solution to the RMP′ is optimal for the original problem.

7.4 Column-and-Row Generation Solution Algorithm

The column-and-row generation solution algorithm implemented in this thesis is developed by combining the fundamental features of the approach developed in Section 7.3. The first stage of the solution algorithm involves the formulation of the SRMP, which is detailed in Section 7.3.1. Using the solution to the SRMP, Section 7.3.2 details the calculation procedure that is required to form an optimal dual solution for the RMP′. The final step in the column-and-row generation solution algorithm, described in Section 7.3.3, executes Algorithm 7.2 to identify favourable rows for the SRMP. The complete column-and-row generation solution algorithm is given by Algorithm 7.3.

The various applications of column-and-row generation in Section 7.1 present two alternative implementations of the column and row generation stages. In particular, the two stages are either performed separately, which is presented by Zak [100] and Muter *et al.* [66], or they are performed together, as in Avella *et al.* [9], Avella *et al.* [10] and Feillet *et al.* [36]. The column-and-row generation solution approach described by Algorithm 7.3 is an example of the former implementation. Specifically, the SRMP is solved to optimality by column generation

Algorithm 7.3 Column-and-row generation algorithm

1: Eliminate columns from the original problem to form the RMP.
2: Eliminate rows (and subsequently columns) from the RMP to form the SRMP.
3: **repeat**
4: Solve the SRMP to optimality.
5: Use Algorithm 7.2 to compute the optimal dual solution to the RMP′ and identify any favourable rows.
6: **until** no rows are added to $\bar{A}_y^{iL}$ in Algorithm 7.2
7: The solution to the SRMP is the optimal solution to the original problem.

prior to the execution of the row generation procedure. By separating the column and row generation stages, a reduction in the runtimes of the column generation subproblems is achieved by restricting the set of permissible columns. This is an important feature of the solution approach that significantly affects the problems investigated in this thesis.

7.5 Summary

This chapter presents a general framework for column-and-row generation that is employed to solve the integrated airline recovery problems developed in Chapters 8 and 9. A contribution of this framework is the consideration of multiple sets of secondary variables with related linking constraints. This extends the column-and-row generation approach presented by Muter *et al.* [65], requiring the solution to multiple column generation subproblems to calculate an optimal dual solution for the RMP$'$. In addition, the framework presented in this chapter is applied to problems with linking constraints in the compact formulation. As such, this framework is a direct alternative to decomposition methods, such as Benders' decomposition, which are commonly applied to this problem type.

The following chapters describe the implementation of column-and-row generation to solve the integrated airline recovery problem. The efficiency of the solution approach is critical in airline recovery, therefore the use of column-and-row generation is investigated to improve the solution runtimes. Chapters 8 and 9 explicitly evaluate the improvements in solution runtime and quality achieved with column-and-row generation compared to a standard column generation approach. While this evaluation is provided for frameworks of Frangioni and Gendron [39] and Sadykov and Vanderbeck [77], this is not the case for the framework by Muter *et al.* [65]. The evaluation presented in the following chapters and the development of related enhancement techniques are contributions of this thesis.

Chapter 8

Integrated Airline Recovery Problem

The application of the recoverable robustness framework to aircraft routing problems is presented in Chapters 5 and 6. The objective of recoverable robustness is to improve the recoverability of the planned solution, which is of critical importance for airline operations. The potential of this technique is demonstrated by the recoverable robust tail assignment (RRTAP) and the recoverable robust single day aircraft maintenance routing problem (SDAMRP-RR). However, focusing on a single stage limits the effectiveness of this approach. An improvement in recoverability for the aircraft routing problem alone does not guarantee that this improvement will be realised in the operating environment involving the related resources of crew and passengers.

Based on the results of Chapters 5 and 6, it is expected that the recoverability of the complete airline operations can be improved by applying recoverable robustness to an integrated airline planning problem. In Chapter 2, the complete airline planning process is described as a very large and intractable problem, with each individual stage presenting a difficult optimisation problem. As such, any model integrating multiple planning stages is very complex, requiring the use of sophisticated solution techniques. Examples of integrated planning problems are presented in Cordeau *et al.* [27] and Dunbar *et al.* [33], each describing alternative solution approaches.

A complicating factor of integrating aircraft and crew in the recoverable robustness framework is the runtime of the evaluation recovery problem. In Chapters 2 and 3 the planning and recovery tail assignment problems are discussed, where the latter is described as a more difficult problem. Given that the improvement in recoverability is highly dependent on the feedback provided from the recovery evaluation stage, it is necessary to solve the recovery problem to near optimality. Therefore, the objective of this chapter is to investigate solution approaches that efficiently solve an integrated airline recovery problem in short runtimes.

Various solution approaches employed to solve each stage of the airline recovery process are presented in Chapter 3. While there has been considerable interest in individual recovery stages, the integrated recovery problem has recently been receiving increased attention. There are two key approaches used to reduce the complexity of the integrated problem and improve solution runtimes, i) applying Benders' decomposition to partition the individual stages into subproblems [55, 70], and ii) partitioning the flight schedule into chronologically ordered sets of flights to define a series of discrete recovery problems [2]. While each of these approaches efficiently solve the integrated recovery problem, approximations are made, which reduces the solution quality. In particular, Benders' decomposition is solved with continuous second stage variables, therefore the optimal solution may not provide integral optimality. This chapter applies column-and-row generation to directly solve the integrated recovery problem in an effort to improve solution runtimes and further develop the available solution techniques.

The contributions of this chapter are:

1. the explicit evaluation of column-and-row generation against a standard column generation approach, identifying various enhancement techniques,

2. integrating column-and-row generation into a branch-and-price framework,

3. solving the integrated airline recovery problem using column-and-row generation, guaranteeing near optimal integer solutions.

Firstly, the column-and-row generation framework developed in Chapter 7 extends the generic scheme presented by Muter *et al.* [65]. The scheme developed by [65] has been applied to various problem, however no evaluation of column-and-row generation against a standard column generation approach is presented. This chapter provides such an evaluation, and as a result a number of enhancement techniques are identified. Column-and-row generation has been developed as a method solve large-scale linear programs displaying a special problem structure. As such, the integration of this approach with a branch-and-price framework has not been published before. This chapter presents a strategy to efficiently solve each node in the branch-and-bound tree by column-and-row generation. Finally, the application of column-and-row generation to solve the integrated recovery problem eliminates the need for approximations in the solution process. Integral optimality is achieved for all components of the integrated problem, contributing to the available solution techniques for this problem.

This chapter presents the integrated airline recovery problem and discusses the application of the column-and-row generation framework presented in Chapter 7. The master problem for the integrated recovery problem is presented in Section 8.1, which is solved using column generation to provide benchmark results. The implementation of column generation and column-and-row generation is presented in Section 8.2, including a number of enhancement techniques identified through the evaluation of the solution approach. To demonstrate the benefits of solving the integrated recovery problem by column-and-row generation, a comparison against the results produced using column generation is made in both solution quality and runtime. These results are presented in Section 8.3. The conclusions provided in Section 8.4 aim to present the technique of column-and-row generation as an alternative solution method for integrated airline and transportation problems. The work that is presented in this chapter appears as part of a paper submitted to Transportation Science by Maher [61].

8.1 Integrated Airline Recovery Problem

The integrated recovery problem (IRP) attempts to minimise the costs associated with flight delays and cancellations and the additional cost of crew following a schedule disruption. This problem is formulated to integrate the schedule, aircraft and crew recovery problems. In Chapter 3 the tail assignment and crew duty recovery problems are presented. These problems form the basis for the IRP, with the link between the two provided by the flight cancellation and delay decisions and specific flights allocated to each aircraft and crew. In this chapter we discuss the key features of the recovery tail assignment problem (RTAP) and crew duty recovery problem (CDRP), presented in Sections 3.1.1 and 3.1.2 respectively, that are relevant to the IRP.

The notation used to describe the IRP is first introduced in Chapters 2 and 3, however for completeness the relevant notation is provided Tables 8.1 and 8.2. The notation used to define the IRP will be described with reference to Chapters 2 and 3. For this problem, the set of all

K	is the set of all planned and reserve crew k
K^{res}	is the set of all reserve crew, $K^{res} \subset K$
R	is the set of all aircraft r
P^k, P^r	is the set of all flight strings p assigned to crew k or aircraft r respectively
N	is the set of all flights j
B^K, B^R	is the set of crew bases/overnight airports b for crew and aircraft respectively
N^D	is the set of disruptable flights j, $N^D \subseteq N$
N^K_{in}, N^R_{in}	is the set of all carry-in activities j, flights and origination nodes, for crew and aircraft respectively, $N^K_{in} \subset N \cup B^K$ and $N^R_{in} \subset N \cup B^R$
N^K_{out}, N^R_{out}	is the set of all carry-out activities j, flights and termination nodes, for crew and aircraft respectively, $N^K_{out} \subset N \cup B^K$ and $N^R_{out} \subset N \cup B^R$
U_j	is the set of all delay copies v for flight $j \in N$
$\hat{N}$	is the set of all nodes in the connection network defined by flight-copy pairs j_v, representing flights, origination and termination nodes
$\hat{N}^D$	is the set of disruptable nodes in the connection network defined by flight-copy pairs j_v, representing flights, origination and termination nodes
C^K, C^R	is the set of connections $(i_u, j_v), i_u, j_v \in \hat{N}$ for crew and aircraft respectively
E	is the set of short connections $E = C^R \backslash C^K$
E^D	is the set of disruptable short connections, $E^D = \{(i_u, j_v) \in E \vert i_u \in \hat{N}^D \vee j_v \in \hat{N}^D\}$

Table 8.1: Sets used in the IRP.

crew is given by K, indexed by k, and the set of all aircraft is given by R, indexed by r. The set of crew K also includes all available reserve crew K^{res}. As an extension of current techniques, the solution approach for the IRP demonstrates an efficient algorithm that includes all crew and aircraft, as defined by K and R respectively. Using all crew and aircraft allows for the optimal allocation of all available resources.

8.1.1 Recovery flight schedule and connection network

A single day flight schedule is used to described and evaluate the IRP, with the set of flights in the schedule given by N. A recovery window is used for the IRP to restrict the number of flights considered in the recovery problem. Thus, the set of disruptable flights N^D, a subset of all flights N, is defined to contain all flights that are primarily affected by the disruption and those that depart after the disruption occurs, but before the end of the specified time window.

Restricting the flights included in the IRP using a recovery window requires the activities performed by crew and aircraft before and after this window to be fixed. This is achieved using the concepts of carry-in and carry-out activities, as described in Section 3.1.1. All carry-in activities for crew and aircraft are contained in the sets N_{in}^{K} and N_{in}^{R} respectively. Similarly all carry-out activities for the crew and aircraft are contained in the sets N_{out}^{K} and N_{out}^{R} respectively.

To efficiently solve the IRP, the recovery policy of flight delays is implemented using flight copies. The technique described in Section 3.1.1 for modelling flight delays using flight copies is used for the IRP. In this model, it is important to note that the nodes representing origination and termination airports are treated as non-disruptable flights. This definition is made for convenience in discussing carry-in and carry-out activities.

The connection network used for this model is described using the flight-copy representation for each node in the network. The set of all nodes is represented by $\hat{N} = \{j_v | j \in N \wedge v \in U_j\}$, detailing all flight-copy pairs that exist for each disruptable and non-disruptable flight. Using the same notation, the set of all disruptable nodes is given by $\hat{N}^D = \{j_v | j \in N^D \wedge v \in U_j\}$. The connection network for this problem is defined by a set of nodes, given by flight-copy pairs, and a set of arcs as connections between the nodes. A connection between two flight-copy pairs $(i_u, j_v), i_u, j_v \in \hat{N}$ is feasible if i) the destination of flight i is the same as the origin of flight j and ii) the departure of flight-copy j_v occurs after a specified amount of time following the arrival of flight-copy i_u. All feasible connections for crew are contained in the set C^K and require a minimum sit time between the arrival of i_u and the departure of j_v. Feasible connections for aircraft require a minimum turn time for each connection contained in C^R.

8.1.2 Aircraft routes and crew duties

The modelling approach for the IRP is based upon the string formulation introduced by Barnhart *et al.* [12]. In the IRP, a flight string is defined as a set of connected flights from a carry-in to a carry-out activity. The flight strings are constructed individually for each crew and aircraft, as such the crew duties describe personalised schedules and the aircraft routes describe individual tail assignments. The terms aircraft route and tail assignment will be used interchangeably throughout this chapter.

Using the flight-copy representation of each node for this model, any reference to flight j without specifying a copy v collectively states all flight-copy pairs associated with that flight. So, the parameters a_{jp}^{k} and a_{jp}^{r} specify whether flight j, representing any flight-copy pair $j_v, v \in U_j$, is included on string p for crew k and aircraft r respectively. A flight string will either terminate within the recovery window, by ending at a crew base or aircraft overnight airport, or will

x_p^k	= 1 if crew k uses flight string p, 0 otherwise
y_p^r	= 1 if aircraft r uses flight string p, 0 otherwise
c_p^k, c_p^r	= the cost of using flight string p for crew k or aircraft r respectively
a_{jp}^k, a_{jp}^r	= 1 if flight j is in string p for crew k or aircraft r respectively, 0 otherwise
$e_{i_u j_v p}^k$, $e_{i_u j_v p}^r$	= 1 if connection (i_u, j_v) is in string p for crew k or aircraft r respectively, 0 otherwise
a_{jp}^{kv}, a_{jp}^{rv}	= 1 if flight-copy j_v is in string p for crew k or aircraft r respectively, 0 otherwise
o_{bp}^r	= 1 if string p, assigned to aircraft r, terminates at airport b within the recovery window, 0 otherwise
z_j	= 1 if the flight j is cancelled, 0 otherwise
d_j	= the cost of cancelling flight j
κ_j^{v+}	= the number crew deadheading on flight-copy j_v
κ_j^{v-}	dummy variable for counting the number of deadheading crew on flight-copy j_v
ν_k	= 1 if crew k deadhead back to their crew base from the start of the disruption period, 0 otherwise
g^{DHD}	= the cost of deadheading crew on one leg within a duty
g^{DHB}	= the cost of deadheading crew k back to their crew base
$\bar{o}_b^r$	= 1 if the planned flight string for aircraft r terminates at airport b within the recovery window, 0 otherwise

Table 8.2: Variables used in the IRP.

terminate at a carry-out flight. If the flight string assigned to an aircraft terminates within the recovery window, the parameter o_{bp}^r describes the terminating airport b for aircraft r. Since flight cancellations are implemented as a recovery policy for the IRP, the flow balance of the original schedule is not maintained. To ensure enough aircraft are positioned at each airport b to operate the schedule for the following day, the minimum number of required aircraft must be specified. Now, the number of planned aircraft flight strings terminating at each end-of-day location within the recovery period is given by $\sum_{r \in R} \bar{o}_b^r, \forall b \in B^R$, where $\bar{o}_b^r$ indicates that aircraft r terminates at airport b. Therefore, this expression defines the minimum number of aircraft required to terminate at each end-of-day location b within the recovery window for the IRP.

Within the set of all aircraft connections, C^R, it is common to have connection times less than the minimum sit time for crew. These connections are called *short connections*, and it is permissible for crew to operate the two flights in succession, as defined by this connection, only if a single aircraft also operates the same two flights. The set of all short connections are contained in $E = C^R \backslash C^K$, and the subset of short connections that include flight-copy pairs in $\hat{N}^D$ are contained in the set E^D. If a flight string includes two flight-copy pairs that form a short connection, the parameters $e^k_{i_u j_v p}$ and $e^r_{i_u j_v p}$ indicate the inclusion of connection (i_u, j_v) on string p for crew and aircraft respectively.

Legality of crew duties

There a numerous rules that dictate the construction of feasible flight strings for crew which must be strictly adhered to during the recovery process. Crew flights strings can be described as either a duty, pairing or schedule, each spanning a different time period. A crew duty is the most fundamental flight string for crew which specifies the sequence of flights performed during a single day. A crew pairing is constructed as a sequence of duties, and similarly a crew schedule is a sequence of pairings. There are rules that are specific to the construction of duties, pairings and schedules, however for a single day schedule, which is used for the IRP, the most important rules that must be considered are the crew duty rules.

One of the most fundamental rules regarding the construction of crew duties is the flight string origination and termination locations. Each crew is employed at one of many crew bases throughout the network, so to avoid overnight stays away from base, ideally a duty is constructed to start and end at the same crew base. Unfortunately, the design of the flight schedule does not permit all crew duties to terminate at their respective crew base, requiring an overnight stay at a permissible airport. This rule is modelled in the IRP through the construction of recovered crew duties and if a crew duty originates from a permissible overnight airport, it must terminate the required crew base.

The number of hours that a crew duty spans is an important consideration to manage the effects of fatigue. There are two rules that are modelled in the IRP, a maximum number of flying hours and a limit on the total duration of the crew duty, which are set of 8 and 13 hours respectively. These are the most important duty rules related to working hours and are strictly adhered to in the IRP. Another important, but complicated, rule is the 8-in-24 rule that requires crew to receive additional rest if more that 8 hours flying is performed in a 24 hour period [13]. Given that crew are assigned personalised schedules and are limited to 8 hours flying in a single day, the 8-in-24 rule will not be violated for most crew in the recovered solution. In the event that this rule is violated by the solution to the IRP, further adjustment can be made to the recovered duties at the end of the day to provide an adequate amount of rest.

It is important to note that prior to a disruption crew may have performed part of a duty, consuming allowable flying and working hours. The personalised schedules are respected in the recovery of crew duties by originating each duty from a carry-in location and accounting for the working *history* prior to the disruption. This ensures that the recovered crew duties, including the flights performed prior to the disruption, respect the crew duty rules.

8.1.3 Recovery policies

The set of recovery policies implemented in the IRP include the generation of new aircraft routes and crew duties, crew deadheading (transportation of crew as passengers), the use of reserve crew, and flight delays and cancellations. In the column generation algorithm, feasible crew duties and aircraft routes are generated for each crew and aircraft contained in K and R respectively. The length of delay that is required on each flight is determined in the generation of these new flight strings for aircraft and crew by the selection of flight-copy pairs. The parameters a_{jp}^{kv} and a_{jp}^{rv} describe the length of delay, as specified by copy v, selected for flight j on string p for the crew and aircraft respectively. The IRP also allows for the cancellation of any flight that can not be covered by both crew and aircraft. Flight cancellations are defined in the IRP through the use of the variables z_j, which equal 1 to indicate that flight j is cancelled at a cost of d_j.

Since a disruption affects the departure and arrival times of flights, it is possible for crew members to violate duty rules if the original duties are operated as planned. To avoid exceeding any duty limits a set of crew specific recovery policies are employed. Firstly, crew deadheading is used to transport crew as passengers to continue the operation of disrupted flight strings. Two different types of deadheading are implemented in the IRP, deadheading within a duty and deadheading back to base. The variables κ_j^{v+} are introduced to count the number of crew that deadhead within a duty on flight-copy j_v. To model crew deadheading within a duty the dummy variables κ_j^{v-} are required to ensure that the number of crew deadheading on flight-copy j_v is one less than the total number of crew assigned to that flight-copy. The cost of deadheading crew on one leg within a duty is given by g^{DHD}. Alternatively, the variables ν_k indicate whether crew k deadhead to their crew base immediately following the start of the disruption at a cost of g^{DHB}.

As a result of recovery actions, it is not guaranteed that the set of originally planned crew are able to operate the recovered schedule. To achieve the greatest coverage of flights, reserve crew are employed to operate duties unable to be performed by the original set of planned crew. This recovery action provides crew to operate flights in an effort to avoid costly flight cancellations.

The integrated recovery problem is presented in a compact formulation with variables x_p^k for crew and y_p^r for aircraft representing feasible flight strings. The full description of this problem is presented below,

$$\min \quad \sum_{k\in K}\sum_{p\in P^k} c_p^k x_p^k + \sum_{j\in N^D}\sum_{v\in U_j} g^{DHD}\kappa_j^{v+} + \sum_{k\in K} g^{DHB}\nu_k + \sum_{r\in R}\sum_{p\in P^r} c_p^r y_p^r + \sum_{j\in N^D} d_j z_j, \tag{8.1}$$

(IRP)

$$\text{s.t.} \quad \sum_{k\in K}\sum_{p\in P^k} a_{jp}^k x_p^k - \sum_{v\in U_j}\kappa_j^{v+} + z_j = 1 \quad \forall j\in N^D, \tag{8.2}$$

$$\sum_{k\in K}\sum_{p\in P^k} a_{jp}^k x_p^k = 1 \quad \forall j\in N_{out}^K, \tag{8.3}$$

$$\sum_{r\in R}\sum_{p\in P^r} a_{jp}^r y_p^r + z_j = 1 \quad \forall j\in N^D, \tag{8.4}$$

$$\sum_{r\in R}\sum_{p\in P^r} a_{jp}^r y_p^r = 1 \quad \forall j\in N_{out}^R, \tag{8.5}$$

$$\sum_{r\in R}\sum_{p\in P^r} o_{bp}^r y_p^r \geq \sum_{r\in R}\bar{o}_b^r \quad \forall b\in B^R, \tag{8.6}$$

$$\sum_{k\in K}\sum_{p\in P^k} a_{jp}^{kv} x_p^k - \kappa_j^{v+} + \kappa_j^{v-} = 1 \quad \forall j\in N^D, \forall v\in U_j, \tag{8.7}$$

$$\sum_{k\in K}\sum_{p\in P^k} e_{i_u j_v p}^k x_p^k - \sum_{r\in R}\sum_{p\in P^r} e_{i_u j_v p}^r y_p^r \leq 0 \quad \forall (i_u, j_v)\in E^D, \tag{8.8}$$

$$\sum_{k\in K}\sum_{p\in P^k} a_{jp}^{kv} x_p^k - \kappa_j^{v+} - \sum_{r\in R}\sum_{p\in P^r} a_{jp}^{rv} y_p^r = 0 \quad \forall j\in N^D, \forall v\in U_j, \tag{8.9}$$

$$\sum_{p\in P^k} x_p^k + \nu_k = 1 \quad \forall k\in K\backslash K^{res}, \tag{8.10}$$

$$\sum_{p\in P^k} x_p^k \leq 1 \quad \forall k\in K^{res}, \tag{8.11}$$

$$\sum_{p\in P^r} y_p^r = 1 \quad \forall r\in R, \tag{8.12}$$

$$x_p^k\in\{0,1\} \;\; \forall k\in K, \forall p\in P^k, \quad y_p^r\in\{0,1\} \;\; \forall r\in R, \forall p\in P^r, \tag{8.13}$$

$$z_j\in\{0,1\} \;\; \forall j\in N, \quad \nu_k\in\{0,1\} \;\; \forall k\in K, \tag{8.14}$$

$$\kappa_j^{v+}\geq 0, \kappa_j^{v-}\geq 0 \quad \forall j\in N^D, \forall v\in U_j. \tag{8.15}$$

The objective of the IRP is to minimise the cost of recovery for aircraft and crew. The recovery costs include the cost of flight delays and cancellations, reserve crew, additional crew duty costs and the cost of crew deadheading.

