Solving the integrated airline recovery problem using column-and-row generation

Stephen J Maher

School of Mathematics and Statistics, University of New South Wales, Sydney NSW 2052, Australia.

Abstract

Airline recovery presents very large and difficult problems requiring high quality solutions within very short time limits. To improve computational performance, various solution approaches have been employed, including decomposition methods and approximation techniques. There has been increasing interest in the development of efficient and accurate solution techniques to solve an integrated airline recovery problem. In this paper, an integrated airline recovery problem is developed, integrating the schedule, crew and aircraft recovery stages, and is solved using column-and-row generation. A general framework for column-and-row generation is presented as an extension of current generic methods. This extension considers multiple secondary variables and linking constraints and is proposed as an alternative solution approach to Benders' decomposition. The application of column-and-row generation to the integrated recovery problem demonstrates the improvement in the solution runtimes and quality compared to a standard column generation approach. Column-and-row generation improves solution runtimes by reducing the problem size and thereby achieving faster execution of each LP solve. As a result of this evaluation, a number of general enhancement techniques are identified to further reduce the runtimes of column-and-row generation. This paper also details the integration of the row generation procedure with branch-and-price, which is used to identify integral optimal solutions.

Key words: airline recovery, column generation, row generation

1 Introduction

Disruptions are very common in the airline industry, greatly impacting the realised operational performance. To mitigate the effect of these disruptions, intervention by the airline is necessary to maintain the many operational requirements of aircraft and crew. Consequently, any disruption results in a significant increase to an airlines operational costs related to additional crew overtime and increased fuel usage. Due to the significant associated costs, the use of efficient and accurate recovery processes is of great importance to the airline industry.

The airline recovery problem, similar to the planning problem, is a very large and complex problem commonly broken into a number of smaller, more tractable sequential stages. These stages are broadly categorised as schedule, aircraft, crew and passenger recovery, also defining clear boundaries for research in this area. The complete recovery process is a sequential problem, where each stage is solved to optimality and fixed for use in subsequent stages. Analogous to the planning problem, this typically results in suboptimal, or even infeasible, solutions due to little interaction between the stages. The integration of two or more stages in the planning problem has been demonstrated to provide higher quality solutions [15, 25, 29], as such there is a similar expectation for airline recovery. The focus of this paper is an integrated airline recovery problem, integrating the schedule, aircraft and crew recovery.

A critical constraint on the airline recovery problem is the allowable time limit to find an optimal solution. Generally the solution to the airline recovery problem is required within minutes of a disruptive event, prompting the development of many solution approaches to achieve this. The proposed solution approaches to achieve improved runtime performance varies between each of the recovery stages. The integration of multiple recovery stages presents very difficult and complex problems, which are commonly solved with the use of decomposition techniques. Unfortunately, the decomposition techniques employed, such as Benders' decomposition, do not provide a guarantee of integral optimality. A contribution of this paper is the development of a column-and-row generation framework as an exact solution approach to solve the integrated airline recovery problem.

1.1 Airline recovery literature

The most common method employed to improve the runtimes of the aircraft recovery problem is through the network design. There are three classes of network design that have been employed, the time-line, time-band and connection networks. The time-line network is a very popular approach used for the aircraft recovery problems, with Jarrah *et al.* [20] presenting an early example using this network description. The work of [20] is extended by Cao and Kanafani [12, 13], integrating the two models developed by [20] and implementing additional recovery policies. The use of the time-line network is developed further by Yan and Yang [40], presenting a unique design that concisely describes the effect of airline disruptions on the planned schedule. The time-band network is presented by Argüello [6] to approximate the flight network by aggregating activities at airports into discrete time bands. Therefore, the number of nodes used to define the recovery network is reduced. This network description has been demonstrated by Bard *et al.* [8] and Eggenberg *et al.* [16] to improve the solution runtimes of the aircraft recovery problem. Finally, Rosenberger *et al.* [31] present an aircraft recovery problem using the classical connection network, which provides a concise description of the recovery flight schedule. 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.

The crew recovery problem is a very complex and difficult problem having a significant impact on the

operational cost of an airline. Similar to the aircraft recovery problem, a number of unique approaches have been proposed to improve solution runtimes. These approaches involve reducing the problem size by selecting only a subset of crew and flights, fixing the flight schedule by using the solution to the aircraft recovery problem (part of the sequential recovery process), or implementing only a selection of recovery policies.