The coverage of flights within the recovery window by the crew and aircraft is enforced by constraints (8.2) and (8.4). The terms $a^k_{jp}x^k_p$ and $a^r_{jp}y^r_p$ in these constraints denote whether flight j is operated by crew k and aircraft r respectively and the variables z_j indicate whether it is cancelled. The constraints (8.3) and (8.5) ensure that each carry-out flight is serviced by crew and aircraft in the recovered solution. The carry-out flight coverage ensures that the solution to the IRP positions the crew and aircraft to continue activities following the end of the recovery window as planned.

This problem is solved for a single day schedule, so the aircraft are required to be positioned at airports to maintain flow balance for the following days operations. Since all recovery actions occur within the recovery window, the positioning of the aircraft must be considered before the conclusion of this window. Two cases can occur in the recovery of aircraft; either i) an aircraft is assigned a carry-out flight, allowing it to follow a planned routing to an end-of-day location, or ii) the recovered flight route terminates within the recovery window requiring an end-of-day location to be specified. The minimum number of aircraft required to terminate at each end-of-day location within the recovery window is enforced through constraints (8.6).

The recovery policy of crew deadheading within a duty is implemented through the surplus crew count constraints (8.7). This set of constraints ensure that crew deadheading is only permitted on flight-copy pairs that are operated by at least one crew. The variables κ^{v+}_j count the number of crew deadheading on flight-copy pair j_v which is penalised in the objective function.

In the IRP, the integration between the crew and aircraft variables is described by the short connection and delay consistency constraints, equations (8.8) and (8.9) respectively. The short connection constraints (8.8) permit the use of connection $(i_u, j_v) \in E^D$ by crew if an aircraft is also using the same connection. The delay consistency constraints (8.9) ensure that the length of delay on any flight in a feasible aircraft recovery solution is identical for the crew recovery solution, and vice versa. This is an improvement upon the common sequential recovery practice, where the delays enforced in one stage may result in subsequent stages being infeasible. The IRP avoids any issues regarding the feasibility of flight delays by simultaneously solving the crew and aircraft recovery problems. Since there exists one delay consistency constraint for each flight-copy pair, this set of constraints grows very quickly with the number of copies. It is on this set of constraints that the row generation procedure is implemented to improve the solution runtime.

The number of crew and aircraft operating the recovered schedule is based upon the originally planned duties and routings. Each crew that is assigned a duty from the planning stage must also be assigned a duty in the IRP or deadheaded back to base, which is captured by constraints (8.10). This is not true for the reserve crew since they are not required to perform any duties during recovery, which is captured by the inequality in constraints (8.11). Similar for crew, each aircraft that is assigned a flight route in the planned solution must be assigned a flight route in recovery, which is given by (8.12).

It is common practice in both the sequential stage and integrated recovery problems to select a subset of crew, aircraft and flights to reduce the problem size and improve solution runtimes. To improve the computational performance of the IRP, the concept of a recovery window has been used to reduce the number of flights included in the optimisation problem. While this provides an upper bound on the optimal recovery cost, it is believed that this approach is realistic and consistent with the objective to quickly return operations to plan. To ensure that all reassignment and rerouting options are available, the full set of crew and aircraft are used in the IRP. The selection of all crew and aircraft for this problem demonstrates that fast solution algorithms are possible on larger data sets using current solution techniques.

8.2 Solution Methodology

In this chapter the IRP is solved using both column generation and column-and-row generation to provide a comparison between the two solution approaches. The column generation solution approach is introduced in Section 4.1 and the general framework for column-and-row generation applied to problems throughout this thesis is presented in Chapter 7. As explained in Chapter 7, there are two key components of the column-and-row generation approach, column generation and row generation, which will be discussed separately in relation to the IRP.

Section 8.2.1 describes the column generation subproblems that are used to identify recovered crew duty and aircraft routing flight strings. The discussion in Section 8.2.1 assumes that the IRP is solved by a standard column generation approach. This is extended in Section 8.2.2 to include row generation, detailing the required modifications to the column generation subproblems. Section 8.2.2 will describe the formulation of the short restricted master problem (SRMP) and the row generation procedure that is employed to solve the IRP.

8.2.1 Column generation

The formulation of the IRP contains two sets of variables for which column generation can be applied. These variables are related to crew duties and aircraft routes, which are defined as flight strings. While each of the variable types have similar structures, there are specific rules governing their generation requiring the implementation of two individual column generation subproblems. In this section the column generation subproblem for each variable type is described, including the relevant solution methods.

In the column generation procedure a restricted master problem (RMP) is defined by including only a subset of all possible columns, P^k and P^r, and is solved to find the optimal dual solution. The dual variables $\boldsymbol{\alpha}^K = \{\alpha_j^K, \forall j \in N^D \cup N_{out}^K\}$ and $\boldsymbol{\alpha}^R = \{\alpha_j^R, \forall j \in N^D \cup N_{out}^R\}$ are defined for the flight coverage constraints (8.2)-(8.3) and (8.4)-(8.5), respectively. The dual variables for the aircraft end-of-day location constraints (8.6) are defined by $\boldsymbol{\epsilon} = \{\epsilon_b, \forall b \in B^R\}$. The dual variables for the surplus crew count constraints (8.7) are given by $\boldsymbol{\eta} = \{\eta_j^v, \forall j \in N^D, \forall v \in U_j\}$. For the short connection constraints (8.8) and the delay consistency constraints (8.9), the dual variables are given by $\boldsymbol{\rho} = \{\rho_{ij}, \forall (i,j) \in E^D\}$ and $\boldsymbol{\gamma} = \{\gamma_j^v, \forall j \in N^D, \forall v \in U_j\}$, respectively. Finally, the dual variables $\boldsymbol{\delta}^K = \{\delta^k, \forall k \in K\}$ and $\boldsymbol{\delta}^R = \{\delta^r, \forall r \in R\}$ are defined for the crew and aircraft assignment constraints, (8.10)-(8.11) and (8.12), respectively. Using the set of optimal dual solutions, the column generation subproblems for crew and aircraft are solved to identify negative reduced cost columns to add to the sets P^k and P^r.

Crew duty subproblem

The crew duty subproblem (PSP^k) is solved as a shortest path problem with one source node and multiple sink nodes. The source node represents a carry-in activity, and the sink nodes represent carry-out activities, including permissible crew and overnight bases. The shortest path must adhere to resource constraints that restrict the number of flying hours and the total hours of the flight string. The decision variables $w_{i_u j_v}^k$ are introduced in the PSP^k that equal 1 if connection (i_u, j_v) is used in the shortest path and 0 otherwise. The connection network used for the shortest path problem is C^R, which permits the possible use of all short connections. The model for the PSP^k is given by,

$$
(\mathrm{PSP}^k) \quad
\begin{aligned}
\hat{c}_p^k = \min\ & RecDutyCost(k, \mathbf{w}^k) - \sum_{(i_u,j_v)\in C^R} \alpha_j w^k_{i_u j_v} - \sum_{(i_u,j_v)\in E^D} \rho_{i_u j_v} w^k_{i_u j_v} \\
& - \sum_{j\in N^D} \sum_{v\in U_j} \sum_{\substack{i_u\in \hat{N}^D \\ |(i_u,j_v)\in C^R}} \left\{ \eta_j^v + \gamma_j^v \right\} w^k_{i_u j_v} - \delta^k, && (8.16) \\
\text{s.t.} \quad & \sum_{\substack{i_u\in \hat{N}^D \\ |(i_u,j_v)\in C^R}} w^k_{i_u j_v} - \sum_{\substack{l_u\in \hat{N}^D \\ |(j_v,l_u)\in C^R}} w^k_{j_v l_u} = 0 \quad \forall j_v \in \hat{N}^D, && (8.17) \\
& \sum_{v\in U_j} \sum_{\substack{i_u\in \hat{N}^D \\ |(i_u,j_v)\in C^R}} w^k_{i_u j_v} \le 1 \quad \forall j \in N^D, && (8.18) \\
& \sum_{u\in U_i} \sum_{\substack{j_v\in \hat{N}^D \\ |(i_u,j_v)\in C^R}} w^k_{i_u j_v} = o_i^k \quad \forall i \in N^K_{in}, && (8.19) \\
& \sum_{j\in N^K_{out}} \sum_{v\in U_j} \sum_{\substack{i_u\in \hat{N}^D \\ |(i_u,j_v)\in C^R}} o_j^k w^k_{i_u j_v} = 1, && (8.20) \\
& \sum_{(i_u,j_v)\in C^R} w^k_{i_u j_v} \omega^n_{i_u j_v} \le \Omega^n \quad n = 1, 2, && (8.21) \\
& w^k_{i_u j_v} \in \{0, 1\} \quad \forall (i_u, j_v) \in C^R. && (8.22)
\end{aligned}
$$

The objective of PSP^k is to find the duty for crew k with the minimum reduced cost by solving a resource constrained shortest path problem (RCSPP). Constraints (8.17) describe the flow balance at each node in the network to ensure a connected path is found. Now, the shortest path problem is formulated on a connection network designed using flight-copies, as such there are many nodes for the same flight but each with a different departure time. Only one departure time is permissible per flight, so a modification to the classic RCSPP is required with constraints (8.18) ensuring that at most one flight-copy per flight is active in the shortest path. The shortest path for crew k must originate from a single carry-in activity, which is achieved by setting $o_i^k = 1$ for a single $i \in N^K_{in}$, 0 otherwise, in constraints (8.19). Also, the termination location of the duty is enforced by constraint (8.20), with all permissible termination locations j captured by the parameter $o_j^k = 1$, and for all locations j that are not permissible $o_j^k = 0$. Constraints (8.21) limit the consumption of each resource along the shortest path with the upper bound on flying hours given by $\Omega^1 = 8$ and on total hours given by $\Omega^2 = 13$. The consumption of each resource on connection (i_u, j_v) is given by $\omega^n_{i_u j_v}, n = 1, 2$.

In the objective function (8.16), the complex cost structure used for crew remuneration is denoted by $RecDutyCost(k, \mathbf{w}^k)$, where $\mathbf{w}^k = \{w^k_{i_u j_v}, \forall (i_u, j_v) \in C^R\}$. This cost structure is identical to equation (3.9), with further definitions for each component provided in this section. Since the PSP^k is solved on a connection network, the consumption of resources is defined for each arc in the network, i.e. the flying hours, $fly = \sum_{(i_u,j_v)\in C^R} w^k_{i_u j_v}\omega^1_{i_u j_v}$ and the total elapsed hours, $elapse = \sum_{(i_u,j_v)\in C^R} w^k_{i_u j_v}\omega^2_{i_u j_v}$. As such, the expression for the cost of a duty in the IRP is given by

$$RecDutyCost(k, \mathbf{w}^k) = \max\{0, \max\{fly(k), f_d \cdot elapse(k), minGuar\} - OrigDutyCost(k)\}, \tag{8.23}$$

where $minGuar$ is set at 6 hours [13] and f_d is a fraction which is airline specific and is set at $f_d = 5/8$ [13].

In consideration to the resource restrictions and complex cost structure, a multi-label shortest path algorithm is required to solve PSP^k. Each label l at node i_u stores the cost of the current shortest path to the node, $\hat{c}_{i_u l}$, the number of flying hours, $H^1_{i_u l}$, and the total elapsed hours, $H^2_{i_u l}$. Now, the connection network described in Section 8.1 forms an acyclic directed graph. Given this network structure, all the nodes can be listed in a topological order, where node i_u is ordered before node j_v if $\exists (i_u, j_v) \in C^R$ [5]. Therefore, the multi-label shortest path algorithm described by Algorithm 4.2 can be implemented to solve the PSP^k.

The description of Algorithm 4.2 stipulates the requirement of a maximum number of labels that can be stored at each node. The implementation of this algorithm for the PSP^k also includes a maximum number of labels for an efficient solution approach. As such, the dominance condition given by Definition 4.1.1 is used to assess the suitability of each label stored at a node.

Aircraft routing subproblem

The column generation subproblem for the aircraft routing variables solves a shortest path problem from a single origination location to one of multiple termination locations or overnight airports. For this problem a set of decision variables $w^r_{i_u j_v}$ are introduced that equal 1 if the connection (i_u, j_v) is used in the shortest path and 0 otherwise. The column generation subproblem for the aircraft routing variables (PSP^r) is defined by,

$$
(\mathrm{PSP}^r) \quad
\begin{aligned}
\hat{c}_p^r = \min \quad & \sum_{(i_u,j_v)\in C^R} c_{i_u j_v}^r w_{i_u j_v}^r - \sum_{(i_u,j_v)\in C^R} \alpha_j w_{i_u j_v}^r - \sum_{b\in B^R} \sum_{\substack{i_u\in\hat{N}^D \\ |(i_u,b)\in C^R}} \epsilon_b w_{i_u b}^r \\
& + \sum_{(i_u,j_v)\in E^D} \rho_{i_u j_v} w_{i_u j_v p}^r + \sum_{j\in N^D} \sum_{v\in U_j} \sum_{\substack{i_u\in\hat{N}^D \\ |(i_u,j_v)\in C^R}} \gamma_j^v w_{i_u j_v}^r - \delta^r, && (8.24) \\
\text{s.t.} \quad & \sum_{\substack{i_u\in\hat{N}^D \\ |(i_u,j_v)\in C^R}} w_{i_u j_v}^r - \sum_{\substack{l_u\in\hat{N}^D \\ |(j_v,l_u)\in C^R}} w_{j_v l_u}^r = 0 \quad \forall j_v \in \hat{N}^D, && (8.25) \\
& \sum_{v\in U_j} \sum_{\substack{i_u\in\hat{N}^D \\ |(i_u,j_v)\in C^R}} w_{i_u j_v}^r \leq 1 \quad \forall j \in N^D, && (8.26) \\
& \sum_{u\in U_i} \sum_{\substack{j_v\in\hat{N}^D \\ |(i_u,j_v)\in C^R}} w_{i_u j_v}^r = o_i^r \quad \forall i \in N_{in}^R, && (8.27) \\
& \sum_{j\in N_{out}^R} \sum_{v\in U_j} \sum_{\substack{i_u\in\hat{N}^D \\ |(i_u,j_v)\in C^R}} w_{i_u j_v}^r = 1, && (8.28) \\
& w_{i_u j_v}^r \in \{0,1\} \quad \forall (i_u,j_v) \in C^R. && (8.29)
\end{aligned}
$$

The objective of the PSP^r is to find a path through the connection network for aircraft r with the minimum reduced cost. Constraints (8.25) ensure flow balance for each node in the network, and the constraints (8.26) ensure at most one departure for each flight is used in the shortest path. Each aircraft is assigned a carry-in activity, which is given by $o_i^r = 1$ for a single $i \in N_{in}^R$, and $o_j^r = 0, \forall j \in N_{in}^R \backslash \{i\}$. The origination of the shortest path from this node is enforced by constraints (8.27). Finally, aircraft are permitted to terminate at any specified carry-out activities or overnight airports, which is captured by constraint (8.28). The termination location for most aircraft is not of great importance, therefore they may terminate at any carry-out activity or overnight airport. If an aircraft is planned to receive maintenance at the end of the day, the termination locations will enforce this requirement.

The PSP^r describes a shortest path problem for which a large number of solution algorithms are available. Similar to the connection network for crew, the network for aircraft is an acyclic directed graph which permits the topological ordering of the nodes. However, the PSP^r only considers a single resource in the shortest path, path distance, therefore Algorithm 4.1 is used to efficiently solve this problem.

In any iteration of the column generation algorithm, the most negative reduced cost for all aircraft, $\hat{c}_p^R$, can be found by solving the PSP^r for each aircraft r and setting $\hat{c}_p^R = \min_{r \in R}\{\hat{c}_p^r\}$. Now, all connection costs and dual variables, except for $\boldsymbol{\delta}^R = \{\delta^r, \forall r \in R\}$, included in (8.24) are aircraft independent. Therefore, by setting $\delta^r = \delta^R, \forall r \in R$, where $\delta^R = \max_{r \in R}\{\delta^r\}$ in (8.24), it is possible to find a lower bound on $\hat{c}_p^R$, labelled as $\tilde{c}_p^R$, by solving this aircraft routing shortest path algorithm only once. The aircraft routing subproblem developed by this method is labelled PSP^R and will be used as part of the row generation procedure described in Section 8.2.2.

8.2.2 Row generation

An important feature of the IRP is the use of a *full* set of recovery policies, which includes flight delays. There are a number of different methods that are available to implement flight delays, such as time windows [81] and discrete flight copies [87], each with relative strengths regarding the problem formulation and solution methods. The technique of flight copies has been selected to model delays as a result of its simplicity in implementation for column generation and to fit within the column-and-row generation framework.

By implementing flight delays using flight copies, a critical consideration of the integrated problem is to ensure that the crew duty and the aircraft routing solutions use the same copy (delay) for each flight. The delay consistency constraints(8.9) capture this. However, this is at the expense of adding a large number of constraints to the restricted master problem (RMP). Since the optimal variables have non-zero coefficients in only a small subset of the delay consistency constraints, many rows related to these constraints are not required in the RMP.

The implementation of delay copies in the IRP provides alternate flight departure times given by a uniform discretisation of a maximum allowable delay. While this is a popular method of implementation that has been employed by Yan and Young [98], Thengvall *et al.* [87] and Andersson and Varbrand [7], Bratu and Barnhart [19] state that a number of copies may be dominated by shorter delay options. Further, Petersen *et al.* [70] suggests the modelling of flight delays by an event-driven approach, linking delays to activities related to each flight. This reduces the size of the recovery problem by only including the delays for each flight that provide feasible connections. While uniform delay options are implemented for the IRP, column-and-row generation provides an optimisation approach to select the most important delay options. While it is possible to implement the recovery flight network reductions as described by [19] and [70], this would simply result in the further enhancement of the column-and-row generation approach.

Comparing the IRP with the RMP in Section 7.3.1, it is clear that the delay consistency constraints (8.9) describe linking constraints similar to (7.4). These constraints provide the link between the primary and secondary sets of variables, which are given by the crew duty and aircraft routing variables in the IRP respectively. While the RMP in Section 7.3.1 describes a problem with multiple secondary variables, the IRP is a special case of this problem class with the aircraft routing variables as the only set of secondary variables. Therefore, the problem structure of the IRP permits the use of the column-and-row generation framework presented in Chapter 7.

The implementation of the column-and-row generation algorithm, Algorithm 7.3 and a description of each feature of this algorithm with respect to the IRP will be provided in this section. As a contribution of this chapter, the column-and-row generation solution approach developed by Muter *et al.* [65] is evaluated against column generation to identify any potential enhancement techniques. A description of the techniques identified by this evaluation will be provided throughout this section.

Formulation of the restricted problems

The column-and-row generation framework requires the formulation of a RMP and SRMP as restrictions on the original problem. The formulation of the RMP is provided in Section 8.2.1 with the SRMP describing a further restriction on the original problem by the elimination of delay consistency constraints (8.9). The SRMP is initialised with all rows related to constraints (8.2)-(8.8) and only a subset of rows for the delay consistency constraints (8.9) as defined by $v \in \bar{U}_j, \forall j \in N^D$. The set $\bar{U}_j$ is initially populated with one copy for most flights j, which is generally the copy representing the scheduled departure time, i.e. $\bar{U}_j = \{0\}$. However, as a result of flight delays caused by the initial disruption it is possible that no feasible connection containing the flight-copy pair j_0 exists. In these situations, the set $\bar{U}_j = \{0, v'\}$ is defined for flight j, where v' represents the copy with the earliest departure time that provides at least one feasible connection for flight j.

The elimination of rows to form the SRMP is coupled with the fixing of variables in the column generation subproblems. This variable fixing is given by $w^k_{i_u j_v} = w^r_{i_u j_v} = 0, \forall u \in U_i \backslash \bar{U}_i, \forall v \in U_j \backslash \bar{U}_j$, which restricts the set of feasible columns. Using this definition, flight delays can be partitioned into those permitted on flight strings (allowable delays) and those which are not permitted (non-allowable delays). Thus, the set of copies $\bar{U}_j$ describes the set of allowable delays for flight j in the current formulation of the SRMP and the non-allowable delays are contained in $U_j \backslash \bar{U}_j$. This distinction between allowable and non-allowable delays is made to describe the restriction on the feasible region through the elimination of constraints from the RMP.

The algorithms implemented to solve the column generation subproblems for the SRMP are identical to those discussed in Section 8.2.1 for the RMP. However, to conveniently describe the variable fixings in the column generation subproblems, the sets $\bar{C}^K \subseteq C^K$, $\bar{C}^R \subseteq C^R$ and $\bar{E}^D \subseteq E^D$ are defined to include only the flight copies contained in $\bar{U}_j, \forall j \in N$. The connection networks, described by $\bar{C}^K$ and $\bar{C}^R$, used in the subproblems for the SRMP are much smaller than that for the RMP, and hence the PSP^k and PSP^r are solved more quickly. Since this restriction on the connection networks limits the set of feasible columns, the solution to the SRMP provides an upper bound on the optimal solution of the IRP.

Row generation algorithm

The row generation algorithm described in Section 7.3.3 involves two key steps, the calculation of an optimal dual solution for the RMP$'$ and identifying favourable rows to add to the SRMP. The results ensuring the accuracy of the dual solution calculation procedure are presented in Section 7.3.2, which are used to develop Algorithm 7.2. The specific details regarding the implementation of Algorithm 7.2 for the IRP will be discussed in this section.

The calculation of the optimal dual solution to the RMP$'$ is a fundamental part of the row generation procedure. By solving the SRMP to optimality using column generation, Theorem 7.3.1 states that the optimal dual solution to the RMP$'$ can be calculated using the solution to the SRMP and Algorithm 7.1. Given the very similar problem structures of the SRMP and the RMP$'$, the dual solutions that must be calculated are related to the rows eliminated to form the SRMP, which are given by $\boldsymbol{\gamma}' = \{\gamma_j^{v'}, \forall j \in N^D, \forall v' \in U_j \backslash \bar{U}_j\}$. The solutions to each of the variables contained in $\boldsymbol{\gamma}'$ are found by executing Algorithm 7.1, solving the PSP^R as the column generation subproblem in step 2. The use of the PSP^R in this algorithm is a problem specific enhancement technique that reduces the runtimes required to calculate the solutions to the dual variables for all eliminated rows.

The second part of the row generation procedure involves using the optimal dual solution to the RMP$'$ to identify favourable rows to add to the SRMP. This process is described by steps 8-11 of Algorithm 7.2, which involves solving the column generation subproblem for the primary variables to find columns feasible for the RMP$'$. Since the primary variables for the IRP are the crew duty variables, the PSP^k is solved as the pricing subproblem in step 8 of this algorithm. The crew duty variables generated by this subproblem describe individual duties for each crew k, hence a larger number of favourable rows can be identified by solving the PSP^k once for each $k \in K$. This is a natural modification of the row generation procedure which is necessary to develop an efficient solution approach for the IRP.

The dual variable estimation of the row generation procedure is a feature of this approach that is identified to be very computationally expensive. Given the set of flight-copy pairs, $\cup_{j \in N^D} U_j \backslash \bar{U}_j$, requires the PSP^R to be solved once for each flight-copy pair contained in this set. Even with the most efficient shortest path algorithm, the large number of executions required to calculate all dual variables can have a significant negative impact on the solution runtimes. Consequently, the number of times that Algorithm 7.2 is executed will affect the overall performance of the column-and-row generation solution process. One approach to address this runtime issue is to vary the number of rows that are added in each call to the row generation procedure. It has been observed that by adding too few rows at each execution requires more calls to the row generation algorithm. Similarly, adding too many rows has the effect of increasing the size of the SRMP too rapidly. A successful approach involves adding more rows to the SRMP based upon the value of the calculated dual variables. This is achieved for the IRP by adding a row for every flight-copy pair which the calculated value of the dual variable is positive. The ideal number of rows to add at each iteration is difficult to determine. However, this approach significantly improves the solution runtimes compared to the standard row generation procedure.

Row generation warm-up

The subset of rows initially included in the SRMP greatly affects the efficiency of the column-and-row generation solution process. In Section 8.2.2, the initialisation of the SRMP involves selecting a single delay copy for each flight which is naïvely set to the scheduled departure time. Ideally, the only rows included in the SRMP should represent the amount of delay for each flight that is required in the optimal solution of the IRP. Unfortunately the optimisation problem to identify the optimal set of delay options is analogous to the original recovery problem, hence an alternative technique is required.

An approach implemented for the IRP uses information from the standard column generation approach to provide a warm-start for column-and-row generation. This approach involves formulating the RMP with all rows from the original problem but only the columns contained in the initial formulation of the SRMP. The RMP is then solved by column generation and in each iteration of the solution algorithm, variables are constructed with no restriction on the allowable delay options. The initial set of delay copies for the SRMP is updated by reviewing each generated flight string p and if $j_v \in p$, then the delay copy v is added to the set $\bar{U}_j$. After n iterations of the column generation solution process, the SRMP is formed to contain only the delay consistency constraints (8.9) described by the sets $\bar{U}_j, \forall j \in N^D$ and all columns in the current formulation of the RMP.

A key feature of this approach is that no additional development work is required and the computational time is equivalent to that of the standard column generation approach. During this warm-up period the runtime of the two solution approaches is identical, therefore the expected runtime improvements are observed by applying column-and-row generation in the succeeding iterations. By retaining the initial columns added during this process, the column-and-row generation approach is provided with a warm-start for the set of columns and rows.

The runtime improvements achieved by this approach demonstrate the importance of an intelligent selection of rows in the initial formulation of the SRMP. This is expected, since this observation is similar to the well known relationship between the initial set of columns and the efficiency of the standard column generation approach. A complicating factor of applying a warm-up period for column-and-row generation is the additional parameter required to specify the number of column generation iterations that must be executed. The value of this parameter has been observed through experiments to greatly affect the efficacy of this approach with an acceptable runtime improvement for the IRP achieved with 20 iterations.

8.2.3 Branching rules

Integral optimality is achieved for the IRP by employing the technique of branch-and-price. This problem includes many different variable types and thus a set of problem specific branching rules have been designed for each. Three different branching rules have been implemented for the IRP, one related to the cancellation or covering of flights and the other two create branches using information from the variable flight strings. The first of the branching rules described here for the IRP is a pure variable branching rule, and the last two are derived from the Ryan/Foster branching technique [76]. By implementing multiple branching rules for the IRP, a priority must be assigned to each dictating when they are used in the solution process. A description of each rule is provided below in the order of their assigned priority.

A key feature of the airline recovery process is the ability to cancel flights in an effort to regain schedule feasibility. This feature appears in the IRP through inclusion of variables z_j in the flight coverage constraints for the crew and aircraft, equations (8.2) and (8.4) respectively. A cancellation variable branching rule is introduced for the IRP which forces the decision of either covering a specified flight on the left branch or cancelling that flight on the right branch. Upon identifying flight j' with the most fractional cancellation variable, $z_{j'}$, branches are created by enforcing $z_{j'} = 0$ on the left branch and $z_{j'} = 1$ on the right branch. The described rule is very simple and fast, and is designed to eliminate fractional cancellation variables early in the branch-and-bound tree.

A very effective branching technique for airline optimisation problems formulated in a set partitioning framework is follow-on branching, as described in Section 4.3.1. The implementation of this branching rule is very similar to that presented in Sections 4.3.1 and 5.2.4, however the multiple sets of variables in the IRP is a complicating factor. A more detailed description of follow-on branching is provided here to explain the identification of branching candidates from multiple variable types.

The implementation of follow-on branching for the IRP identifies the most fractional pair of connected flights for either the crew duty or aircraft routing variables. Since the flight coverage constraints, equations (8.2) and (8.4), are delay independent, each flight is identified without considering the delay copies. By ignoring the multiple copies for each flight, the index i is used to reference all flight-copy pairs $i_u, u \in U_i$ and the connection (i, j) identifies all connections $(i_u, j_v) \in C^K \cup C^R, \forall u \in U_i, \forall v \in U_j$. The set of fractional variables for crew k is defined as $P^k_f = \{p \in P^k | x^k_p \notin \mathbb{Z}\}$, and similarly the set of fractional variables for aircraft r is defined as, $P^r_f = \{p \in P^r | y^r_p \notin \mathbb{Z}\}$. The fractionality of a connection (i, j) is calculated by,

$$\begin{aligned} frac^K_{fOn}(i,j) &= \min\left\{ \sum_{k\in K} \sum_{\substack{p\in P^k_f \\ |(i,j)\in p}} x^k_p, 1 - \sum_{k\in K} \sum_{\substack{p\in P^k_f \\ |(i,j)\in p}} x^k_p \right\} \quad \text{and} \\ frac^R_{fOn}(i,j) &= \min\left\{ \sum_{r\in R} \sum_{\substack{p\in P^r_f \\ |(i,j)\in p}} y^r_p, 1 - \sum_{r\in R} \sum_{\substack{p\in P^r_f \\ |(i,j)\in p}} y^r_p \right\}, \end{aligned} \tag{8.30}$$

for crew and aircraft variables respectively. The connection with the greatest fractionality for either crew or aircraft is identified as (i^*, j^*) and is selected as the branching candidate. The candidate variable type, crew or aircraft, that this branching applies to is also identified by this selection. Upon identifying connection (i^*, j^*) and the candidate variable type, the branching is performed on all variables of this type using the method described in Section 4.3.1.

An alternative branching rule is developed for the IRP that examines the allocation of specific flights to individual crew and aircraft. This branching rule selects a crew group k or aircraft r and enforces or disallows the use of an identified flight-copy. The fractionality of a variable identifier/flight-copy pair, (k, i_u) and (r, i_u), is calculated by,

$$\begin{aligned} frac^K_{flt}(k, i_u) &= \min\left\{ \sum_{\substack{p \in P^k_f \\ |i_u \in p}} x^k_p, 1 - \sum_{\substack{p \in P^k_f \\ |i_u \in p}} x^k_p \right\} \quad \text{and} \\ frac^R_{flt}(r, i_u) &= \min\left\{ \sum_{\substack{p \in P^r_f \\ |i_u \in p}} y^r_p, 1 - \sum_{\substack{p \in P^r_f \\ |i_u \in p}} y^r_p \right\}, \end{aligned} \tag{8.31}$$

for the crew and aircraft variable types respectively. Branching is performed on the variable identifier/flight-copy pair that has the greatest fractionality, as described by the equations (8.31). On the left branch, all variables associated with the identifier, k^* or r^*, must contain the flight-copy $i^*_{u^*}$ in the flight string. On the right branch all flight strings for the variables associated with the identifier, k^* or r^*, must not contain the flight-copy $i^*_{u^*}$.

As a contribution to the column-and-row generation solution method, the row generation procedure is integrated into the branch-and-price framework. Since the branch-and-price algorithm is executed with a subset of all rows contained in the IRP, without allowing the addition of rows throughout this process, any identified lower bounds are potentially greater than the true bound. To avoid this inaccuracy in the solution process, the row generation algorithm is called only at nodes where the column generation procedure concludes with a lower bound greater than the current best bound. By executing the row generation algorithm in these selected situations ensures that the optimal solution is found with branch-and-price and avoids unnecessary executions of this time-costly procedure.

8.3 Computational Results

The computational results demonstrate the benefit of using column-and-row generation (CRG) to solve the IRP compared to a standard column generation approach (Colgen). The following discussion compares these two approaches based upon *computational performance*. For this analysis computational performance is defined as the runtime required to solve the IRP and the final solution quality as measured by the best found solution and the optimality gap. These results justify the use of column-and-row generation as a viable alternative to column generation for solving integrated airline optimisation problems.

8.3.1 Description of data and disruption scenarios

The performance of the IRP is evaluated on a flight schedule provided by an airline consisting of 262 flights, serviced by 48 aircraft of a single fleet type and 79 crew groups. This schedule is designed for a point-to-point carrier servicing 20 airports; 12 of the airports are overnight bases for aircraft and 4 are crew bases. The original duties and routings are generated by solving an integrated airline planning problem with an objective of minimising crew costs and the total number of aircraft operating the schedule.

A set of 16 disruption scenarios are generated as test cases for the IRP. The scenarios describe airport closures at two major airports in the network, occurring in the morning at 6am, 7am, 8am and 9am for either 3 or 5 hours. The numbers used to reference each scenario are provided in Table 8.3. An airport closure imposes a delay on all flights that are scheduled to arrive at or depart from the affected airport until the end of the closure period. In these experiments a recovery window of 6 hours is used, representing the total time allowed to return operations back to plan. The recovery window starts from the reopening of the affected airport, thereby the set of disruptable flights N^D includes all flights departing within a 9 or 11 hour window from the start of the closure. Within the recovery window, the IRP implements a full set of recovery policies including flight delays and cancellations, crew deadheading, the use of reserve crew and the generation of new crew duties and aircraft routes.

Scenario Start Time	6am	7am	8am	9am
Scenario Number	(0,8), **(1,9)**	(2,10), **(3,11)**	(4,12), **(5,13)**	(6,14), **(7,15)**

Table 8.3: Scenario numbers. The bracketed values indicate two different closure durations (3 hours, 5 hours) and bold represents the scenarios related to airport two.

A common approach to improve the runtime of airline recovery problems is to use only a subset of crew, aircraft and flights, which are those identified as disruptable. A key feature of the IRP is the use of the full sets of crew and aircraft to provide the greatest number of duty

Scenario	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$\lvert N^D \rvert$	150	151	149	150	147	145	150	149	182	183	185	186	184	182	184	183

Table 8.4: The number of disruptable flights for each scenario.

and routing options. Now, a recovery window is used to identify the set of flights N^D to include in the IRP and the size of this set for each scenario is documented in Table 8.4. While the use of a recovery window is an approximation of the full recovery problem, it is consistent with the common objective to return operations to plan as quickly as possible. The size of the recovery window dictates the allowable recovery time and at the conclusion of the window no further recovery actions can be taken.

Airlines incur significant realised and unrealised costs due to flight delays and cancellations during the recovery process. These costs are modelled in the IRP to quantitatively define the effects of the disruption on the airline and passengers. The data provided for this problem includes the number of passengers booked on each flight, which is used in the calculation of the delay and cancellation costs. The cost of flight delays has been estimated from the EUROCONTROL report by Cook and Tanner [25], where it is stated that the average cost of delay for a full aircraft is €81 per minute. For convenience, this value is converted into Australian dollars, so the cost of a full aircraft delayed for a minute is $100 AUD.

Flight delays are implemented in the IRP using the technique of flight copies, as described in Section 8.1. The maximum allowable delay on any flight is set at 180 minutes, and 7 flight copies have been used to divide this delay into discrete blocks. Therefore, the minimum possible delay on any flight is 30 minutes, with each subsequent flight copy departure occurring at 30 minute intervals. Since flight delays are discretised with the use of flight copies, the resulting recovery costs are an overestimate of the best possible solution. This occurs because there potentially exist shorter feasible connections within the 30 minute delay window that could provide an improved recovery solution. It is possible to increase the number of flight copies to improve the solution quality. However, the number of delay consistency constraints, equation (8.9), is dependent on the chosen number of copies. Providing a greater granularity of delays with more flight copies results in a much larger column generation master problem and a larger connection network for the pricing subproblem, degrading the computational performance. The results will demonstrate that by using column-and-row generation the improvement in the computational performance over a standard column generation approach is still achieved as the problem size grows with an increased number of copies.

Quantitatively defining the cost of flight cancellations is difficult due to the indirect costs related passenger dissatisfaction. In the event of a flight cancellation, passengers are either i) rebooked onto an alternative flight operated by the airline, ii) rebooked onto a flight operated by a different airline, or iii) provided a refund and some compensation and must rebook their own flight. Case iii) is the most uncommon outcome and results in the greatest passenger dissatisfaction compared to cases i) and ii). However, in all situations it is difficult to estimate the proportion of passengers that are *lost* from potential future bookings with the airline. In these experiments, it is assumed that only the ticket revenue is lost and passengers are not deterred from booking with the airline in the future. The calculation of the total lost revenue for each flight assumes an average ticket price of $350 multiplied by the number of booked passengers. The IRP is solved assuming that the passengers are not rebooked by the airline onto any flights, resulting in the loss of the total expected revenue from the cancelled flight.

This model is implemented in C++ by calling SCIP 3.0.1 [3] to solve the integer program using CPLEX 12.4 as the linear programming solver.

8.3.2 Comparison of solution runtimes

It is of high importance for the practical application of any recovery algorithm that a solution can be found in short runtimes. Figure 8.1 compares the runtime required to solve the IRP when using the solution approaches of column generation and column-and-row generation. To demonstrate the appropriate use of this model in practical applications, a maximum runtime of 1200 seconds (20 minutes) is applied. There is an expected trade-off between solution speed and quality, where reducing the allowable runtime time generally causes an increase in the objective value and the widening of the optimality gap given by the best found solution. Figure 8.1 shows that in the vast majority of experiments, the optimal solution is found with runtimes much less than 1200 seconds. Thus, it is possible to reduce the maximum allowable runtimes without a great impact on the solution quality in most cases.

The results presented in Figure 8.1 demonstrate that the solution to the IRP is achieved much faster using column-and-row generation compared to column generation for all but two cases. Across the 16 scenarios used for the experiments, the average relative improvement in runtimes achieved by column-and-row generation is 27.01%, with a range of -84.3% (scenario 1) to 126.76% (scenario 14). These improvements demonstrate a significant runtime benefit from

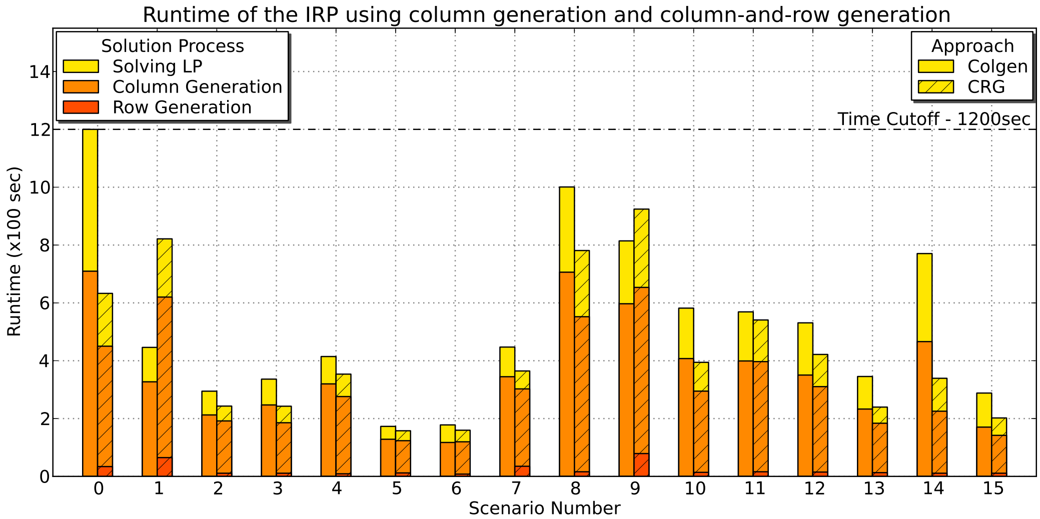

Figure 8.1: The runtimes to solve the IRP for each scenario with a maximum of 1200 seconds (20 minutes). This figure compares the solution approaches of column generation (bars) and column-and-row generation (bars with hatching).

using column-and-row generation to solve the IRP.

A key feature of column-and-row generation is the smaller set of rows used in the formulation of the SRMP compared to the RMP. There is a well known direct relationship between the number of constraints in a problem and the expected time required to solve the linear programming relaxation. Both solution approaches involve a column generation process, which involves solving the LP relaxation of the RMP, and SRMP, each time a set of columns is added. Since the RMP and SRMP are formulated as very similar problems, with the latter containing less constraints, solving the LP for the SRMP requires significantly less simplex iterations resulting in faster execution times.

Figure 8.1 provides a breakdown of the solution runtimes into the processes of LP solve, column generation and row generation. It is clear from this figure that the reduction in the time spent solving the LP is a major component of the solution runtime improvement. Using scenario 12 as an example, solving the LP of the SRMP requires a total of 111.7 seconds for column-and-row generation, where column generation requires 180.22 seconds to solve the LP of the RMP. The total solution runtime improvement for scenario 12 is 109.76 seconds, of which a significant proportion can be attributed to the reduced execution time for solving the LP relaxation. This demonstrates that column-and-row generation significantly improves a solution process which is integral to both approaches.

Another significant improvement in solution runtimes is attributable to the time required during the column generation procedure. Since the reduced set of rows results in a smaller connection network, the column generation subproblems are solved much quicker in the column-and-row generation solution approach. It is observed that the time required for each call to the column generation subproblem is on average 0.2857 seconds quicker for column-and-row generation compared to column generation. The magnitude of this improvement can be explained using scenario 4 as an example, where the column generation subproblems are called 311 times for both solution approaches. For this scenario, column-and-row generation requires 0.1369 seconds less per call, which results in a 42.57 second improvement in the solution runtimes. This runtime reduction is achieved through the variable fixings in the column generation subproblems related to the rows eliminated from the SRMP. It is clear from Figure 8.1 that the required additional process of row generation does not greatly contribute to the runtimes of the column-and-row generation approach. Therefore, there is a significant runtime advantage in solving the LP relaxation and the column generation subproblems from applying column-and-row generation.

Effects of problem size

The number of rows initially removed from the RMP to form the SRMP is directly proportional to the number of flight copies used in the model. An increase in the number of flight copies impacts the two competing factors affecting the runtime of the column-and-row generation approach, i.e. the smaller problem size and the row generation algorithm. With a greater number of flight copies, the SRMP initially defines a problem much smaller than the related RMP, potentially providing a considerable speed up in the runtime required for each LP solve. However, the more flight copies that are removed may require additional executions of the row generation procedure to identify the optimal set of rows, having a negative effect on the runtime of the column generation subproblems.

Figure 8.2 displays the relative difference in runtimes between the column generation (x) and column-and-row generation (y) approaches by varying the number of flight copies. The relative difference is calculated by $(x - y)/\min\{x, y\}$, reporting the difference in runtimes relative to the best performing solution approach. For example, a relative difference of 25% indicates that the runtime of column-and-row generation is 80% of that achieved by column generation. Conversely, a relative difference of -50% indicates that column generation solves the IRP in 66.67%

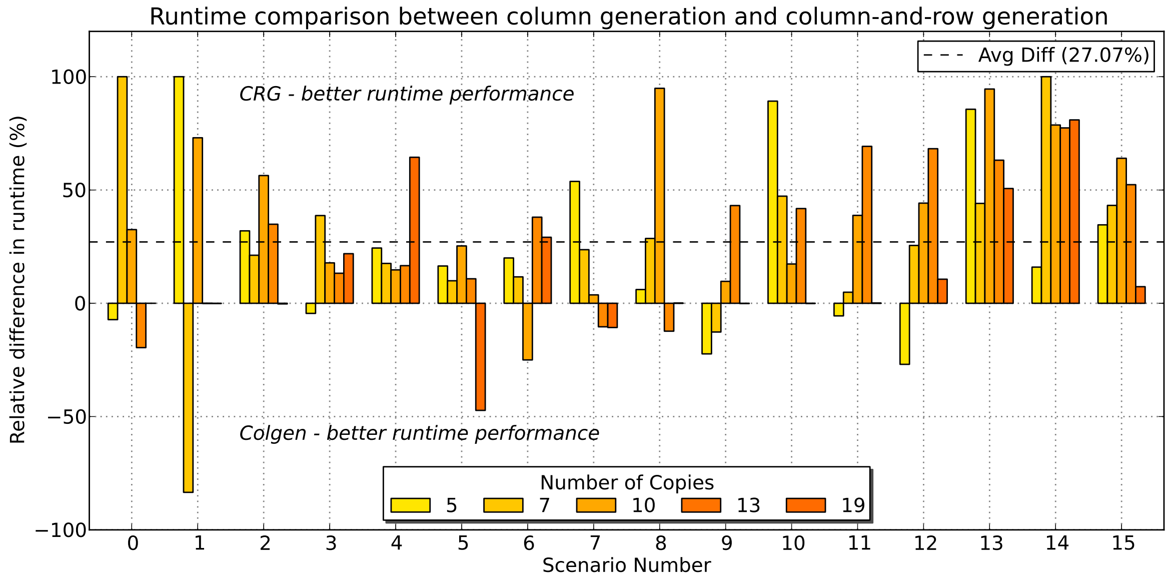

Figure 8.2: The relative difference in runtimes between column generation (x) and column-and-row generation (y) using different sets of flight copies. The values in the figure are calculated by $(x - y)/\min\{x, y\}$ with a maximum reported improvement capped at 100%. Note: maximum runtime of 1 hour was used for these results.

of the runtime required by column-and-row generation. The results in Figure 8.2 demonstrate that across all experiments performed, column-and-row generation outperforms the column generation approach in most cases, with an average relative improvement of 27.07%.

The results in Figure 8.2 demonstrate a better average runtime performance for column-and-row generation compared to column generation. While this is true for the average case, there are many individual experiments where the reverse result is observed. This generally occurs when the column-and-row generation approach requires more branching to identify the optimal integer solution. For example, scenario 1 formulated with 7 flight copies is solved significantly faster with column generation due to the column-and-row generation approach requiring 29 more nodes in the branch-and-bound tree. While the LP of the root node is solve much faster by column-and-row generation (327 seconds compared to 439 seconds), the branch-and-price algorithm has more difficultly converging to integrality for the SRMP. The structure of the SRMP appears to affect the efficacy of the branching rules and the performance of the primal heuristics. This is a common observation from the experiments in this chapter regarding the implementation of column-and-row generation within the branch-and-price framework.