Solving the crew recovery problem with a fixed flight schedule is first presented by Wei *et al.* [39], attempting to repair any pairings affected by a schedule disruption. Fast solution runtimes are achieved with the use of a branch-and-bound heuristic designed with consideration to the actions of an AOCC. Another approach using a fixed flight schedule is the crew rescheduling problem proposed by Stojković *et al.* [36]. In [36], the problem size is reduced by defining a recovery window to identify a set of disruptable flights. Finally, Medard and Sawhney [24] and Nissen and Haase [28] present crew recovery problems that are solved using heuristic approaches.

The crew recovery problem permitting the use of flight delays and cancellations is more complex problem than the comparable fixed flight schedule problems. An example of such a problem is presented by Stojković and Soumis [34, 35] as extensions of Stojković *et al.* [36], introducing flight delays and constructing individual pairings for each crew member. Lettovsky *et al.* [23] present a crew recovery problem that introduces the use of flight cancellations, extending Johnson *et al.* [21]. In [23], a number of approaches to reduce the computational time of the recovery algorithm are proposed, including a search scheme to identify the disruptable crew and compact storage for the generated columns. Finally, a novel approach to the crew recovery problem is developed by Abdelghany *et al.* [1], partitioning flights by their resource independence. The partitioning process reduces the solution runtimes by defining a series of distinct recovery problems that are more readily solvable than the original problem.

Many solution approaches involve the approximation of the recovery problem to improve runtimes. For the aircraft recovery problem, these approximations are made of either the network or the equipment included in the model. Similarly, the runtimes for the crew recovery problem are reduced through the use of heuristics or the selection of included crew. Column-and-row generation is presented in this paper as an alternative, exact solution approach to improve runtimes. The results will show that column-and-row generation solves the IRP in short runtimes without requiring any approximation of the affected crew and aircraft. These results will provide a lower bound on the solution runtime reduction for practical applications, which can be improved further using approximation techniques.

The sequential process used to solve the airline recovery problem generally results in suboptimal solutions due to fixing the solution at each stage in the process. The PhD thesis of Lettovsky [22] is an early attempt to develop a recovery model integrating multiple stages. This model integrates all aspects of the recovery process; schedule, aircraft, crew and passenger recovery problems; and solved using Benders' decomposition. Further exploring the idea of employing Benders' decomposition for the integrated recovery problem, Petersen *et al.* [30] present a model that shares some of the characteristics

of Lettovsky [22]. Petersen *et al.* [30] describe the implementation of this integrated recovery model, providing an evaluation against a set of major disruptions. The reported runtimes are within the specified goal of 30 minutes for the selected set of disruption scenarios. Extending the novel approach by Abdelghany *et al.* [1], Abdelghany *et al.* [2] develop a recovery model integrating aircraft, pilots and flight attendants. The results from experiments demonstrate significant delay reductions that are achievable within very short runtimes.

Unfortunately the partitioning process of Abdelghany *et al.* [2] and the use of Benders' decomposition by Lettovsky [22] and Petersen *et al.* [30] does not guarantee integer optimality of the integrated recovery problem. A contribution of this paper is the development of a general column-and-row generation framework that is applied to problems commonly solved by Benders' decomposition. Further, column-and-row generation is an exact approach that provides a guarantee of near integer optimal solutions for the integrated airline recovery problem.

1.2 Column-and-row generation

Column-and-row generation is a solution approach that extends standard column generation to reduce problem complexity and improve solution runtimes. This solution approach involves the simultaneous generation of variables and structural constraints. A key feature of column-and-row generation is the reduction in the size of the master problem through the elimination of constraints. The problem size reduction achieves runtime improvements through faster LP solves and quicker execution of the column generation subproblems.

There have been a number of generic schemes developed for column-and-row generation [18, 26, 33], however these schemes do not directly apply to the integrated recovery problem considered in this paper. Fragioni and Gendron [18] and Sadykov and Vanderbeck [33] present schemes to solve mixed integer programs by dynamically generating structural constraints. The solution process in these two approaches involves identifying an upper bound from the linear relaxation and a lower bound from the Lagrangian subproblem. A feature of the reformulations in [18] and [33], is the ability to ignore the dual variables associated with the missing constraints without any loss of correctness in the algorithm. This is not the case for the integrated recovery problem, requiring a row generation procedure to calculate the dual variables of the missing constraints. The column-and-row generation approach presented here is similar to that of Muter *et al.* [26], however the integrated recovery problem does not satisfy all three assumptions required to apply their solution technique. As a contribution to the column-and-row generation approach, this paper extends Muter *et al.* [26] by considering problems with multiple secondary variables and linking constraints. This extension requires additional processes in the row generation procedure to ensure the accurate calculation of an optimal dual solution. The framework developed in this paper is presented algorithmically to succinctly describe the implementation.

Column-and-row generation is employed to solved various applied integer programming problems.

Example applications of this solution approach are Zak [42] with the multi-stage cutting stock problem, Avella *et al.* [7] solving a time-constrained routing problem and Muter *et al.* [27] to solve a robust crew pairing problem. While these papers present implementations of column-and-row generation there is little evaluation against a standard column generation approach. A contribution of this paper is such an evaluation using the integrated airline recovery problem as an example. The integrated airline recovery problem is a large-scale, real-world optimisation problem that provides an appropriate test bed for the column-and-row generation framework. This evaluation will demonstrate the improvement in solution runtime and quality compared to column generation and identify a number of enhancements that contribute to the general column-and-row generation approach. The enhancement techniques identified are a variation on the number of rows added in the row generation procedure and a row warm-up process. Since the framework developed in this paper extends Muter *et al.* [26], the evaluation performed demonstrates the strength of the framework by [26]

The various implementations of column-and-row generation have very little discussion regarding row generation in the branch-and-price algorithm. For example, Zak [42] only solves the LP of the multi-stage cutting stock problem using column-and-row generation, this is also the case in Wang and Tang [38] for the batch machine scheduling problem. Also, Frangioni and Gendron [17, 18] solve the multicommodity capacitated network design problem with column-and-row generation at the root node and then employ branch-and-bound to identify integer solutions. Sadykov and Vanderbeck [33] mention the use of their approach in a branch-and-price algorithm, however no details are provided regarding the implementation. Finally, the framework by Muter *et al.* [26] is designed to solve large-scale linear programming problems, as such the integration with a branch-and-price algorithm is not considered. A contribution of this paper is a clear description of the application of row generation in a branch-and-price algorithm.

1.3 Outline of paper

This paper presents an integrated airline recovery problem, integrating the schedule, aircraft and crew recovery stages. The integrated airline recovery problem is used in this paper as a real-world example to demonstrate the ability of column-and-row generation to improve solution runtimes and quality. The master problem for the integrated recovery problem is presented in Section 2, which is solved using column generation to provide benchmark results. A general framework for the column-and-row generation solution approach is presented in Section 3. The application of column generation and column-and-row generation is discussed in Section 4, including the description of enhancement techniques. To demonstrate the benefits of solving the integrated recovery problem by column-and-row generation, a comparison with the results produced using column generation is made in both solution quality and runtime. These results are presented in Section 5. The conclusions provided in Section 6 aim to present the technique of column-and-row generation as an alternative solution method for integrated airline and transportation problems.

2 Integrated Recovery Problem - IRP

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. The link between the aircraft and crew recovery problems in the IRP is provided by the flight cancellation and delay decisions and specific flights allocated to each aircraft and crew.

The notation used to describe the IRP is introduced in Tables 1 and 2. For this problem, the set of all crew is given by K, indexed by k, and the set of all aircraft is given by R, indexed by r. The set K also includes all available reserve crew K^{res}. As an extension on 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 the optimal allocation of all available resources.

K	is the set of all planned and reserve crew in the model, indexed by k
K^{res}	is the set of all reserve crew, $K^{res} \subset K$
R	is the set of all aircraft in the model, indexed by r
P^k, P^r	is the set of all strings p assigned to crew k or aircraft r respectively
N	is the set of all flights j
T^K, T^R	is the set of crew bases/overnight airports t for crew and aircraft respectively
N^D	is the set of all 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, $N^K_{in} \subset N \cup T^K$ and $N^R_{in} \subset N \cup T^R$
N^K_{out}, N^R_{out}	is the set of all carry-out activities j, flights and termination nodes, for crew and aircraft respectively, $N^K_{out} \subset N \cup T^K$ and $N^R_{out} \subset N \cup T^R$
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 all 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, $E^D = \{(i_u, j_v) \in E \mid i_u \in \hat{N}^D \vee j_v \in \hat{N}^D\}$

Table 1: Sets used in the IRP.