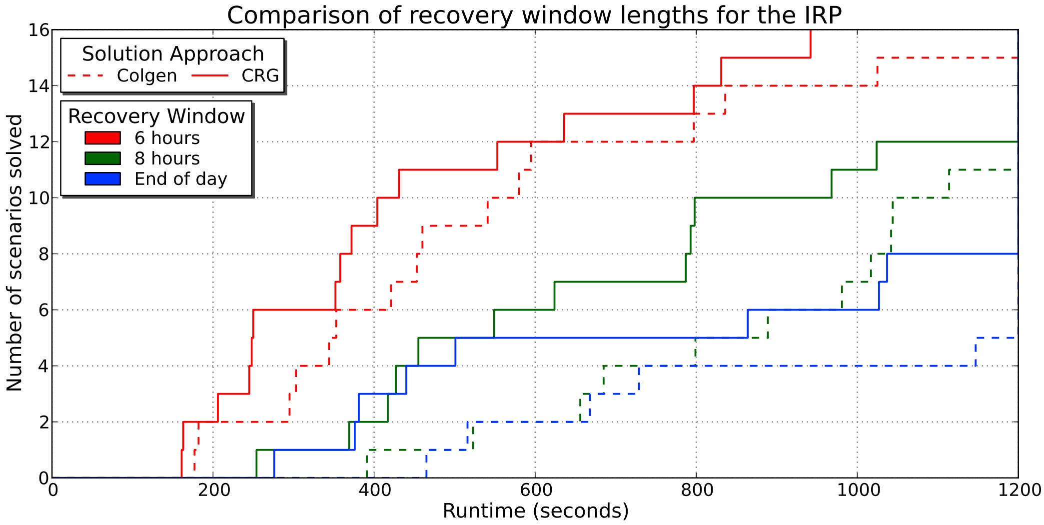

Figure 8.3: Time required to solve scenarios with different recovery window lengths. The recovery windows are set at 6 hours (red), 8 hours (green) and until the end of the day (blue). A maximum runtime of 1200 seconds (20 minutes) and 7 flight copies per flight are used

The length of the recovery window is another feature of the IRP that affects the problem size by directly impacting the number of flights included in the set N^D. As the length of the recovery window increases, the number of flights that depart within that period also increases. A larger set of disruptable flights N^D has two main effects on the IRP, i) an increase in the number of constraints in both the RMP and SRMP, and ii) an increase in the size of the connection network used in the column generation subproblems.

Figure 8.3 presents the time required to solve all of the scenarios used in the experiments with different recovery window lengths, 6 hours, 8 hours and until the end of the day. An interesting observation from this figure is the runtimes required to solve all scenarios using a window of 6 hours is significantly shorter than when the other two window lengths are used. This indicates that the increase in the size of N^D has a great effect on the solution runtime, which is very evident when the recovery window is extended from 6 to 8 hours.

It is observed in Figure 8.3 that the increased problem complexity from extending the recovery window has a more pronounced effect on the runtime of the column generation approach. This is evident from the significant decrease in the number of scenarios that column generation solves to optimality within the runtime of 1200 seconds. Also, the separation of the frontiers produced by the two solution approaches using an 8 hour window demonstrates the greater degradation of runtime performance from column generation.

Analysing enhancement techniques

A contribution of this chapter is the explicit evaluation of the column-and-row generation approach against column generation and the development of specific enhancement techniques. While the techniques introduced in Section 8.2.2 are presented in relation to the IRP, they may be applied in any implementation of column-and-row generation. The most important enhancement techniques discussed are the number of rows added during the row generation procedure and the row warm-up technique used to provide an initial formulation of the SRMP with a meaningful set of delay options. Figure 8.4 presents the relative difference in solution runtimes for column-and-row generation implemented with different enhancements compared to a standard column generation approach.

The strength of the column-and-row generation approach to improve upon solution runtimes of column generation is evident in Figure 8.4. This is demonstrated by all implementations of column-and-row generation achieving an improvement in the solution runtimes compared to the standard column generation approach. In particular, column-and-row generation implemented without any enhancement improves upon the solution runtimes of column generation by 11.91%. This result also outperforms the implementations of column-and-row generation using the additional rows or warm-up enhancements in isolation. This is surprising since the enhancements are expected to improve upon the runtime of the standard implementation, not

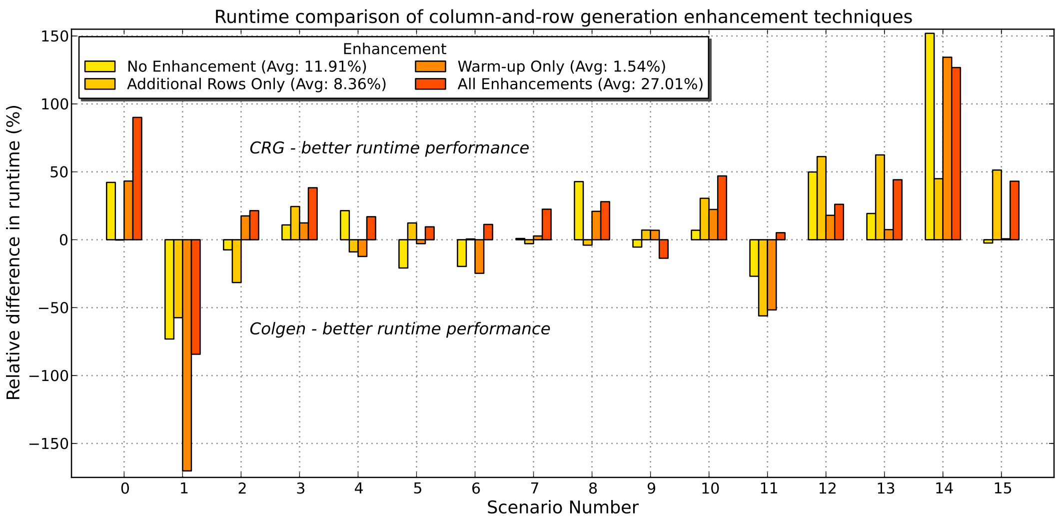

Figure 8.4: The relative difference in runtimes between column generation (x) and column-and-row generation implemented with different enhancement techniques (y). The values in the figure are calculated by $(x - y)/\min\{x, y\}$.

degrade the performance. However, the runtimes using the individual enhancements are still shorter on average than that achieved by the standard column generation approach.

While each enhancement implemented individually degrades the performance of the column-and-row generation approach, significant runtime improvements are observed by their combined use. This is demonstrated in Figure 8.4, implying that a strong relationship exists between the two enhancement approaches. In this figure, column-and-row generation using all enhancements demonstrates runtimes that are 27.01% shorter on average when compared to the standard column generation approach. Comparing this result to the standard implementation of column-and-row generation, the use of all enhancements improves runtimes by 15.88%. This exhibits a significant runtime improvement from the use of enhancement techniques in the implementation of column-and-row generation.

The experiments performed in this section demonstrate that the greatest difference in the efficiency of the solution approaches is the convergence to the integer optimal solution. In each of the figures presented above, the largest variations in solution runtimes is commonly caused by an increased number of nodes in the branch-and-bound tree. This effect is observed in a number of results presented in Figure 8.4. The use of enhancement techniques greatly reduces the runtime required to solve the root node with ineffective branching eroding any gains. For example, the implementation of column-and-row generation with all enhancements outperforms column generation in the time required to solve the root node, with scenarios 1 and 9 being solved 110 and 153 seconds faster respectively. These two scenarios display the greatest improvement in the root node solving time but the integer optimal solution is found quicker by column generation. While column-and-row generation achieves an improvement in solution runtimes, these results demonstrate that greater reductions can be achieved through more effective branching techniques.

8.3.3 Analysis of solution quality

While the speed of solution is critical for airline recovery problems, there is also a high importance placed on the quality of the best solution found. The solution quality of the IRP is assessed by the optimality gap that is achieved at the termination of the maximum allowable runtime. The results presented in Figure 8.1 display the time required for the IRP to achieve an optimality gap of 1%, given a maximum runtime of 1200 seconds. While column-and-row generation solves all scenarios within 1200 seconds, this runtime may be prohibitive for practical

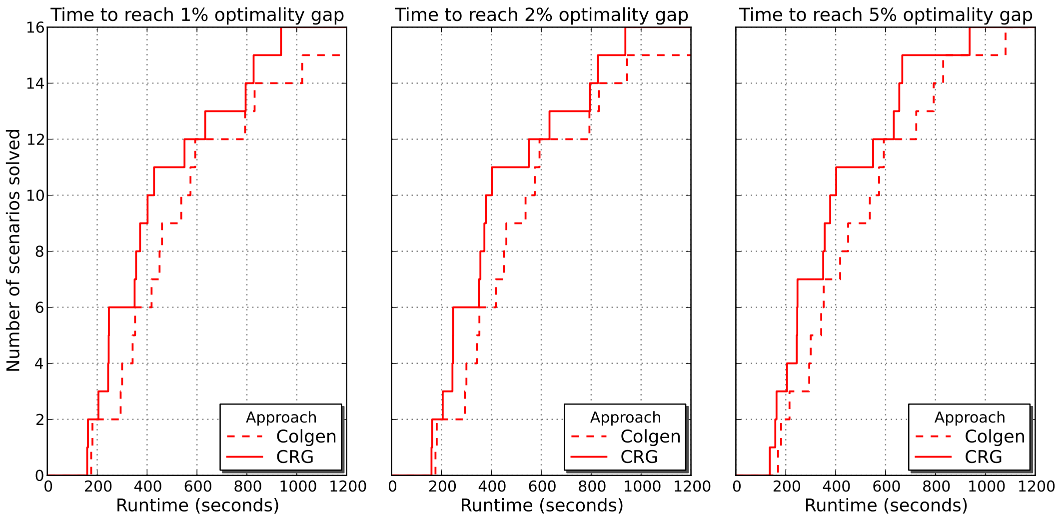

Figure 8.5: Time for the IRP to solve to within a 1%, 2% and 5% optimality gap. A maximum runtime of 1200 seconds is used for these experiments. A comparison between column generation (dashed line) and column-and-row generation (solid line).

application of the solution algorithm.

Figure 8.5 presents the time required to achieve an optimality gap of 1%, 2% and 5% within a maximum runtime of 1200 seconds. This figure demonstrates that reducing the solution runtime greatly affects the ability of both column generation and column-and-row generation to solve each scenario to within a 1% optimality gap. Therefore it is necessary to reduce the solution quality in order to improve the solving rate of the two approaches. While the frontiers in each figure of Figure 8.5 are similar, there are critical points where the increased optimality gap permits the early termination of the solution algorithm. This is observed for the column-and-row generation approach at 300, 400 and 700 seconds where the number of scenarios terminating within the maximum allowable runtime increases by at least one for the 2% and 5% optimality gaps. This is also observed for the column generation approach, however the effect is not as pronounced.

It is clear from Figure 8.5 that as the maximum allowable runtime decreases, the number of scenarios that are solved to optimality for both solution approaches also decreases. The column-and-row generation is less affected by the reduction in runtimes as demonstrated by the frontier in this figure dominating the frontier achieved by the column generation approach. This indicates that column-and-row generation achieves a faster convergence to all displayed optimality gaps, resulting in higher quality solutions early in the solution process.

8.3.4 Analysis of recovery statistics

The composition of the recovery policies in the solution to the IRP greatly affects passenger satisfaction and the acceptability to the operations control centre. The main recovery policies that are implemented for the IRP are flight delays and cancellations and the crew-specific policies of deadheading and the use of reserve crew. Figure 8.6 demonstrates the use of each of these recovery policies in the solutions achieved by column generation and column-and-row generation. It is important to note that scenario 0 is not solved to optimality by column generation, therefore the recovery solution is expected to present a greater use of recovery policies compared to generating recovered crew duties and aircraft routes.

Column generation and column-and-row generation both solve the IRP to within an optimality gap of 1% for most scenarios. Hence, very little difference between the solutions will be observed. This is demonstrated in Figure 8.6 with only subtle variations in the use of the reported recovery policies. The most significant difference in the solutions is given by the number of deadheaded crew, with the column generation solution exceeding column-and-row generation solution in four of the scenarios examined compared to two in the reverse. A feature of the solution approaches affecting the composition of recovery policies is related to the construction of flight strings for the SRMP and RMP. Since the SRMP contains less rows than the RMP throughout the solution process, there are many columns that are initially not permissible in the

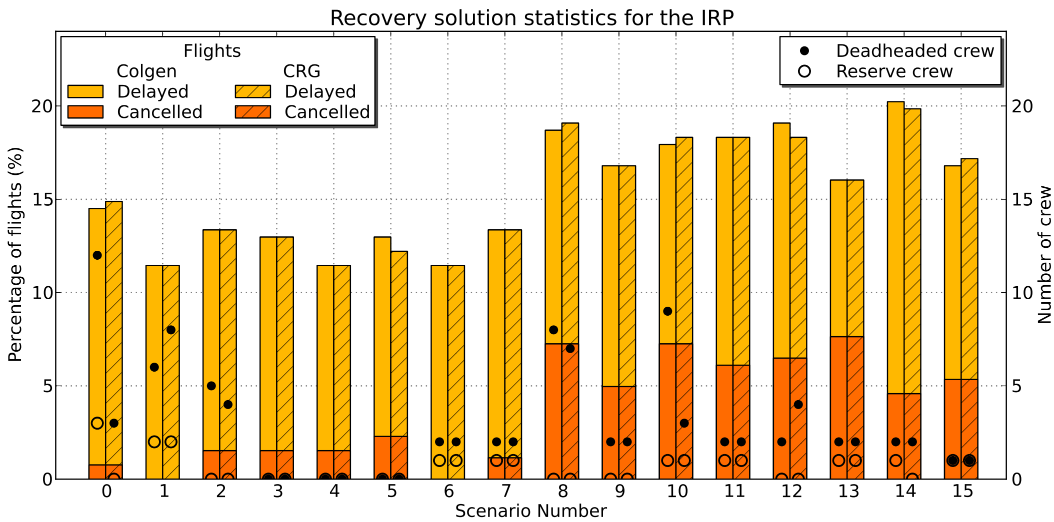

Figure 8.6: Comparison of recovery statistics from the column generation (bars) and column-and-row generation (bars with hatching) solutions. The left and right axes are related to flight statistics and crew specific recovery policies respectively.

column-and-row generation approach. This results in feasible integer solutions with minimal delay being added to the solution pool in the column-and-row generation approach, promoting the use of alternative recovery policies. Consequently, the different sets of columns and feasible solutions found during the solving process can impact upon the composition of recovery actions in the optimal solution.

It is clear from Figure 8.6 that the magnitude of the disruptions simulated in scenarios 8-15 is much greater than that in scenarios 0-7. This is shown by the number of flight delays and cancellations that are made using both solution approaches. In particular, a significant number of disrupted flights in scenarios 8-15 are due to cancellations. The prevalence of each recovery action in the optimal solution to the IRP is controlled through the use of penalty parameters as described in Section 8.3.1. While it is trivial to modify these parameters to adjust the composition of the recovery actions in the optimal solution, it is important to note that this will also affect the solution runtimes of both column generation and column-and-row generation.

8.4 Conclusions

This chapter presents column-and-row generation as an alternative approach for solving the integrated airline recovery problem. The integrated recovery problem is a very large and computationally difficult problem for which there have been many attempts to develop efficient solution methods. The use of column-and-row generation to solve the IRP has been demonstrated to significantly improve the solution runtime and quality compared to a standard column generation approach.

The column-and-row generation methodology applied for the IRP is based on the work by Muter *et al.* [65], who presented a framework to solve large scale linear programs with column-dependent-rows. Due to the structure of the IRP, a branch-and-price procedure is required to achieve integral optimality. The optimality of the problem solved by column-and-row generation is ensured by calling the row generation procedure at selected nodes within the branch-and-bound tree. The treatment of the row generation procedure in the branch-and-price framework for the IRP extends the current column-and-row generation approaches and improves upon the efficiency of the solution process.

A motivation for applying column-and-row generation to solve the IRP is to reduce solution runtimes. The results in Section 8.3 demonstrate that across the majority of the experiments, column-and-row generation outperforms column generation in solution runtime and quality. The improvement in runtimes is observed through a decrease in the time required for each LP solve in the column-and-row generation procedure. The number of flight copies and the length of the recovery window has a direct effect on the size of the IRP, impacting the solution runtimes. The results demonstrate that as the problem size increases, column-and-row generation still achieves significant improvements over a standard column generation approach. The improvements achieved through the use of column-and-row generation demonstrate a practical solution approach for the integrated recovery problem.

Section 8.3 explicitly evaluates the column-and-row generation approach against standard column generation. This evaluation of the framework presented by Muter *et al.* [65] has not previously been published, and the results demonstrate a significant improvement in solution runtimes. From this evaluation a number of enhancement techniques have been developed which are applicable to any implementation of column-and-row generation. In particular, it is demonstrated that the addition of extra rows during the row generation procedure and the use of a warm-up period achieves the greatest improvement for the column-and-row generation approach when implemented in combination.

Column-and-row generation provides a direct solution approach for the IRP that achieves near optimal solutions within the desired time-frame. This is a significant improvement on alternative solution approaches where integral optimality is not guaranteed, such as Benders' decomposition. Further, at each iteration of the column-and-row generation approach the optimal solution to the SRMP provides an upper bound on the original problem. This is a significant advantage of this approach, permitting the early termination of the solution algorithm.

While the application of column-and-row generation reduces the runtime of the IRP, this improvement is not great enough to implement this algorithm into the recoverable robustness framework. Recoverable robustness requires the explicit evaluation of a planned solution against numerous recovery scenarios during the solution process. Unfortunately, the results for the IRP demonstrate runtimes that are prohibitively large to efficiently solve the recoverable robust integrated aircraft and crew scheduling problem.

The improvements in the runtime for the IRP using column-and-row generation are encouraging for the real-world application of this approach. This integrated recovery model solves the key elements of the complete recovery process, schedule, aircraft and crew. Another critical aspect of the recovery problem is the consideration of passengers, which is neglected in the formulation of the IRP. While passengers do not contribute a direct cost to an airline, the indirect costs associated with good-will can have considerable effects on future revenue. A natural extension to the IRP is the consideration of passenger flows following the cancellation of flights. A novel approach to considering passengers in the integrated recovery problem is presented in the following chapter.

Chapter 9

Integrated Airline Recovery Problem with Passenger Reallocation

The quality of the feedback from the evaluation stage of the recoverable robustness framework significantly affects the efficacy of the approach. In Chapters 5 and 6, high quality feedback is achieved by employing a full set of recovery options. A full set of recovery options is also employed for the integrated airline recovery problem (IRP) investigated in Chapter 8 for use in the evaluation stage of the recoverable robustness framework. A feature omitted from the recovery problems developed in Chapters 5, 6 and 8 is the consideration of passenger flows following a disruption. Passenger recovery is only considered through the costs of flight delays and cancellations, neglecting the possibility of reallocating passengers to alternative flights. Considering passengers in the evaluation stage of the recoverable robustness framework is expected to significantly improve the feedback quality, and hence the recoverability.

As explained in Chapter 3, there are very few examples in the literature with a specific focus on passengers during the recovery process. Since passenger recovery occurs as the final stage of the sequential recovery process, it is necessary to integrate this problem with other stages to improve operational performance. Passenger recovery is most commonly integrated with the aircraft recovery problem [19, 48] or within a complete integrated framework [55, 70]. The passenger recovery problem attempts to identify new itineraries for each passenger that is disrupted during recovery. While this formulation may be necessary for some airlines, point-to-point carriers more commonly operate single flight itineraries, hence this problem description is unnecessarily complex. This chapter introduces a novel modelling approach for passenger recovery to reaccommodate passengers on alternative flights in the event of a flight cancellation.

The integrated airline recovery problem with passenger reallocation (IRP-PR) is a direct extension of the IRP presented in Chapter 8. This extension permits the evaluation of the passenger recovery approach developed in this chapter by comparing the solutions of the IRP and IRP-PR. In addition, column-and-row generation was presented in the previous chapter as an effective method to improve the runtimes of the IRP compared to a standard column generation approach. Since the IRP-PR displays a similar problem structure to the IRP, column-and-row generation is also applied in this chapter.

The contributions of this chapter are:

1. the introduction of a novel modelling technique for passenger recovery,

2. further evaluation of the column-and-row generation solution approach.

The most important contribution of this chapter is the modelling of flight cancellation variables to prescribe alternative travel arrangements for disrupted passengers. This is a simple but novel modification to the IRP, which is very effective in reducing the operational costs of the airline. To the best of the author's knowledge, this modelling approach has not previously been considered in airline recovery problems. Secondly, the experiments in Chapter 8 provide an explicit evaluation of the column-and-row generation framework developed by Muter *et al.* [65], which has not been previously performed. This chapter extends that contribution by evaluating the application of this solution approach to a more complex and difficult optimisation problem.

This chapter presents the IRP-PR as an extension of the IRP by considering passenger reallocation options in the event of a flight cancellation. The problem description is provided in Section 9.1 and for conciseness only the features specific to the IRP-PR will be discussed. The IRP-PR is solved by column-and-row generation and a description of its implementation will be given in Section 9.2. Finally, the computational experiments for the IRP-PR are presented in Section 9.3. The results will demonstrate the potential reduction in operational costs that can be achieved by considering passenger reallocation during recovery. In addition, an evaluation of the column-and-row generation solution approach against column generation will be provided.

9.1 Integrated Airline Recovery Problem with Passenger Reallocation

The integrated airline recovery problem with passenger reallocation (IRP-PR) integrates the schedule, aircraft and crew recovery problems with consideration to passenger flows through the recovered network. The integration of aircraft and crew recovery ensures that the optimal solution to both problems is found with consistent flight delay and cancellation decisions. The IRP-PR explicitly considers passengers by modelling the alternative travel arrangements for passengers booked on cancelled flights.

The IRP-PR is modelled with three major types of decision variables which are described as the aircraft, crew and cancellation variables. The aircraft and crew variables detail the movement of these resources through the flight network. The cancellation variables provide the reallocation options for the passengers booked on cancelled flights. The three variable types are linked in the IRP-PR by the flight delay and cancellation decisions and specific flights allocated to each aircraft and crew. An important aspect of the IRP-PR is the use of all aircraft and crew resources in the recovery problem, allowing for the optimal allocation of all available resources.

The notation and model descriptions provided in Chapter 8 are relevant to the formulation of the IRP-PR. As such, only the notation required to describe the passenger reallocation in the integrated recovery problem along with a detailed description of the modelling approach is provided in this chapter. The additional notation used to describe the IRP-PR is presented in Table 9.1

9.1.1 Cancellation variables

A major contribution of this chapter is the modelling of passenger flow in the recovery problem using the flight cancellation variables. A common approach used to describe the cancellation of flights is with a single variable z_j that equals 1 to indicate the cancellation of flight j and 0 otherwise. This modelling approach is employed in the recovery tail assignment problem (RTAP) and the crew duty recovery problem (CDRP) in Chapter 3 and for the IRP in Chapter 8. In the IRP-PR, the variable z_{jp} is defined to indicate the cancellation of flight j but also the reallocation of passengers to alternative flights as given by scheme p. Since there are multiple reallocation options following the cancellation of each flight, the set P^j is defined to contain each passenger reallocation scheme p available following the cancellation of flight j.

P^j	is the set of all schemes p describing the reallocation of passengers following the cancellation of flight j
N^j	is the set of flights i departing earlier than flight j within a specified time window. Flights i and j have the same O-D pair, $N^j \subseteq N$
N_i^{post}	is the set of flights j departing later than flight i within a specified time window. Flights i and j have the same O-D pair, $N_i^{post} \subseteq N$
z_{jp}	$= 1$ if the flight j is cancelled and reallocation scheme p is used, 0 otherwise
d_{jp}	$=$ the cost of cancelling flight j and using reallocation scheme p
h_{ijp}^v	is the number of passengers from flight i reallocated to flight j departing on copy v in scheme p
$Maxcap$	is the total passenger capacity of the aircraft
$Pax(j)$	is the number of passengers originally booked on flight j
r_j	is the number of seats on flight j available for passenger reallocation, $r_j = Maxcap - Pax(j)$

Table 9.1: Additional notation for the IRP-PR.

To conveniently describe the passenger reallocation schemes in the IRP-PR, the sets N_i^{post} and N^j are defined to contain the alternative flights for disrupted passenger. The alternative flights for passengers following the cancellation of flight i are identified by having the same origin-destination (O-D) pair as i and depart within a time window commencing from the departure time of i. Using this definition, the set N_i^{post} contains all flights j that are travel alternatives for passengers booked on flight i. Conversely, the set N^j contains all flights i for which flight j is a travel alternative.

Each of the reallocation schemes p for the cancellation variables can be likened to the solution of a knapsack problem. For example, following the cancellation of flight i the size of the knapsack is given by the number passengers booked on that flight, $Pax(i)$, and the items to enter the knapsack are the available seats $r_j = Maxcap - Pax(j)$ on alternative flights $j \in N_i^{post}$. In the model constraints, the number of passengers reallocated to the flight-copy pair j_v following the cancellation of flight i on the reallocation scheme p is given by h_{ijp}^v.

The cost of a reallocation scheme, d_{jp}, attempts to quantitatively describe the impact of a flight cancellation on passenger satisfaction. For this modelling approach, the cost of a flight cancellation considers the delay experienced by the reallocated passengers and the number of passengers not provided with alternative travel arrangements. The cost of reallocating passengers to alternative flights is estimated to be equivalent to a passenger experiencing a flight delay of the same duration. The cost of not providing alternative travel arrangements is identical to the standard flight cancellation modelling approach, which includes the revenue lost and a measure of passenger dissatisfaction.

The mathematical model of the IRP-PR is given by,

$$
\text{(IRP-PR)}\quad
\begin{aligned}
\min \quad & \sum_{k \in K} \sum_{p \in P^k} c_p^k x_p^k + \sum_{j \in N^D} \sum_{v \in U_j} g^{DHD} \kappa_j^{v+} + \sum_{k \in K} g^{DHB} \nu_k \\
& + \sum_{r \in R} \sum_{p \in P^r} c_p^r y_p^r + \sum_{j \in N^D} \sum_{p \in P^j} d_{jp} z_{jp}, && (9.1) \\
\text{s.t.} \quad & \sum_{k \in K} \sum_{p \in P^k} a_{jp}^k x_p^k - \sum_{v \in U_j} \kappa_j^{v+} + \sum_{p \in P^j} z_{jp} = 1 \quad \forall j \in N^D, && (9.2) \\
& \sum_{k \in K} \sum_{p \in P^k} a_{jp}^k x_p^k = 1 \quad \forall j \in N_{out}^K, && (9.3) \\
& \sum_{r \in R} \sum_{p \in P^r} a_{jp}^r y_p^r \leq 1 \quad \forall j \in N^D, && (9.4) \\
& \sum_{r \in R} \sum_{p \in P^r} a_{jp}^r y_p^r = 1 \quad \forall j \in N_{out}^R, && (9.5) \\
& \sum_{r \in R} \sum_{p \in P^r} o_{bp}^r y_p^r \geq \sum_{r \in R} \bar{o}_b^r \quad \forall b \in B^R, && (9.6) \\
& \sum_{k \in K} \sum_{p \in P^k} a_{jp}^{kv} x_p^k - \kappa_j^{v+} + \kappa_j^{v-} = 1 \quad \forall j \in N^D, \forall v \in U_j, && (9.7) \\
& \sum_{k \in K} \sum_{p \in P^k} e_{ijp}^k x_p^k - \sum_{r \in R} \sum_{p \in P^r} e_{ijp}^r y_p^r \leq 0 \quad \forall (i,j) \in E^D, && (9.8) \\
& \sum_{k \in K} \sum_{p \in P^k} a_{jp}^{kv} x_p^k - \kappa_j^{v+} - \sum_{r \in R} \sum_{p \in P^r} a_{jp}^{rv} y_p^r = 0 \quad \forall j \in N^D, \forall v \in U_j, && (9.9) \\
& \sum_{k \in K} \sum_{p \in P^k} r_j a_{jp}^{kv} x_p^k - r_j \kappa_j^{v+} - \sum_{i \in N^j} \sum_{p \in P^i} h_{ijp}^v z_{ip} \geq 0 \quad \forall j \in N^D, \forall v \in U_j, && (9.10) \\
& \sum_{p \in P^k} x_p^k + \nu_k = 1 \quad \forall k \in K \backslash K^{res}, && (9.11) \\
& \sum_{p \in P^k} x_p^k \leq 1 \quad \forall k \in K^{res}, && (9.12) \\
& \sum_{p \in P^r} y_p^r = 1 \quad \forall r \in R, && (9.13) \\
& x_p^k \in \{0,1\} \ \forall k \in K, \forall p \in P^k, \quad y_p^r \in \{0,1\} \ \forall r \in R, \forall p \in P^r, && (9.14) \\
& z_{jp} \in \{0,1\} \ \forall j \in N^D, \forall p \in P^j, \quad \nu_k \in \{0,1\} \ \forall k \in K, && (9.15) \\
& \kappa_j^{v+} \geq 0, \kappa_j^{v-} \geq 0 \quad \forall j \in N^D, \forall v \in U_j. && (9.16)
\end{aligned}
$$

The alternative modelling approach used for the cancellation variables requires significant modification to the constraints of the IRP to form the IRP-PR. In the IRP-PR, a cancellation variable is defined for each passenger reallocation scheme p following the cancellation of flight j, z_{jp}. As such, the single cancellation variable z_j in the objective and constraints of the IRP must be replaced by the summation over all reallocation schemes p for flight j given by $\sum_{p \in P^j} z_{jp}$. In addition, the cancellation variables are eliminated from the flight coverage constraints for aircraft (9.4), which is then modified to be an inequality constraint. This modification is possible since the consistency of flight cancellations is enforced by other constraints in the model. By eliminating the cancellation variables from constraint (9.4), the IRP-PR presents a model formulation that has a structure identical to the general problem presented in Section 7.3.1. This permits the use of the column-and-row generation framework presented in Chapter 7 to solve the IRP-PR.

The passenger reallocation approach developed in this chapter also requires the addition of constraints to model the rebooking of passenger onto alternative flights. If flight i is cancelled, passengers can be rebooked onto flights contained in the set N_i^{post}. The rebooking process must ensure that the alternative flights are operated by crew and aircraft in the recovered schedule and that the delays on the reallocation options are respected. In addition, an upper bound on the number of passengers reallocated to flight j is given by $r_j = Maxcap - Pax(j)$. These conditions are enforced with the passenger reallocation constraints (9.10), providing further integration between the crew and cancellation variables.

9.2 Solution Methodology

Airline operations are very dynamic with the current state of the system changing almost every minute. For recovery problems to be implemented in practice, it is necessary to achieve high quality solutions in very short runtimes. Column-and-row generation has been demonstrated in Chapter 8 to achieve this requirement for the IRP. The extension presented by the IRP-PR introduces additional variables and constraints that further increases the complexity of the recovery problem, potentially resulting in the longer solution runtimes. As explained above, the problem structure of the IRP-PR permits the use of the general framework presented in Chapter 7. Therefore, it is expected that solving the IRP-PR by column-and-row generation achieves similar improvements in the solution runtimes and quality compared to a standard column generation approach.

The column-and-row generation solution approach will be discussed in relation to the key components of column generation and row generation. Section 9.2.1 will discuss the column generation approach applied to solve the IRP-PR. This description is provided assuming that the IRP-PR is solved by column generation alone. A particular focus of Section 9.2.1 is the column generation subproblem for the cancellation variables. Finally, Section 9.2.2 will describe the application of the column-and-row generation framework developed in Chapter 7 to the IRP-PR.

9.2.1 Column generation

There are three variable types that column generation is applied to in the IRP-PR, the crew, aircraft and cancellation variables. The crew and aircraft variables are defined as variable flight strings, which are dynamically generated by solving a shortest path problem. The cancellation variables detail passenger reallocation schemes which are described in Section 9.1.1 as the solution to a bounded knapsack problem. The column generation subproblems for the crew and aircraft variables are presented in detail in Section 8.2.1, so only the key features will be discussed here. To provide a complete overview of the modelling approach for the cancellation variables, a detailed description of the cancellation variable subproblem will be presented in this section.

The restricted master problem (RMP) is formulated to contain a subset of all possible variables that are included in the IRP-PR. This involves defining $\bar{P}^k \subseteq P^k$, $\bar{P}^r \subseteq P^r$ and $\bar{P}^j \subseteq P^j$ for the crew, aircraft and cancellation variables respectively to form a smaller, more tractable optimisation problem. The column generation subproblems are solved to identify any columns favourable to the original problem, which are then added to the sets $\bar{P}^k$, $\bar{P}^r$ and $\bar{P}^j$. This provides an iterative process that continues until no further favourable columns are identified, indicating that the current solution to the RMP is the optimal solution to the original problem.

The column generation procedure involves identifying the minimum reduced cost variables for crew, aircraft and flight cancellations by solving respective subproblems with the current optimal dual solution to the RMP. The dual variables for the flight coverage constraints (9.2)-(9.3) and (9.4)-(9.5) are defined as $\boldsymbol{\alpha}^K = \{\alpha_j^K, \forall j \in N^D \cup N^K_{out}\}$ and $\boldsymbol{\alpha}^R = \{\alpha_j^R, \forall j \in N^D \cup N^R_{out}\}$ respectively. The dual variables $\boldsymbol{\epsilon} = \{\epsilon_b, \forall b \in B^R\}$ are defined for the aircraft end-of-day location constraints (9.6). The dual variables for the surplus crew count constraints (9.7) are defined by $\boldsymbol{\eta} = \{\eta_j^v, \forall j \in N^D, \forall v \in U_j\}$. The dual variables for the short connection constraints (9.8), delay consistency constraints (9.9) and the passenger reallocation constraints (9.10) are given by $\boldsymbol{\rho} = \{\rho_{ij}, \forall (i,j) \in E^D\}$, $\boldsymbol{\gamma} = \{\gamma_j^v, \forall j \in N^D, \forall v \in U_j\}$ and $\boldsymbol{\lambda} = \{\lambda_j^v, \forall j \in N^D, \forall v \in U_j\}$ respectively. Finally, the dual variables $\boldsymbol{\delta}^K = \{\delta^k, \forall k \in K\}$ and $\boldsymbol{\delta}^R = \{\delta^r, \forall r \in R\}$ are defined for the crew and aircraft assignment constraints, (9.11)-(9.12) and (9.13), respectively. These sets of dual variables are used to define the reduced cost functions for the crew, aircraft and cancellation variables. The reduced cost functions are set as the objective functions for the respective column generation subproblems.

Crew and aircraft column generation subproblems

The column generation subproblems for the crew duty and aircraft routing variables can be described as;

$$\begin{aligned}\hat{c}_p^k = \min_{p \in P^k} \Bigg\{ & RecDutyCost(k) - \sum_{j \in N^D \cup N_{out}^K} \alpha_j a_{jp}^k - \sum_{(i_u, j_v) \in E^D} \rho_{i_u j_v} e_{i_u j_v p}^k \\ & - \sum_{j \in N^D} \sum_{v \in U_j} \left\{ \eta_j^v + \gamma_j^v + r_j \lambda_j^v \right\} a_{jp}^{kv} - \delta^k \Bigg\} \quad \forall k \in K,\end{aligned} \tag{9.17}$$

$$\begin{aligned}\hat{c}_p^r = \min_{p \in P^r} \Bigg\{ & \sum_{j \in N^D} \sum_{v \in U_j} c_{jp}^{rv} a_{jp}^{rv} - \sum_{j \in N^D \cup N_{out}^R} \alpha_j a_{jp}^r - \sum_{b \in B^R} \epsilon_b o_{bp}^r \\ & + \sum_{(i_u, j_v) \in E^D} \rho_{i_u j_v} e_{i_u j_v p}^r + \sum_{j \in N^D} \sum_{v \in U_j} \gamma_j^v a_{jp}^{rv} - \delta^r \Bigg\} \quad \forall r \in R.\end{aligned} \tag{9.18}$$

In problems (9.17) and (9.18), the minimisation problem is defined over the sets P^k and P^r, which contain all possible flight strings for crew k and aircraft r respectively. While this description indicates that additional variables can be identified from the sets $\bar{P}^k \subseteq P^k$ and $\bar{P}^r \subseteq P^r$ the reduced costs for all variables in these sets is at least zero. Therefore, all negative reduced cost variables identified by solving (9.17) and (9.18) will be contained in $P^k \backslash \bar{P}^k$ and $P^r \backslash \bar{P}^r$.

The feasible regions describing problems (9.17) and (9.18) are identical to that of the column generation subproblems for crew (PSP^k) and aircraft (PSP^r) respectively in Chapter 8. The only difference between (9.17) and the PSP^k is the additional dual variables related to the passenger reallocation constraints in the objective function. This modification of the objective function has no effect on the structure of the crew flight strings or solution methods that may be applied to solve (9.17). Therefore, the same definitions and solution algorithms used to solve the PSP^k and PSP^r can be applied to the problems (9.17) and (9.18) respectively.

Cancellation variable subproblem

Each cancellation variable defines a reallocation scheme providing alternative travel arrangements for passengers in the event of a flight cancellation. Ideally this modelling approach provides an alternative travel arrangement for all disrupted passengers, however this is not always possible. As such, passengers on cancelled flights are partitioned into two categories, stranded and reallocated passengers. To model the cost of a flight cancellation d_{ip}, a cost is assigned to each of these categories. Namely, the parameter g^{CAN} is defined as the cost of leaving a single passenger stranded and g_{ij}^{RA} as the cost of reallocating a single passenger to flight j following the cancellation of flight i. Both the costs given by g^{CAN} and g_{ij}^{RA} include a quantitative measure of passenger dissatisfaction and loss of good will. The definition of d_{ip} for the reallocation of passengers by scheme p following the cancellation of flight i is given by,

$$d_{ip} = g^{CAN} Pax(i) + \sum_{j \in N_i^{post}} \sum_{v \in U_j} h_{ijp}^{v} \left(g_{ij}^{RA} - g^{CAN} \right). \tag{9.19}$$

Equation (9.19) consists of a fixed cost which assumes all passengers are stranded and a variable cost that is dependent on the number of passengers rebooked onto alternative flights. The traditional approach used to model flight cancellations only considers the fixed cost of this equation. The variable cost in equation (9.19) is a contribution of this chapter, which is implemented in an attempt to reduce the operational cost of an airline.

Using the definitions of the dual variables given in Section 9.2.1, the reduced cost of a cancellation variable for flight i is given by,

$$\bar{d}_{ip} = g^{CAN} Pax(i) - \alpha_i^K + \sum_{j \in N_i^{post}} \sum_{v \in U_j} h_{ijp}^{v} \left(\lambda_j^v + g_{ij}^{RA} - g^{CAN} \right). \tag{9.20}$$

It is clear that the first two terms of equation (9.20) are not dependent on the reallocation of passengers to alternative flights. Therefore, the objective function for the cancellation variables column generation subproblem is defined by the final term of equation (9.20).

The column generation subproblem for the cancellation variables (CV-PSP) to identify passenger reallocation schemes for flight i is given by,

$$
\text{(CV-PSP)} \quad
\begin{aligned}
\tilde{d}_{ip} = \max \quad & \sum_{j \in N_i^{post}} \sum_{v \in U_j} h_{ijp}^v \left(g^{CAN} - \lambda_j^v - g_{ij}^{RA}\right), && (9.21)\\
\text{s.t.} \quad & \sum_{j \in N_i^{post}} \sum_{v \in U_j} h_{ijp}^v \leq Pax(i), && (9.22)\\
& h_{ijp}^v \leq r_j w_{ij}^v \quad \forall j \in N_i^{post}, \forall v \in U_j, && (9.23)\\
& \sum_{v \in U_j} w_{ij}^v = 1 \quad \forall j \in N_i^{post}, && (9.24)\\
& h_{ijp}^v \in \mathbb{Z}^+ \quad \forall j \in N_i^{post}, \forall v \in U_j, && (9.25)\\
& w_{ij}^v \in \{0, 1\} \quad \forall j \in N_i^{post}, \forall v \in U_j. && (9.26)
\end{aligned}
$$

The problem described by CV-PSP is a difficult integer programming problem that shares many similarities with a bounded knapsack problem. The main difference between the CV-PSP and a bounded knapsack problem arises from the multiple copies for flight j given by the set U_j. The multiple copies complicates the formulation of the CV-PSP, requiring that at most one copy v for each flight j is used as a reallocation option. This is enforced through the addition of the variables w_{ij}^v and constraints (9.23)-(9.24).

A more convenient formulation of the cancellation variable subproblem can be found by eliminating the need to consider multiple copies for each flight. This is achieved for the CV-PSP by identifying a single copy v' for each flight j that satisfies $\lambda_j^{v'} = \min_{v \in U_j}\{\lambda_j^v\}$. Replacing $\sum_{v \in U_j} \lambda_j^u$ with $\lambda_j^{v'}$ and setting $w_{ij}^{v'} = 1$ for all flights $j \in N_i^{post}$ forces the use of at most one copy for each flight. The resulting problem is called the reduced pricing subproblem (CV-PSPR), which is a bounded knapsack problem equivalent to the CV-PSP. The CV-PSPR is given by,

$$
\text{(CV-PSPR)} \quad
\begin{aligned}
\tilde{d}_{ip} = \max \quad & \sum_{j \in N_i^{post}} h_{ijp}^{v'} \left(g^{CAN} - \lambda_j^{v'} - g_{ij}^{RA}\right), && (9.27)\\
\text{s.t.} \quad & \sum_{j \in N_i^{post}} h_{ijp}^{v'} \leq Pax(i), && (9.28)\\
& h_{ijp}^{v'} \leq r_j \quad \forall j \in N_i^{post}, && (9.29)\\
& h_{ijp}^{v'} \in \mathbb{Z}^+ \quad \forall j \in N_i^{post}. && (9.30)
\end{aligned}
$$

The following theorem will prove that the optimal solution to the CV-PSPR is also optimal for the CV-PSP.

Theorem 9.2.1. *The optimal solutions to the CV-PSP and CV-PSPR are identical.*

Proof. Assume that an optimal solution to CV-PSP for a given flight i has been found describing passenger reallocation scheme p with an objective value $\tilde{d}'_{ip}$. The optimal solution is described by the variables $\bar{h}^{v'}_{ijp} > 0, \forall j \in N^{post}_i$, where v' is the copy for flight j where $w^{v'}_{ij} = 1$.

Let $\exists j_0 \in N^{post}_i$, such that $\lambda^{v'}_{j_0} > \min_{v \in U_{j_0}}\{\lambda^v_{j_0}\}$, and $\forall j \in N^{post}_i \backslash j_0$, $\lambda^{v'}_j = \min_{v \in U_j}\{\lambda^v_j\}$. Therefore, the objective value for the optimal solution to the CV-PSP is can be written as,

$$\tilde{d}'_{ip} = \sum_{j \in N^{post}_i \backslash j_0} \bar{h}^{v'}_{ijp}\left(g^{CAN} - \lambda^{v'}_j - g^{RA}_{ij}\right) + \bar{h}^{v'}_{ij_0p}\left(g^{CAN} - \lambda^{v'}_{j_0} - g^{RA}_{ij_0}\right). \tag{9.31}$$

Since $\bar{h}^{v'}_{ij_0p} > 0 \Rightarrow \sum_{v \in U_{j_0} \backslash v'} \bar{h}^{v'}_{ij_0p} = 0$. Let $\lambda^{v''}_{j_0} = \min_{v \in U_{j_0}}\{\lambda^v_{j_0}\}$ and since r_{j_0} does not depend on $v \in U_{j_0}$ another feasible solution to CV-PSP is given by setting $\hat{h}^v_{ijp} = \bar{h}^v_{ijp}, \forall j \in N^{post}_i, \forall v \in U_j$ with $\hat{h}^{v''}_{ij_0p} = \bar{h}^{v'}_{ij_0p}$ and $\hat{h}^{v'}_{ij_0p} = 0$.

Now $\lambda^{v''}_{j_0} < \lambda^{v'}_{j_0}$, so the objective value for this new feasible solution $\tilde{d}''_{ip}$ is greater than $\tilde{d}'_{ip}$. Hence, $\tilde{d}'_{ip}$ is not the objective value for the optimal solution to the CV-PSP, which is a contradiction to the original assumption.

Therefore, the optimal solution to the CV-PSP exists only when $\lambda^{v'}_j = \min_{v \in U_j}\{\lambda^v_j\}, \forall j \in N^{post}_i$, which is the objective value given by the solution to the CV-PSPR. So the optimal solutions to the CV-PSP and CV-PSPR are identical. □

The CV-PSPR describes a bounded knapsack problem, hence the linear relaxation is solved to optimality using a greedy heuristic. Additionally, the right hand side of constraints (9.28) and (9.29) are integer, so the solution to the greedy heuristic also provides the integer optimal solution. This feature of the CV-PSPR ensures very small runtimes, resulting in very little time spent generating columns for the cancellation variables.

Scaling of the passenger reallocation constraint coefficients. Computational experiments indicate that the passenger reallocation constraints (9.10) significantly increase complexity of the recovery problem, resulting in long solution runtimes. These constraints can be likened to big-M constraints, which are well known for negatively affecting the ability to identify integer solutions. A method identified to improve the computational performance of the IRP-PR is to scale the passenger reallocation constraints (9.10) by r_j. By applying this scaling, the set of constraints (9.10) can be restated as,

$$\sum_{k \in K}\sum_{p \in P^k} a^{kv}_{jp} x^k_p - \kappa^{v+}_j - \sum_{i \in N^j}\sum_{p \in P^i} \tilde{h}^v_{ijp} z_{ip} \geq 0 \quad \forall j \in N^D, \forall v \in U_j, \tag{9.32}$$

where $\tilde{h}^v_{ijp} = h^v_{ijp}/r_j$. Replacing constraints (9.10) with the scaled constraints (9.32) ensures that the coefficients of the crew duty variables are one in all rows of the IRP-PR. This replacement requires a modification to the reduced cost functions for the crew duty and cancellation variables. The modified reduced cost functions are given by,

$$\bar{c}^k_p = c^k_p - \sum_{j \in N^D \cup N^K_{out}} \alpha^K_j a^k_{jp} - \sum_{(i,j) \in E^D} \rho_{ij} e^k_{ijp} - \sum_{j \in N^D} \sum_{v \in U_j} \left\{ \eta^u_j + \gamma^u_j + \lambda^v_j \right\} a^{kv}_{jp} - \delta^k, \tag{9.33}$$

$$\bar{d}_{ip} = g^{CAN} Pax(i) - \alpha^K_i + \sum_{j \in N^{post}_i} \sum_{v \in U_j} \tilde{h}^v_{ijp} \left(\lambda^v_j + \left(g^{RA}_{ij} - g^{CAN} \right) r_j \right). \tag{9.34}$$

This constraint modification significantly affects the structure of the column generation subproblem for the cancellation variables. In particular, the scaled coefficients $\tilde{h}^v_{ijp}$ are no longer integer, therefore integrality is not required for the decision variables. The scaled reduced cancellation variable pricing subproblem (CV-PSPRS) is defined as,

(CV-PSPRS)

$$\tilde{d}_{ip} = \max \sum_{j \in N^{post}_i} \tilde{h}^{v'}_{ijp} \left(\left(g^{CAN} - g^{RA}_{ij} \right) r_j - \lambda^{v'}_j \right), \tag{9.35}$$

$$\text{s.t.} \sum_{j \in N^{post}_i} \tilde{h}^{v'}_{ijp} r_j \leq Pax(i), \tag{9.36}$$

$$\tilde{h}^{v'}_{ijp} \in [0,1] \quad \forall j \in N^{post}_i. \tag{9.37}$$

Since the CV-PSPRS is a bounded knapsack problems formulated as a linear program, the optimal solution is found using a greedy heuristic. Similar to the CV-PSPR, the right hand side of constraints (9.36) are all integer, hence only integer quantities of passengers are reallocated to each available flight. The computational experiments presented in Section 9.3 will discuss the improvements in the solution runtimes for the IRP-PR achieved by applying the described scaling of the coefficients for the cancellation variables.

9.2.2 Row generation

The inclusion of multiple sets of secondary variables in the general problem formulation is a contribution of the column-and-row generation framework presented in Chapter 7. Chapter 8 describes a problem that contains a set of primary variables and a single set of secondary variables. While the general framework presented in Chapter 7 can be applied to this type of problem, the implementation in Chapter 8 does not demonstrate the strength of the column-and-row generation approach. By contrast, the IRP-PR presents a problem with a set of primary variables, given by the crew duty variable, and two sets of secondary variables, the aircraft routing and cancellation variables. The IRP-PR fits within the more general case of the framework presented in Chapter 7, providing an example of the contribution to the column-and-row generation solution approach. The application of the column-and-row generation framework to the IRP-PR will be discussed in the following sections.

Formulation of the restricted problems

The short restricted master problem (SRMP) is defined as a further restriction of the RMP by eliminating a set of structural constraints. The SRMP is formulated for the IRP-PR to including only a subset of all possible rows from the delay consistency (9.9) and the passenger reallocation constraints (9.10) that form the RMP. Since the recovery policy of flight delays is modelled using flight copies, the set of rows selected to include in the SRMP is given by a subset of all copies $\bar{U}_j \subseteq U_j$ for each flight j. The initialisation of the sets $\bar{U}_j$ is identical to the method described in Section 8.2.2. As explained in Section 7.3.1, the elimination of rows to form the SRMP is coupled with variable fixings in the column generation subproblems. Therefore, all feasible solutions to the SRMP are feasible for the RMP and the IRP-PR.

The formulation of the SRMP is a fundamental feature of the column-and-row generation solution approach. The modelling approach for flight delays in the IRP-PR conveniently describes the eliminated rows by the sets $\bar{U}_j$. This notation is also used to describe the restriction on the column generation subproblems, where U_j is replaced by $\bar{U}_j$. Similar to Chapter 8, the variable fixings in the column generation subproblems are described by the elimination of connections from the connection networks. Namely the sets $\bar{C}^K \subseteq C^K$, $\bar{C}^R \subseteq C^R$ and $\bar{E}^D \subseteq E^D$ are defined to contain only the connections related to the flight copies in $\bar{U}_j, \forall j \in N^D$. This reduces the runtimes of the column generation subproblems, which is a source of the improvement in the solution runtimes for the complete problem.

Row generation algorithm

The general framework for the row generation procedure is detailed in Algorithm 7.2. This algorithm involves two stages, the calculation of the optimal dual solution and identifying favourable rows to include in the SRMP. Since the IRP-PR is formulated with two secondary variables, a modification to the dual calculation procedure presented for the IRP is required.

The calculation of an optimal dual solution is necessary to accurately evaluate the reduced costs of variables feasible for the RMP$'$. As stated by Theorem 7.3.1, the optimal dual solution to the RMP$'$ is calculated from the optimal solution of the SRMP using Algorithm 7.1. The rows contained in the RMP$'$ are partitioned into two groups defined by the sets $\bar{U}_j$ and $U_j \backslash \bar{U}_j$. Using the results from Section 7.3.2, the dual solutions for the rows related to $\bar{U}_j$ in the RMP$'$ are simply equated to the solutions of the respective rows in the SRMP.

The dual variables for the rows related to $U_j \backslash \bar{U}_j$ in the RMP$'$ are given by $\boldsymbol{\gamma}' = \{\gamma_j^{v'}, \forall j \in N^D, \forall v' \in U_j \backslash \bar{U}_j\}$ and $\boldsymbol{\lambda}' = \{\lambda_j^{v'}, \forall j \in N^D, \forall v' \in U_j \backslash \bar{U}_j\}$. Since the dual variables contained in $\boldsymbol{\gamma}'$ and $\boldsymbol{\lambda}'$ are related to the rows not included in the SRMP, the solutions must be calculated using the PSP^R, which is defined in Section 8.2.1, and CV-PSPRS respectively in step 2 of Algorithm 7.1. As stated in Section 8.2.2, the PSP^R is used as a problem specific enhancement to the row generation procedure. Unfortunately, no such general column generation subproblem can be formed for the cancellation variables, requiring the CV-PSPRS to be solved once for each $\lambda_j^{v'} \in \boldsymbol{\lambda}'$.

Favourable rows for the SRMP are identified by evaluating the reduced costs of primary variables feasible for the RMP$'$. Similar to the IRP, the primary variables for the IRP-PR are the crew duty variables, as such problem (9.17) is solved in step 8 of Algorithm 7.2. To identify crew duty variables feasible for the RMP$'$ in Algorithm 7.2, problem (9.17) is solved with the complete set of possible delay copies given by U_j. If any negative reduced cost variables are found, the flight-copy pairs contained on the related flight strings indicate the rows that must be included in the SRMP. In addition to adding rows to the SRMP, the related variable fixings in the column generation subproblems are relaxed, increasing the size of the connection network. If no negative reduced cost variables are found, the current solution for the SRMP is the optimal solution to the IRP-PR. The procedure to identify favourable rows for the SRMP is identical to the process described for the IRP in Section 8.2.2. Therefore, to identify a greater number of rows to add to the SRMP, problem (9.17) is solved once for each $k \in K$.

Section 8.2.2 describes a number of enhancement techniques that are employed in the solution process of the IRP. In particular, a method to increase the number of rows added to the SRMP and a row warm-up procedure are discussed. The results in Section 8.3 demonstrate improved runtime performance from the implementation of these techniques, as such they are also employed in the solution process of the IRP-PR. In the following section an additional enhancement technique is described that uses properties of the row generation framework to improve the convergence of the branch-and-price algorithm.

9.2.3 Variable fixing

In Section 8.2.1, the PSP^k and PSP^r are formulated as network flow problems describing variables contained in P^k and P^r respectively. The formulation of the PSP^k and PSP^r introduces arc-based variables $w^k_{i_u j_v p}$ for crew ($w^r_{i_u j_v p}$ for aircraft) that equal 1 to indicate that connection (i_u, j_v) is used on flight string p. The elimination of rows to form the SRMP is coupled with the fixing these arc-based variables $w^k_{i_u j_v p} = 0$ ($w^r_{i_u j_v p} = 0$) in the network flow problems, where $v \in U_j \backslash \bar{U}_j$. As explained above, this variable fixing can also be described as the elimination of connections from the networks used to solved the shortest path problems for crew and aircraft.

The row generation procedure described in Section 9.2.2 involves the addition of rows to the SRMP and the unfixing of variables in the column generation subproblems. This causes the size of connection networks used for each shortest path problem to increase, having a negative effect of solution runtimes. Ideally, the SRMP is formed to contain only the rows related to the flight delays in the optimal solution, which would result in the smallest possible connection networks for column generation subproblems. As explained in Section 8.2.2, identifying such a set of rows is analogous to solving the original problem, therefore only an approximation can be made.

A variable fixing heuristic is proposed to reduce the size of the column generation subproblems using the optimal solution to the SRMP at the root node of the branch-and-bound tree. This heuristic assumes that the flight-copy pairs used in the optimal solution of the LP relaxation for the SRMP have a high probability of occurring in the integer optimal solution. Given the high level of degeneracy in airline planning and recovery problems, it is also assumed that the flight-copy pairs included on variables that have a reduced cost of zero will potentially be used in the integer optimal solution. More formally, the set $I = \{j_v | \bar{c}^k_p = 0 \wedge a^{kv}_{jp} = 1, \forall k \in K, \forall p \in P^k\}$ defines the set of flight-copy pairs that are expected to be included on basic variable flights strings. This identifies a subset of flight-copy pairs that is used to eliminate rows from the SRMP and reduce the size of the column generation subproblems. From I, the set of connections included in the column generation subproblems is defined as $\bar{C} = \{(i_u, j_v) | i_u \in I \wedge j_v \in I\}$, hence the variable fixing, $w^k_{i_u j_v} = 0, w^r_{i_u j_v} = 0, \forall (i_u, j_v) \notin \bar{C}$ can be applied.

This heuristic uses information from the optimal solution to the LP relaxation of the IRP-PR as a proxy for the integer optimal solution. As such, it is not guaranteed that the solutions providing the best lower and upper bounds for the IRP-PR will contain only the flight-copy pairs included in I. Consequently, it is possible that the integer optimal solution that is found using this heuristic will overestimate the true optimal solution. To provide a meaningful comparison of the solution quality achieved with the solution approaches of column generation and column-and-row generation, the optimality gap in this chapter is calculated using a lower bound set at the LP solution found at the root node.

9.2.4 Branching rules

The integer optimal solution for the IRP-PR is found using the technique of branch-and-price. This involves solving the LP relaxation of the original problem to optimality and applying branching rules to partition the problem and enforce integrality. At each node in the branch-and-bound tree, column generation is used to solve the LP relaxation of the modified problems to optimality. The branching rules implemented must efficiently partition the feasible region of the problem without destroying the column generation subproblem structure. This is achieved by employing constraint branching techniques as described in Ryan and Foster [76].

Since the IRP-PR is an extension of the IRP, the branching rules that are described in Chapter 8 are also implemented here. However, the alternative modelling approach that is used for the flight cancellation variables in the IRP-PR requires some modification to the branching rules given in the previous chapter. In particular the cancellation variable branching is modified to include multiple variables for each flight coverage constraint. In addition, the application of the identifier/flight branching rule for the flight cancellation variables will also be described. Finally, an additional branching rule is implemented for the IRP-PR to eliminate any fractional solutions to the cancellation variables.

The first of the branching rules presented in Chapter 8 is a variable branching rule for the cancellation variables. In the IRP, the decision to cover or cancel a flight is determined by explicitly branching on the cancellation variables, forcing the variable to 1 and 0 on the left and right branches respectively. Since the IRP-PR is formulated with multiple cancellation variables for each flight, it is not possible to directly apply this variable branching. Flight j is identified as a branching candidate if that flight is partially covered by a variable flight string and partially cancelled. Branching is performed by forcing all aircraft and crew flights strings including flight j to zero on the right branch and all cancellation variables for flight j to zero on the left branch. The application of this branching rule does not completely eliminate the fractionality of all variables, as such additional branching rules are required.

The final rule in Chapter 8 describes an identifier/flight branching rule for branching on the aircraft and crew flight string variables. This rule is applied in the IRP-PR, also branching on the cancellation variables z_{ip}, which are identified by the cancelled flight i. The calculation of the fractionality for a cancellation variable identifier/flight pair is similar to equations (8.31) for the crew and aircraft variables in the IRP. The implementation of this branching rule is identical to that given in the previous chapter, and is easily extended to the cancellation variables.

The modelling approach for the cancellation variables introduces difficulties regarding the branch-and-price framework. To ensure integrality of the cancellation variables is achieved, an additional variable branching rule is implemented for the IRP-PR. The variable branching rule is defined to select the cancellation variable that is most fractional, i.e. $\max_{i \in N^D}\{\min_{p \in P_f^i}\{z_{ip}, 1 - z_{ip}\}\}$, which is then fixed to 0 on the left branch and 1 on the right branch. In the solution process of the IRP-PR this branching rule is assigned the lowest priority resulting in it being seldom called. However, in some instances it is required to find the integer optimal solution.

9.3 Computational Results

The results for the IRP-PR aim to demonstrate the benefits of considering passengers in the recovery process with the reduction in operational costs. The reallocation considerations within the IRP-PR attempts to provide passengers with alternative travel arrangements in situations where flight cancellations are required. This is a two-fold benefit for the airline with the recapturing of lost revenue and the improvement of passenger satisfaction. Since the IRP-PR is an extension upon the IRP, a comparison will be made between the two models in regards to runtime, cost and disruption statistics.

Section 9.2 discusses the implementation of column-and-row generation for the IRP-PR. This solution approach is employed to eliminate a large number of constraints from the RMP and improve the solution runtime and quality, as measured by the optimality gap. The results from experiments will provide a justification for employing column-and-row generation to solve the IRP-PR with a comparison to a standard column generation approach.

The measure of solution quality is based upon the optimality gap achieved at the termination of the maximum allowable runtime. This measure is important for the scenarios that fail to solve within this runtime, requiring the best found integer solution to be implemented. In Section 9.2.3, a variable fixing heuristic is introduced which potentially overestimates the lower bound of the IRP-PR. As stated, a meaningful comparison between the column generation and column-and-row generation solution approaches is provided by calculating the optimality gap as the relative difference between the LP solution at the root node and the best found integer solution. Any reference to the optimality gap hereafter is a reference to the gap calculated in this manner.

9.3.1 Description of data and disruption scenarios

The data and disruption scenarios that are used for this problem are identical to those used in the computational experiments of Chapter 8. The motivation for using the same data sets is to provide consistency in the analysis of the two different recovery models. Therefore only a brief description of the key features will be provided in this section.

The IRP and IRP-PR are evaluated using a single day flight schedule that contains 262 flights transporting 28,492 passengers which are serviced by 48 aircraft and 79 crew groups. The flight schedule services 20 airports, of which 12 are overnight bases for aircraft and 4 are crew bases. Of the 20 airports, the majority of flights originate or terminate at only 2. A set of 16 disruption scenarios are generated for the evaluation of the IRP and IRP-PR, each representing a closure at one of the major airports starting at 6am, 7am, 8am or 9am and lasting for a total duration of 3 or 5 hours. Airport closure scenarios are selected for the evaluation purposes due to the significant associated recovery costs. Throughout this section the scenarios will be referred to by an identifying number as detailed in Table 9.2.

Start Time	6am	7am	8am	9am
Airport 1	(0,8)	(2,10)	(4,12)	(6,14)
Airport 2	(1,9)	(3,11)	(5,13)	(7,15)

Table 9.2: Scenario numbers used in the presentation of results. The bracketed values indicate two different closure durations (3 hours, 5 hours).

A full set of recovery policies, including the generation of new aircraft routes and crew duties, flight delays and cancellations are implemented for the IRP and IRP-PR. The recovery actions are permissible directly after the disruption occurs until the end of a selected recovery window. For these experiments the recovery window is set at 6 hours, commencing after the affected airport is reopened. While the use of a recovery window approximates the complete recovery problem, this approach is consistent with the objective to return operations back to plan as quickly as possible. The number of flights that depart within the closure period and

Scenario	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$\lvert N^D \rvert$	150	151	149	150	147	145	150	149	182	183	185	186	184	182	184	183

Table 9.3: The number of disruptable flights for each scenario.

recovery window is documented in Table 9.3. It is important to note that while the number of included flights is reduced, no approximation is made to reduce the number of affected crew and aircraft.

Flight delays and cancellations are two fundamental actions that are available to airlines to recover from disruptive events. Flight delays have been modelled using the common flight copies technique, with 7 copies used for the IRP and IRP-PR to discretise a maximum delay of 180 minutes. Attempting to provide a true cost of delays, the estimate detailed in the EUROCONTROL report by Cook and Tanner [25] has been used in these experiments. In [25] the cost of delaying a full aircraft for a minute is estimated at \$100 AUD.

The IRP-PR introduces a direct consideration of passengers following a flight cancellation. As stated in the previous chapter, the assumption is made that as a result of a flight cancellation passengers are either i) rebooked onto an alternative flight operated by the airline, ii) rebooked onto a flight operated by a different airline, or iii) provided a refund, including some compensation, and must rebook their own flight. In Chapter 8, only case iii) was modelled assuming no loss of *good will* and that passengers are not deterred from making future bookings with the airline. For the IRP-PR case i) is modelled in addition to case iii), so the cost of cancelling a flight must be adjusted to account for the number of passengers rebooked onto alternative flights. If a passenger is rebooked onto another flight operated by the airline, the full revenue is retained minus any delay costs associated with waiting for a later departure time. It is common that the number of available flying seats is less than the number of passengers requiring reaccommodation, as such some passengers are not provided alternative arrangements resulting in the full loss of ticket revenue for the airline.

The IRP solves a recovery problem integrating aircraft and crew to fit within the commonly applied sequential recovery process. As part of this process, passenger recovery is undertaken after all preceding stages have been solved. To directly compare the impact of considering passenger reallocation in the integrated recovery problem, as in the IRP-PR, this reallocation process is undertaken *a posteriori* using the solution to the IRP. This is achieved by employing a greedy approach to rebook passengers onto alternative flights, which are defined as having the same O-D pair and departing within a maximum allowable delay window.

This model is implemented in C++ by calling SCIP 3.0.1 [3] to solve the integer program using CPLEX 12.4 as the linear programming solver.

9.3.2 Comparison between the IRP and IRP-PR

Analysing the solutions to the IRP and IRP-PR aims to identify the effect of modelling passenger reallocation on the recovery costs, the recovery actions taken and the solution runtime. The results are produced by solving the two recovery models against the set of 16 scenarios detailed in Table 9.2. The optimal solution to each scenario is identified when the relative difference between the primal and dual bounds is less than 5% for the IRP and IRP-PR.

Recovery costs

The motivation for considering the alternative passenger recovery model given by the IRP-PR is to reduce the recovery costs incurred while accommodating disrupted passengers. The relative difference in the recovery costs between the IRP and IRP-PR is given in Figure 9.1. This figure demonstrates that a significant reduction in the operating costs of an airline can be realised by employing this alternative modelling technique. On average, using the IRP-PR to solve the recovery problem reduces the operational costs of the airline by 15.32%.

The largest relative improvement in recovery costs is 32.54%, which is achieved in scenario 2. Comparing the solution for this scenario given by the two alternative recovery models, there is a marked difference in the number of flight cancellations in the recovery process. Specifically the IRP cancels 4 flights compared to 12 for the IRP-PR. Now, this increase in the number of cancelled flights achieves a smaller recovery cost for the solution to the IRP-PR since all but 7 passengers are provided alternative travel arrangements. By contrast, applying the greedy reallocation approach using the solution to the IRP is only able to provide alternative arrangements for 83 of the 373 passengers on cancelled flights. The number of passengers stranded following a flight cancellation is greater than the total number of seats on a single flight. This demonstrates the magnitude of the disruption experienced by passengers as a result of solving the IRP. The ability of the IRP-PR to provide alternative travel arrangements for the majority of passengers on cancelled flight is common across all scenarios. Therefore, the consideration of passenger reallocation allows the strategic cancellation of flights to reduce operational costs of the airline.

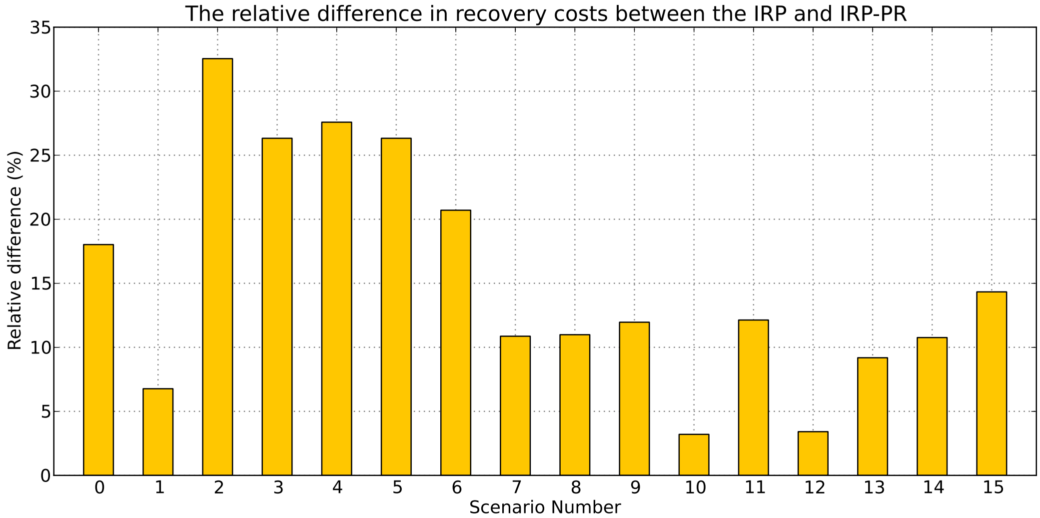

Figure 9.1: The relative difference between the recovery costs of the IRP (x) and IRP-PR (y). The value of the bars is given by $(x - y)/x$.

Cancellation and delay information

Considering passengers in the recovery process aims to reduce the impact of cancellations and delays on passenger travel arrangements. In particular, directly modelling reallocation options following flight cancellations ensures that passengers are routed to their destination with minimal delay. The reduction in recovery costs, presented in Section 9.3.2, benefits the airline, however this does not necessarily imply improved passenger satisfaction. In the analysis presented in this section it is assumed that providing passengers with alternative travel arrangements directly improves passenger satisfaction.

Figure 9.2 presents the total number of disrupted passengers and the effect of the disruption on their travel arrangements. The bars are divided into three different groups, i) the reallocated group are the passengers on cancelled flights that are rebooked onto alternative operating flights, ii) the delayed group includes all passengers that are booked on flights that depart later than scheduled, and iii) the cancelled group are the passengers for which no alternative travel arrangements are provided. For all passengers in the cancelled group, the tickets are refunded and the passengers must rebook themselves onto alternative flights. The passengers that are placed in this group are the most affected by the disruption, resulting in poor passenger satisfaction and the potential loss of future bookings.

The passenger reallocation approach developed for the IRP-PR increases the number of can-

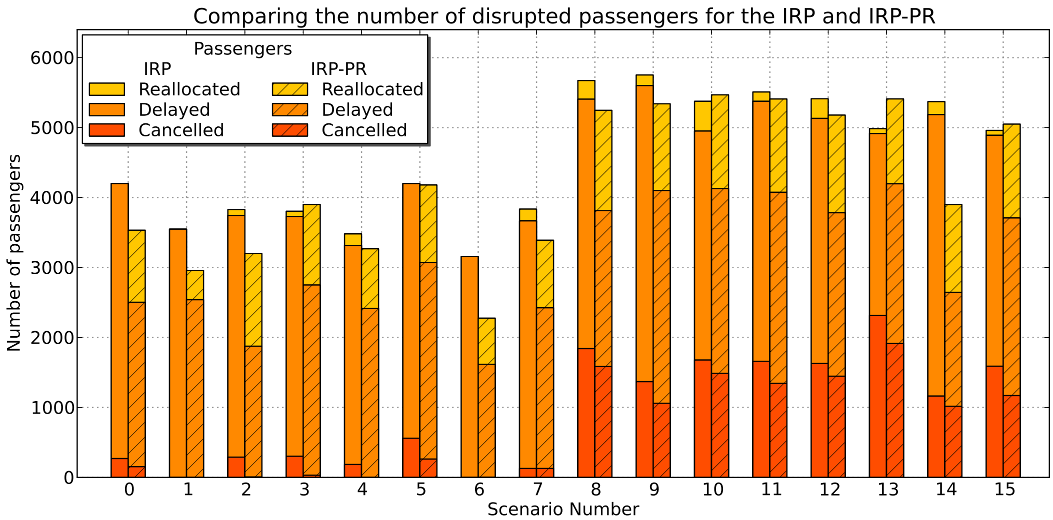

Figure 9.2: The number of reallocated, delayed and cancelled passengers from the IRP (bars) and IRP-PR (bars with hatching) solutions.

celled flights, however this is combined with an increase in the number of passengers reallocated onto alternative flights. Since reallocation options are directly modelled, flight delay decisions are made to ensure an adequate number of seats are available for the passengers on cancelled flights. It is clear in Figure 9.2 that by solving the IRP-PR more passengers are provided with alternative travel arrangements compared to the IRP, greatly reducing the number of cancelled passengers in each scenario. This result has a direct positive impact on passenger satisfaction by ensuring that a greater number of passenger arrive at their destination within a reasonable time frame.

The average on-time performance of the airline is very similar using the solutions to the IRP and the IRP-PR. The major difference between the two models is improved flow of passengers through the network achieved by solving the IRP-PR. On average, the solution to the IRP-PR disrupts 4232 passengers compared to 4568 for the IRP, providing a decrease of 336 passengers. Given that the solution to the IRP-PR provides a smaller recovery cost than the IRP, the decrease in the number of disrupted passengers demonstrates that this cost saving is not at the expense of passengers.

Another important passenger related metric is the average number of delay minutes experienced, which is presented in Figure 9.3. While a decrease in the average number of disrupted passenger is observed, this results in a marginal increase in the average delay minutes per passenger and disrupted passenger, 0.633 and 7.456 minutes respectively. There are two factors

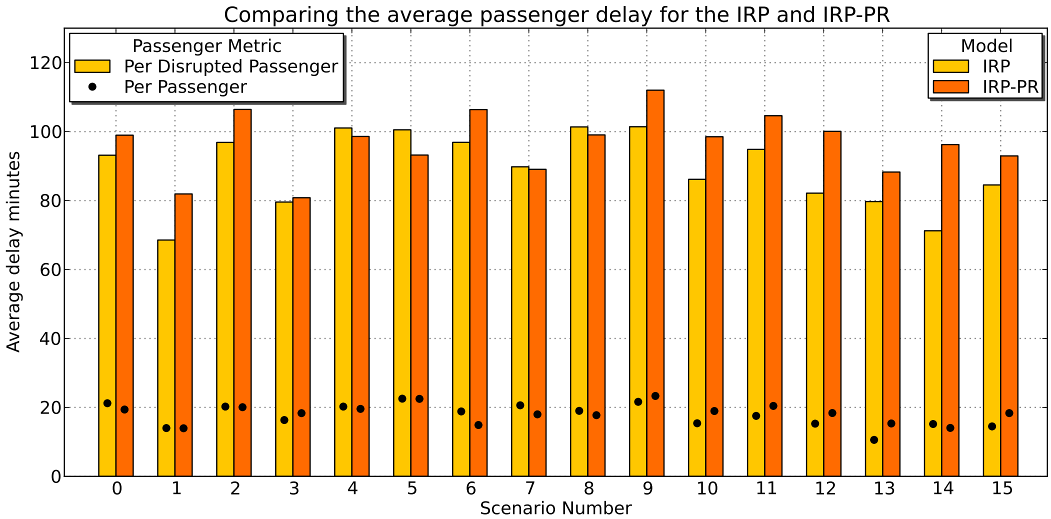

Figure 9.3: The average delay minutes per disrupted passenger (bars) and per passenger (dots) from the IRP (yellow) and IRP-PR (orange) solutions.

that affect the average number of delay minutes, i) the number of disrupted passengers and ii) the number of passengers delayed. The number of delay minutes per passenger is almost identical in the solution to the two models, this implies that the total number of passenger delay minutes remains unchanged. Since the solution to the IRP-PR disrupts less passengers, the number of delay minutes per disrupted passenger must increase. Also, the reallocated passengers experience a delay while waiting for the next available flight, where the cancelled passenger do not contribute to this statistic. By considering the decrease in the number of disrupted passengers and the small increase in the average delay minutes, it can be concluded that an overall positive outcome is achieved by providing alternative travel arrangements for passengers on cancelled flights.

The key feature of considering passenger reallocation options in the airline recovery problem is to provide alternative travel arrangements for disrupted passengers. This is demonstrated in Figure 9.2, with a large shift in the number of passengers from the cancelled group to the reallocated group, providing improved passenger flow through the network. Therefore it is not surprising that an increase in the average delay minutes per passenger and disrupted passenger is observed since flight delays are used as a strategy to ensure that alternative flights exist. The passenger reallocation of the IRP-PR is designed to reduce the effect of flight cancellations on passengers with the success measured by the increase of passenger flow through the flight network.

Solution runtimes

The additional variables and constraints related to passenger reallocation decisions in the IRP-PR increases the problem complexity compared to the IRP. As a result, it is expected that longer runtimes are required to identify the optimal integer solution for the IRP-PR. This is evident in Figure 9.4 where the IRP is solved to optimality faster than the IRP-PR for 10 of the presented scenarios. The main difference between the solution process for each model is that the IRP-PR generally requires a greater number of branches to converge to the integer optimal solution. Across all scenarios, the IRP-PR requires 47.44 nodes on average compared to 8.63 for the IRP. This is a significant difference in the solution process that has a great effect on the algorithm runtimes.

The slow convergence displayed by the IRP-PR can be explained further by analysing the optimality gap of the two models at the root node, presented in Table 9.4. This table shows that 3 of the scenarios are solved to optimality by the LP solution at the root node for the IRP compared to just 1 for the IRP-PR. In addition, the optimality gap for the IRP-PR is significantly larger than that for the IRP in all but one scenario, indicating the potential requirement of a greater number of branches in the branch-and-price algorithm. This is to be expected since the passenger reallocation constraints (9.10) are in the form of big-M constraints, which commonly display difficulties in identifying integer solutions.

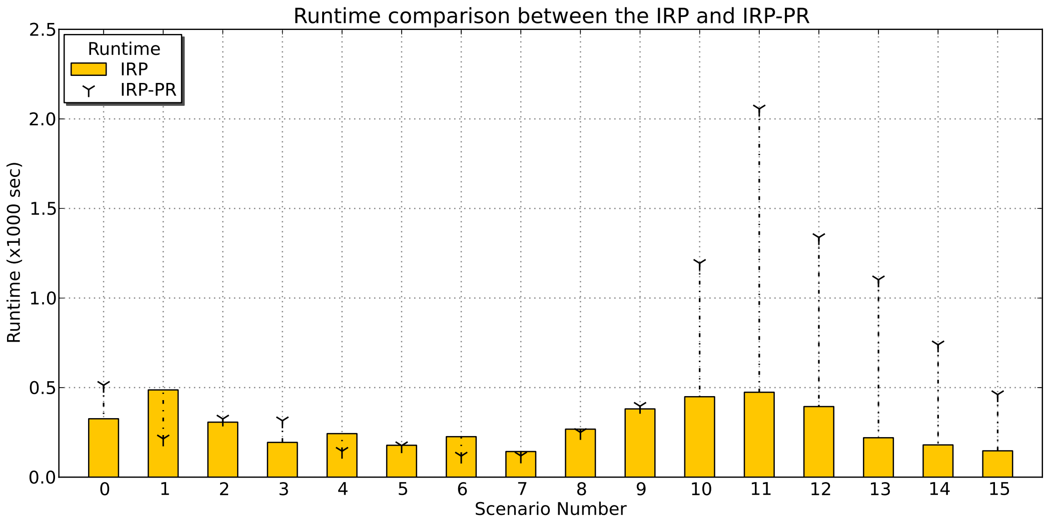

Figure 9.4: The runtimes to solve the IRP (bars) and IRP-PR (stars) using column-and-row generation for each scenario with a maximum of 2700 seconds.

Scenario	0	1	2	3	4	5	6	7
IRP	2888.35%	4630.87%	2312.73%	3303.54%	2605.06%	2093.85%	3528.54%	2729.15%
IRP-PR	3938.63%	6882.72%	4526.38%	5457.22%	0.0%	3155.8%	5648.75%	5100.72%
Scenario	8	9	10	11	12	13	14	15
IRP	900.55%	1425.56%	1162.0%	1700.12%	1114.59%	0.0%	0.01%	0.0%
IRP-PR	1415.53%	2034.81%	1705.44%	2240.13%	2499.49%	2029.96%	5843.9%	4862.17%

Table 9.4: The optimality gap at the root node from solving the IRP and the IRP-PR.

The runtime comparison between the IRP and the IRP-PR demonstrate that the increased complexity introduced by the passenger reallocation decisions increases the solution runtime. While this increase in solution runtimes is not ideal for an airline recovery problem, the difference is not prohibitively large. Figure 9.4 demonstrates that the vast majority of scenarios for the IRP-PR are solved within 600 seconds (10 minutes), which is within an acceptable runtime for practical use of the algorithm. Comparing Figures 9.1 and 9.4, it can be concluded that the potential gains from included passenger reallocation outweigh the resulting increase in runtime.

9.3.3 Comparison of solution methodology

The structure of the IRP-PR is very similar to the IRP, as such the solution approach of column-and-row generation has been implemented to provide fast runtimes with a high solution quality. The implementation of column-and-row generation for the IRP is evaluated in the previous chapter, demonstrating a significant improvement in solution runtime and quality over a standard column generation approach. In this section, further evaluation of column-and-row generation is performed in relation to the IRP-PR. This evaluation demonstrates the significant benefits from applying column-and-row generation and justifies its use to solve the IRP-PR.

The runtime required to solve the scenarios using the IRP-PR with column generation and column-and-row generation is presented in Figure 9.5. This figure demonstrates a reduction in solution runtimes for the IRP-PR when solved with column-and-row generation. This is an important result since it indicates that a higher quality solution is achievable using column-and-row generation compared to column generation with shorter runtimes. The runtimes presented in Figure 9.5 show that column-and-row generation outperforms column generation in 10 of the presented experiments. This provides a relative difference in solution runtimes of 42.83%, significantly improving upon the results presented in Chapter 8. It is clear through the explicit

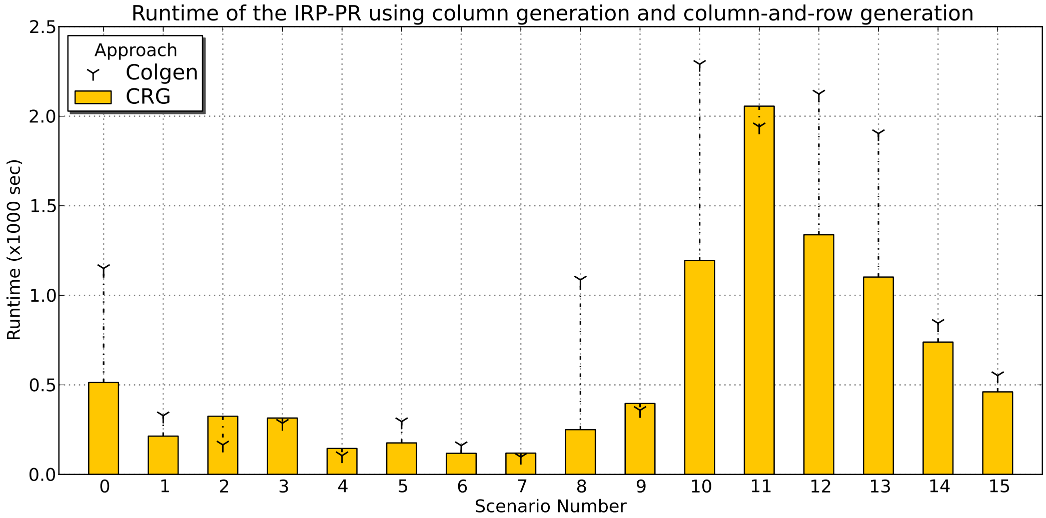

Figure 9.5: The runtimes to solve the IRP-PR for each scenario with a maximum of 2700 seconds. This figure compares the solution approaches of column-and-row generation (bars) and column generation (stars).

evaluation of the solution runtimes, column-and-row generation significantly outperforms a standard column generation approach.

9.3.4 Runtime enhancement techniques

A number of enhancements techniques designed to improve the solution runtime and quality of column-and-row generation are proposed in Section 9.2. Many of these enhancements, such as problem specific branching rules, are commonly applied to mixed integer programming problems. In this chapter a variable fixing heuristic and the reformulation of the cancellation variables have been developed specifically for the IRP-PR.

The variable fixing heuristic restricts the allowable delay copies used to construct columns, therefore reducing the feasible region of the IRP-PR. This technique attempts to reduce the runtimes of the subproblem and the LP solves in each iteration of the column generation algorithm. The modelling of the cancellation variables in this chapter is a unique approach for considering passenger reallocation for the integrated recovery problem. Section 9.2.1 details the scaling of the coefficients for the crew duty and cancellation variables in the reallocation constraints (9.32). Since the reallocation constraints (9.10) are in the form of big-M constraints, this scaling is proposed to alleviate the common issues related to identifying integer solutions.

Figure 9.6 details the time required to solve the scenarios with the IRP-PR, implementing

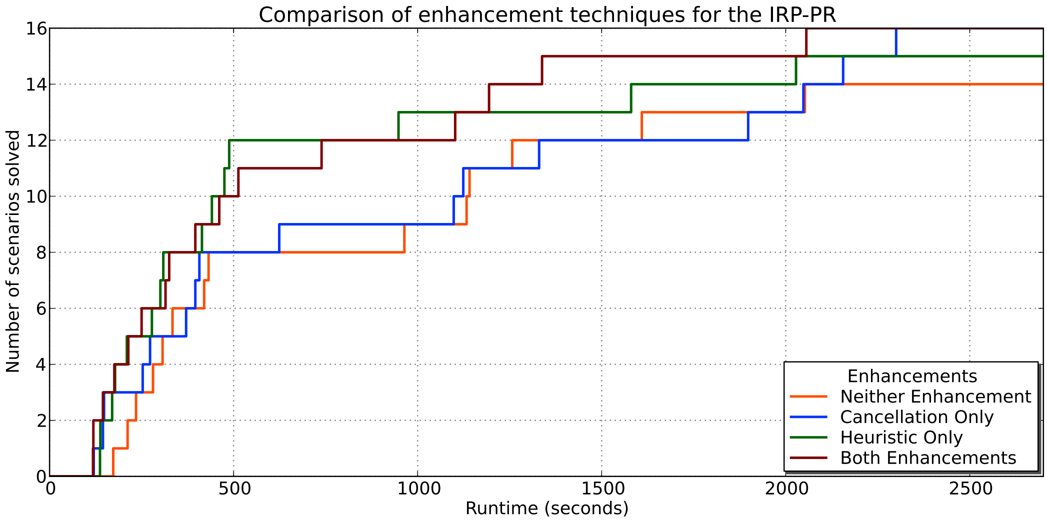

Figure 9.6: Time required to solve scenarios by employing different enhancement techniques, specifically the variable fixing heuristic and/or the cancellation variable scaling enhancements. A maximum runtime of 2700 seconds (45 minutes) is applied.

the variable fixing heuristic and/or the cancellation variable scaling enhancement techniques. It is interesting to note that all scenarios are solved to optimality within the maximum runtime of 2700 seconds (45 minutes) only when the cancellation variable scaling or both enhancements are used. As expected, the best overall runtime result is achieved when both of the enhancement techniques are implemented.

The enhancement techniques attempt to form a SRMP that is more easily solvable by branch-and-price. Comparing the frontiers created by each of the enhancement techniques, the greatest improvement is observed for scenarios requiring greater than 500 seconds runtime. Now, the average time to solve the root node when no enhancements are used is 208 seconds, with a maximum of 373 seconds. This suggests that the enhancements aid in improving the convergence to the integral optimal solution for the larger scenarios.

9.4 Conclusions

This chapter presents a novel approach for considering passengers in an integrated airline recovery problem through the modelling of cancellation variables. The integrated airline recovery problem integrates the schedule, aircraft and crew recovery problems, attempting to service the maximum number of flights following a disruption. The cancellation variables have been modelled as knapsack variables to describe the possible reallocation options for passengers in the event of a flight cancellation. This modelling approach contributes to airline recovery problems by providing a simple and effective method to consider passengers in recovery problems.

The passenger reallocation approach developed for the IRP-PR is demonstrated through experiments to greatly reduce the recovery costs achieved by the IRP. The reduction in recovery costs is directly related to the recapturing of lost revenue following flight cancellations by providing passengers with alternative travel arrangements. This demonstrates the significant cost benefits that can be achieved by considering passengers during the recovery process.

The solution to the IRP-PR achieves a greater passenger flow through the network as a result of strategic delay decisions. This is a feature of the modelling approach which affects the composition of flight delays and cancellations in optimal solution. While the solution to the IRP-PR results in a larger number of flight cancellations than the IRP, on average less passengers are disrupted. The delay decisions made in the solution to the IRP-PR provides a greater number of reallocation options for passengers on cancelled flights, reducing the number of stranded passengers. This has a significant impact on passenger satisfaction whereby more disrupted passengers arrive at their destination with a minimal amount of delay. The benefits achieved by solving the IRP-PR are realised by the airline and passengers through a reduction in costs and the magnitude of disruption.

Extending the analysis presented in Chapter 8, a comparison of the column generation and column-and-row generation solution approaches is performed for the IRP-PR. This comparison further emphasises the runtime improvements achieved by the column-and-row generation solution approach. In addition, the formulation of the IRP-PR displays two sets of secondary variables, providing an example of the contribution of the general framework presented in Chapter 7. To the best of the author's knowledge, the column-and-row generation framework requiring the calculation of two sets of dual variables has not been previously considered.

A number of enhancements have been proposed to reduce the solution runtime of the IRP-PR by improving the convergence to the integer optimal solution. These enhancements include a variable fixing heuristic and scaling the cancellation variable coefficients, which are developed specifically for the IRP-PR. The results demonstrate that the implementation of both enhancements provides the greatest improvement to the solution process by solving all scenarios to within the desired optimality gap. The variable fixing heuristic is developed using characteristics of the column-and-row generation solution approach, as such it is possible to implement such a heuristic for similar problems where column-and-row generation is applied.

The potential benefits from directly considering passengers during the recovery process are presented in this chapter. Passenger reallocation approach in the IRP-PR is a simple and novel approach that is designed to reduce passenger dissatisfaction resulting from flight cancellations. While this modelling approach reduces the number of cancelled and disrupted passengers, it does not have a significant effect on the average delay experienced per passenger. Future work on this problem involves integrating the passenger reallocation approach into a broader passenger recovery scheme in an attempt to reduce the impact of flight delays. There is an expectation that further reductions in recovery costs and improvements in passenger satisfaction can be achieved with the greater consideration of passenger flows in airline recovery problems.

Chapter 10

Conclusions

The concept of recoverable robustness is introduced by Liebchen *et al.* [57] as a framework to identify planning solutions that are more recoverable in the event of a disruption. This framework is developed to improve the recoverability of railway transportation problems. While there are similarities between railway and airline planning problems, there are significant differences affecting the application of the recoverable robustness framework. This thesis addresses these differences and presents the first application of recoverable robustness to airline planning problems. The key features of this thesis are i) investigating the application of recoverable robustness to various planning problems and ii) identifying and overcoming the challenges of implementing this technique. The planning problems considered are the tail assignment, aircraft maintenance routing and the integrated aircraft routing and crew duty problems. Each problem presents a variety of modelling challenges requiring the investigation of enhancement techniques to improve solution runtimes. In particular, a number of enhancements for the Benders' decomposition solution process are presented and an alternative general framework for column-and-row generation is developed. The various planning problems considered in this thesis and the investigation of relevant solution techniques demonstrates the potential and limitations of the recoverable robustness framework.

The recoverable robust tail assignment problem is presented in Chapter 5 as the first ever airline application of recoverable robustness. The application of recoverable robustness to the tail assignment problem is a proof-of-concept to demonstrate the potential improvements in recoverability for airline planning problems. A key contribution of Chapter 5 is the inclusion of a full set of recovery policies in the evaluation stage of the recoverable robustness framework. Formulating the recovery problem with a full set of recovery policies generates high quality feedback in the evaluation stage to achieve the greatest improvement in recoverability. An efficient solution approach is developed in Chapter 5 with the integration of Benders' decomposition and column generation. The enhancement techniques of the Magnanti-Wong method [60] and the two-phase method for the Benders' decomposition solution process are investigated. It is concluded that with the use of various enhancement techniques, recoverable robustness is an efficient and effective method to improve the recoverability of the tail assignment problem.

The aircraft maintenance routing problem is considered in Chapter 6 as part of a further investigation of the recoverable robustness framework. A contribution of this chapter is a novel modelling approach that integrates the aircraft routing and maintenance planning problems using a *one-day routes* formulation. The experiments on this novel maintenance planning approach demonstrate significant reductions in the number of maintenance misalignments for flight schedules of various sizes. The application of recoverable robustness in Chapter 6 attempts to improve the recoverability of the aircraft maintenance routing problem. A contribution to the recoverable robustness framework in this chapter is the ability to improve recoverability while ensuring the maintenance schedule is satisfied in the event of a disruption. Extending upon the results of Chapter 5 and as a contribution of Chapter 6, the recoverable robust aircraft maintenance routing problem is solved with a large number of evaluation scenarios and large flight schedules. The results demonstrate significant improvements in the recoverability of the maintenance planning problem from applying recoverable robustness. Hence, the results from Chapters 5 and 6 show that the recoverable robustness framework is an effective method to improve the recoverability of various airline planning problems. However, the solution quality and runtimes presented in Chapter 6 indicate that further development of solution techniques is required to address the limitations of the recoverable robustness framework.

Chapter 2 introduces the integration of multiple stages from the sequential approach as a potential method to improve the solution quality of the complete planning problem. The integration of multiple stages attempts to address the common result of suboptimal, and even infeasible, solutions from the sequential planning process. As an extension on Chapter 5, the integration of the aircraft and crew planning problems in the recoverable robustness framework is investigated in this thesis to improve both the solution quality and recoverability. The results from Chapter 6 indicate that alternative solution techniques are required to reduce the runtimes of the evaluation stage of the recoverable robustness framework. Therefore, column-and-row generation is investigated to reduce the complexity and improve the solution runtimes of the integrated airline recovery problems presented in Chapters 8 and 9.

Column-and-row generation is described in Chapter 7 as the simultaneous generation of variables and structural constraints. This solution approach is developed as an extension upon column generation to improve solution runtimes through a smaller and more restricted master problem. Chapter 7 details the various applications of this solution approach and introduces the current generic methods that have been proposed. It is not possible to directly apply the generic column-and-row generation schemes to the integrated airline recovery problems developed in this thesis, as such an alternative framework is developed. A contribution of the framework presented in Chapter 7 is the application of column-and-row generation to problems with multiple sets of secondary variables. This framework is presented as a direct alternative to Benders' decomposition. In addition, the framework developed in Chapter 7 is presented algorithmically, clearly describing its implementation. The integrated airline recovery problems developed in Chapters 8 and 9 are given as examples to demonstrate the implementation of the column-and-row generation framework.

The integrated airline recovery problem is investigated in Chapter 8, integrating the schedule, aircraft and crew recovery problems. The general framework for column-and-row generation developed in Chapter 7 is applied to the integrated airline recovery problem to improve the solution runtime and quality. A contribution of this chapter is the explicit evaluation of column-and-row generation against a standard column generation approach. Since the framework presented in Chapter 7 is developed from the generic scheme presented by Muter *et al.* [65], the evaluation in this chapter also demonstrates the performance of the scheme by [65]. The evaluation of column-and-row generation in Chapter 8 identifies a number of enhancement techniques that are applicable to the general implementation of the solution approach. A contribution of solving the integrated airline recovery problem by column-and-row generation is the guarantee of near optimal solutions, which is not provided by alternative solution methods such as Benders' decomposition. The results in Chapter 8 demonstrate that column-and-row generation is very effective in reducing the solution runtimes to achieve near optimal solutions for the integrated airline recovery problem. While a reduction in the solution runtimes is achieved with column-and-row generation, the magnitude of the runtimes preclude the use of this recovery problem in the evaluation stage of the recoverable robustness framework. The results from Chapter 8 reinforce the conclusion that efficient solution methods must be investigated to apply recoverable robustness to more complex planning stage problems.

Passenger recovery is an important stage of the recovery problem that is commonly omitted from automated solution approaches. A novel modelling approach for passenger recovery is investigated in Chapter 9 as a simple, but effective, method of considering passengers in any airline recovery problem. The potential of this modelling approach to improve passenger flows and reduce recovery costs is demonstrated by solving an integrated airline recovery problem including passenger considerations in Chapter 9. A contribution of Chapter 9 is the modelling of the cancellation variables to prescribe passenger reallocation options in the event of a flight cancellation. This modelling approach is directly applicable to point-to-point carriers where itineraries commonly contain only a single flight. This chapter demonstrates that higher quality feedback from the evaluation stage of the recoverable robustness framework can be achieved with this simple, novel passenger recovery approach. Another contribution of Chapter 9 is the further evaluation of the column-and-row generation solution approach on a more complex and difficult optimisation problem. The results demonstrate that the column-and-row generation framework presented in Chapter 7 effectively reduces the solution runtimes of complex integer programs with multiple linking constraints.

The recoverable robustness technique is developed throughout this thesis with a focus on airline applications. The various applications discussed identify a number of potential areas for further research. In particular, the computational experiments performed employ flight schedules collected from small to medium sized airlines. To improve the practical applicability of this approach, further investigation into solution methods and acceleration techniques is required to address the needs of large airlines. Additionally, the application of the column-and-row generation solution approach of Chapter 7 in this thesis identifies the need for acceleration techniques. Identifying the most efficacious of these techniques and developing an enhanced generic column-and-row generation solution approach is a possible avenue of further research.

This thesis demonstrates the potential improvements in recoverability achieved by applying recoverable robustness to airline planning problems. The results from each chapter shows that the application of the recoverable robustness framework requires sophisticated techniques, such as Benders' decomposition, column generation and column-and-row generation, to develop efficient solution approaches. The experiments applying recoverable robustness to various planning problems and data sets demonstrate the versatility and suitability of this approach. Through further research of this technique and with improvements to the relevant solution approaches, the full potential of recoverable robustness for airline applications will be realised.

Bibliography

[1] A. Abdelghany, G. Ekollu, R. Narasimhan, and K. Abdelghany. A proactive crew recovery decision support tool for commercial airlines during irregular operations. *Annals of Operations Research*, 127(23):309–331, 2004.

[2] K. F. Abdelghany, A. F. Abdelghany, and G. Ekollu. An integrated decision support tool for airlines schedule recovery during irregular operations. *European Journal of Operational Research*, 185(2):825–848, 2008.

[3] T. Achterberg. SCIP: solving constraint integer programs. *Mathematical Programming Computation*, 1(1):1–41, 2009.

[4] Y. Ageeva. Approaches to incorporating robustness into airline scheduling. Master's thesis, Massachusetts Institute of Technology, August 2000.

[5] R. Ahuja, T. Magnanti, and J. Orlin. *Network flows: theory, algorithms, and applications.* Prentice Hall, 1993.

[6] T. Andersson. Solving the flight perturbation problem with meta heuristics. *Journal of Heuristics*, 12(1-2):37–53, 2006.

[7] T. Andersson and P. Varbrand. The flight perturbation problem. *Transportation Planning and Technology*, 27(2):91–117, 2004.

[8] M. Argüello. *Framework for exact solutions and heuristics for approximate solutions to airlines' irregular operations control aircraft routing problem.* PhD thesis, University of Texas at Austin, 1997.

[9] P. Avella, B. D'Auria, and S. Salerno. A LP-based heuristic for a time-constrained routing problem. *European Journal of Operational Research*, 173(1):120–124, 2006.

[10] P. Avella, A. Sassano, and I. Vasil'ev. Computational study of large-scale p-median problems. *Mathematical Programming*, 109(1):89–114, 2007.

[11] J. Bard, G. Yu, and M. Argüello. Optimizing aircraft routings in response to groundings and delays. *IIE Transactions*, 33(10):931–947, 2001.

[12] C. Barnhart, N. L. Boland, L. W. Clarke, E. L. Johnson, G. L. Nemhauser, and R. G. Shenoi. Flight string models for aircraft fleeting and routing. *Transportation Science*, 32(3):208–220, 1998.

[13] C. Barnhart, A. Cohn, E. Johnson, D. Klabjan, G. Nemhauser, and P. Vance. Airline crew scheduling. In R. W. Hall, editor, *Handbook of Transportation Science*, volume 56 of *International Series in Operations Research and Management Science*, pages 517–560. Springer US, 2003.

[14] C. Barnhart, C. Hane, and P. Vance. Using branch-and-price-and-cut to solve origin-destination integer multicommodity flow problems. *Operations Research*, 48(2):318–326, 2000.

[15] C. Barnhart, E. Johnson, G. Nemhauser, M. Savelsbergh, and P. Vance. Branch-and-price: column generation for solving huge integer programs. *Operations Research*, 46(3):316–329, 1998.

[16] J. F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4(1):238–252, 1962.

[17] S. Bisaillon, J.-F. Cordeau, G. Laporte, and F. Pasin. A large neighbourhood search heuristic for the aircraft and passenger recovery problem. *4OR*, 9(2):139–157, 2011.

[18] R. Borndörfer, I. Dovica, I. Nowak, and T. Schickinger. Robust tail assignment. In *Proceedings of the fiftieth annual symposium of AGIFORS*, 2010.

[19] S. Bratu and C. Barnhart. Flight operations recovery: new approaches considering passenger recovery. *Journal of Scheduling*, 9(3):279–298, 2006.

[20] V. Cacchiani, A. Caprara, L. Galli, L. Kroon, G. Maróti, and P. Toth. Recoverable robustness for railway rolling stock planning. Technical report, ARRIVAL-Project, 2009.

[21] J. Cao and A. Kanafani. Real-time decision support for integration of airline flight cancellations and delays part I: mathematical formulation. *Transportation Planning and Technology*, 20(3):183–199, 1997.

[22] J. Cao and A. Kanafani. Real-time decision support for integration of airline flight cancellations and delays part II: algorithm and computational experiments. *Transportation Planning and Technology*, 20(3):201–217, 1997.

[23] L. Clarke, E. Johnson, G. Nemhauser, and Z. Zhu. The aircraft rotation problem. *Annals of Operations Research*, 69:33–46, 1997.

[24] J. Clausen, A. Larsen, J. Larsen, and N. J. Rezanova. Disruption management in the airline industry–concepts, models and methods. *Computers & Operations Research*, 37(5):809–821, 2010.

[25] A. Cook and G. Tanner. European airline delay cost reference values. *URL: http://www.eurocontrol.int/documents/european-airline-delay-cost-reference-values*, 2011.

[26] A. Cook, G. Tanner, and S. Anderson. Evaluating the true cost to airlines of one minute of airborne or ground delay, EUROCONTROL. *URL: http://www.eurocontrol.int*, 2004.

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

[28] G. Côté and M. A. Laughton. Large-scale mixed integer programming: Benders'-type heuristics. *European Journal of Operational Research*, 16(3):327–333, 1984.

[29] G. Desaulniers, J. Desrosiers, and M. M. Solomon, editors. *Column generation*, volume 5 of *GERAD 25th Anniversary Series*. Springer, 2005.

[30] M. Desrochers and F. Soumis. A generalized permanent labelling algorithm for the shortest path problem with time windows. *INFOR*, 26(3):191–212, 1988.

[31] J. Desrosiers, J. B. Gauthier, and M. E. Lübbecke. Row-reduced column generation for degenerate master problems. In *Optimization Online*, 2013.

[32] I. Dumitrescu and N. Boland. Improved preprocessing, labeling and scaling algorithms for the weight-constrained shortest path problem. *Networks*, 42(3):135–153, 2003.

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

[34] N. Eggenberg. *Combining Robustness and Recovery for Airline Schedules.* PhD thesis, École Polytechnique Fédérale De Lausanne, December 2009.

[35] N. Eggenberg, M. Bierlaire, and M. Salani. A column generation algorithm for disrupted airline schedules. Technical report, Ecole Polytechnique Federale de Lausanne, 2007.

[36] D. Feillet, M. Gendreau, A. L. Medaglia, and J. L. Walteros. A note on branch-and-cut-and-price. *Operations Research Letters*, 38(5):346–353, 2010.

[37] T. Feo and J. Bard. Flight scheduling and maintenance base planning. *Management Science*, 35(12):1415–1432, 1989.

[38] M. Fischetti, D. Salvagnin, and A. Zanette. A note on the selection of Benders' cuts. *Mathematical Programming*, 124(1-2):175–182, 2010.

[39] A. Frangioni and B. Gendron. A stabilized structured Dantzig-Wolfe decomposition method. *Mathematical Programming*, 140(1):45–76, 2013.

[40] G. Froyland, S. J. Maher, and C.-L. Wu. The recoverable robust tail assignment problem. *Transportation Science*. To appear.

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

[42] A. M. Geoffrion and G. W. Graves. Multicommodity distribution system design by Benders' decomposition. *Management Science*, 20(5):822–844, 1974.

[43] P. C. Gilmore and R. E. Gomory. A linear programming approach to the cutting-stock problem. *Operations Research*, 9(6):849–859, 1961.

[44] R. Gopalan and K. T. Talluri. The aircraft maintenance routing problem. *Operations Research*, 46(2):260–271, 1998.

[45] M. Grönkvist. *The Tail Assignment Problem.* PhD thesis, Chalmers University of Technology and Göteborg University, August 2005.

[46] M. Grönkvist. Accelerating column generation for aircraft scheduling using constraint propagation. *Computers & Operations Research*, 33(10):2918–2934, 2006.

[47] A. Heinhold. Maintenance reachability at Southwest Airlines. In *Proceedings of the AGIFORS Scheduling and Strategic Planning Study Group Meeting, Dallas*, 2008.

[48] N. Jafari and S. H. Zegordi. Simultaneous recovery model for aircraft and passengers. *Journal of the Franklin Institute*, 348(7):1638–1655, 2011.

[49] A. I. Z. Jarrah, G. Yu, N. Krishnamurthy, and A. Rakshit. A decision support framework for airline flight cancellations and delays. *Transportation Science*, 27(3):266–280, 1993.

[50] E. Johnson, L. Lettovsky, G. Nemhauser, R. Pandit, and S. Querido. Final report to Northwest Airlines on the crew recovery problem. Technical report, Georgia Institute of Technology, 1994.

[51] L. S. Kang. *Degradable airline scheduling: an approach to improve operational robustness and differentiate service quality.* PhD thesis, Massachusetts Institute of Technology, February 2004.

[52] E. Lacasse-Guay, G. Desaulniers, and F. Soumis. Aircraft routing under different business processes. *Journal of Air Transport Management*, 16(5):258–263, 2010.

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

[54] M. Lapp and A. Cohn. Modifying lines-of-flight in the planning process for improved maintenance robustness. *Computers & Operations Research*, 39(9):2051–2062, 2012.

[55] L. Lettovsky. *Airline operations recovery: an optimization approach.* PhD thesis, Georgia Institute of Technology, 1997.

[56] L. Lettovsky, E. Johnson, and G. Nemhauser. Airline crew recovery. *Transportation Science*, 34(4):337–348, 2000.

[57] C. Liebchen, M. Lübbecke, R. Möhring, and S. Stiller. The concept of recoverable robustness, linear programming recovery, and railway applications. In R. Ahuja, R. Möhring, and C. Zaroliagis, editors, *Robust and Online Large-Scale Optimization*, volume 5868 of *Lecture Notes in Computer Science*, pages 1–27. Springer Berlin Heidelberg, 2009.

[58] J. Linderoth and S. Wright. Decomposition algorithms for stochastic programming on a computational grid. *Computational Optimization and Applications*, 24(2-3):207–250, 2003.

[59] M. E. Lübbecke and J. Desrosiers. Selected topics in column generation. *Operations Research*, 53(6):1007–1023, 2005.

[60] T. Magnanti and R. Wong. Accelerating Benders' decomposition: algorithmic enhancement and model selection criteria. *Operations Research*, 29(3):464–484, 1981.

[61] S. J. Maher. Solving the integrated recovery problem using column-and-row generation. In preparation, 2012.

[62] C. P. Medard and N. Sawhney. Airline crew scheduling from planning to operations. *European Journal of Operational Research*, 183(3):1013–1027, 2007.

[63] A. Mercier, J. Cordeau, and F. Soumis. A computational study of Benders' decomposition for the integrated aircraft routing and crew scheduling problem. *Computers & Operations Research*, 32(6):1451–1476, 2005.

[64] A. Mercier and F. Soumis. An integrated aircraft routing, crew scheduling and flight retiming model. *Computers & Operations Research*, 34(8):2251–2265, 2007.

[65] İ. Muter, Ş. İ. Birbil, and K. Bülbül. Simultaneous column-and-row generation for large-scale linear programs with column-dependent-rows. *Mathematical Programming*. To appear.

[66] İ. Muter, Ş. İ. Birbil, K. Bülbül, G. Şahin, H. Yenigün, D. Taş, and D. Tüzün. Solving a robust airline crew pairing problem with column generation. *Computers & Operations Research*, 40(3):815–830, 2013.

[67] R. Nissen and K. Haase. Duty-period-based network model for crew rescheduling in European airlines. *Journal of Scheduling*, 9(3):255–278, 2006.

[68] N. Papadakos. Practical enhancements to the Magnanti-Wong method. *Operations Research Letters*, 36(4):444–449, 2008.

[69] N. Papadakos. Integrated airline scheduling. *Computers & Operations Research*, 36(1):176–195, 2009.

[70] J. Petersen, G. Sölveling, E. Johnson, J. Clarke, and S. Shebalov. An optimization approach to airline integrated recovery. *Transportation Science*, 46(4):482–500, 2012.

[71] W. Rei, J.-F. Cordeau, M. Gendreau, and P. Soriano. Accelerating Benders' decomposition by local branching. *INFORMS Journal on Computing*, 21(2):333–345, 2009.

[72] Research and Innovative Technology Administration. Understanding the reporting of causes of flight delays and cancellations. Technical report, Bureau of Transportation Statistics, *URL: www.bts.gov/help/aviation/html/understanding.html*, 2011.

[73] J. M. Rosenberger, E. L. Johnson, and G. L. Nemhauser. Rerouting aircraft for airline recovery. *Transportation Science*, 37(4):408–421, 2003.

[74] J. M. Rosenberger, E. L. Johnson, and G. L. Nemhauser. A robust fleet-assignment model with hub isolation and short cycles. *Transportation Science*, 38(3):357–368, 2004.

[75] A. Ruszczyński. A regularized decomposition method for minimizing a sum of polyhedral functions. *Mathematical Programming*, 35(3):309–333, 1986.

[76] D. M. Ryan and B. A. Foster. An integer programming approach to scheduling. In A. Wren, editor, *Computer Scheduling of Public Transport: Urban Passenger Vehicle and Crew Scheduling*, pages 269–280. North-Holland Publishing Company, Amsterdam, 1981.

[77] R. Sadykov and F. Vanderbeck. Column generation for extended formulations. *EURO Journal on Computational Optimization*, 1(1-2):81–115, 2013.

[78] T. Santoso, S. Ahmed, M. Goetschalckx, and A. Shapiro. A stochastic programming approach for supply chain network design under uncertainty. *European Journal of Operational Research*, 167(1):96–115, 2005.

[79] B. C. Smith and E. L. Johnson. Robust airline fleet assignment: imposing station purity using station decomposition. *Transportation Science*, 40(4):497–516, 2006.

[80] C. Sriram and A. Haghani. An optimization model for aircraft maintenance scheduling and re-assignment. *Transportation Research Part A: Policy and Practice*, 37(1):29–48, 2003.

[81] M. Stojković and F. Soumis. An optimization model for the simultaneous operational flight and pilot scheduling problem. *Management Science*, 47(9):1290–1305, 2001.

[82] M. Stojković and F. Soumis. The operational flight and multi-crew scheduling problem. *Yugoslav Journal of Operations Research*, 15(1):25–48, 2005.

[83] M. Stojković, F. Soumis, and J. Desrosiers. The operational airline crew scheduling problem. *Transportation Science*, 32(3):232–245, 1998.

[84] D. Teodorovic and S. Guberinic. Optimal dispatching strategy on an airline network after a schedule perturbation. *European Journal of Operational Research*, 15(2):178–182, 1984.

[85] D. Teodorović and G. Stojković. Model for operational daily airline scheduling. *Transportation Planning and Technology*, 14(4):273–285, 1990.

[86] D. Teodorović and G. Stojković. Model to reduce airline schedule disturbances. *Journal of Transportation Engineering*, 121(4):324–331, 1995.

[87] B. G. Thengvall, J. F. Bard, and G. Yu. Balancing user preferences for aircraft schedule recovery during irregular operations. *IIE Transactions*, 32(3):181–193, 2000.

[88] B. G. Thengvall, J. F. Bard, and G. Yu. A bundle algorithm approach for the aircraft schedule recovery problem during hub closures. *Transportation Science*, 37(4):392–407, 2003.

[89] B. G. Thengvall, G. Yu, and J. F. Bard. Multiple fleet aircraft schedule recovery following hub closures. *Transportation Research Part A: Policy and Practice*, 35(4):289–308, 2001.

[90] F. Vanderbeck. Branching in branch-and-price: a generic scheme. *Mathematical Programming*, 130(2):249–294, 2011.

[91] G. Wei, G. Yu, and M. Song. Optimization model and algorithm for crew management during airline irregular operations. *Journal of Combinatorial Optimization*, 1(3):305–321, 1997.

[92] O. Weide. *Robust and integrated airline scheduling.* PhD thesis, University of Auckland, 2009.

[93] O. Weide, D. Ryan, and M. Ehrgott. An iterative approach to robust and integrated aircraft routing and crew scheduling. *Computers & Operations Research*, 37(5):833–844, 2010.

[94] C. Wu. *Airline operations and delay management: insights from airline economics, networks, and strategic schedule planning.* Ashgate, Farnham, UK, 2010.

[95] S. Yan and C.-G. Lin. Airline scheduling for the temporary closure of airports. *Transportation Science*, 31(1):72–82, 1997.

[96] S. Yan and Y.-P. Tu. Multifleet routing and multistop flight scheduling for schedule perturbation. *European Journal of Operational Research*, 103(1):155–169, 1997.

[97] S. Yan and D.-H. Yang. A decision support framework for handling schedule perturbation. *Transportation Research Part B: Methodological*, 30(6):405–419, 1996.

[98] S. Yan and H.-F. Young. A decision support framework for multi-fleet routing and multi-stop flight scheduling. *Transportation Research Part A: Policy and Practice*, 30(5):379–398, 1996.

[99] J. W. Yen and J. R. Birge. A stochastic programming approach to the airline crew scheduling problem. *Transportation Science*, 40(1):3–14, 2006.

[100] E. J. Zak. Row and column generation technique for a multistage cutting stock problem. *Computers & Operations Research*, 29(9):1143–1156, 2002.