2.1 Recovery flight schedule and connection network

A single day flight schedule is used to describe and evaluate the IRP, with the set of flights in the schedule given by N. A critical aspect of the model is the use of a recovery window that specifies the time allocated to return the operations back to plan. The recovery window is described as a time period which defines the set of disruptable flights $N^D \subseteq N$. The set N^D contains all flights that are primarily

x_p^k	= 1 if crew k uses string p, 0 otherwise
y_p^r	= 1 if aircraft r uses string p, 0 otherwise
c_p^k, c_p^r	= the cost of using 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
$b_{i_u j_v p}^k$, $b_{i_u j_v p}^r$	= 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 copy v for flight j is in string p for crew k or aircraft r respectively, 0 otherwise
o_{tp}^r	= 1 if string p, assigned to aircraft r, terminates at airport t 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
$\bar{o}_t^r$	= 1 if the planned string for aircraft r terminates at airport t within the recovery window, 0 otherwise

Table 2: Variables used in the IRP.

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 introduces the concepts of carry-in and carry-out activities. A carry-in activity is a flight or origination airport that is assigned to a crew or aircraft directly preceding the disruption, contained in the sets N_{in}^K and N_{in}^R respectively. A carry-out activity is a flight or termination airport assigned to a crew or aircraft immediately after the end of the recovery window, contained in the sets N_{out}^K and N_{out}^R respectively. Now, the carry-in activity describes an origination airport when a planned crew duty or aircraft routing begins after the start of the disruptive event. In a similar manner, if a planned crew duty or aircraft routing concludes before the end of the recovery window, the carry-out activity is defined as the terminating airport.

To achieve an efficient solution approach for the IRP, the recovery policy of flight delays is implemented using flight copies. The flight copies technique involves the multiple duplication of each flight contained in N with each copy assigned a progressively later departure time. For each flight $j \in N$ the set of discrete copies is given by U_j, and a flight-copy pair is described by j_v, where $v \in U_j$. Since the set of all flights N is partitioned into disruptable and non-disruptable sets, the set of discrete flight copies requires a different definition for each partition. As such, for all non-disruptable flights, $j \in N \backslash N^D$, only one copy is included to represent the original scheduled departure, i.e. $U_j = \{0\}$. Further, for all disruptable flights, $j \in N^D$, the set of flight copies U_j contains a copy representing the original departure time and at least one additional copy to represent some delay on that flight. 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, representing 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) 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. Similarly, all feasible connections for aircraft contained in C^R require a *minimum turn time* between the connecting flights.

2.2 Aircraft routes and crew duties

The modelling approach for the IRP is based upon the string formulation introduced by Barnhart *et al.* [9]. In the IRP, a flight string is defined as a set of connected flights from a carry-in to a carry-out activity. Using the flight-copy notation 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^k_{jp} and a^r_{jp} 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 port, or will terminate at a carry-out flight. If the flight string assigned to an aircraft terminates within the recovery window, the parameter o^r_{tp} describes the terminating airport t for aircraft r. Flight cancellations are implemented as a recovery policy for the IRP, so the flow balance of the original schedule is not maintained. To ensure enough aircraft are positioned at each airport t to operate the schedule for the following day, the minimum number of required aircraft must be specified. Now, the number of planned aircraft strings terminating at end-of-day locations within the recovery period is given by $\sum_{r \in R} \bar{o}^r_t, \forall t \in T$, where $\bar{o}^r_t$ indicates that aircraft r terminates at airport t. Therefore this expression defines the minimum number of aircraft required to terminate at each end-of-day location t 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 $b^k_{i_u j_v p}$ and $b^r_{i_u j_v p}$ indicate the inclusion of connection (i_u, j_v) on string p for crew and aircraft respectively.

2.2.1 Legality of crew duties

There a numerous rules that dictate the construction of feasible flight strings for crew that must be strictly adhered to in the recovery problem. A good review of the crew scheduling problem is presented by Barnhart *et al.* [10], discussing the numerous crew rules. 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.

The origination and termination locations are critical in the construction of legal crew duties. Each crew is employed at one of many crew bases throughout the network, so 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 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 crucial in managing the effects of fatigue. There are two rules related to working hours modelled in the IRP, a maximum number of flying hours and limit on the 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 of more that 8 hours flying is performed in a 24 hour period [10]. This rule must be considered in the construction of crew pairings, however it may be reviewed in the recovery of crew duties. By ensuring that crew only perform 8 hours flying in a single day and given that individual crew members have planned work starting times, the 8-in-24 rule will not be violated in most cases. 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.

2.3 Recovery policies

The set of recovery policies implemented for 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 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 flight j is cancelled at a cost of d_j.

Following a disruption, it is not always possible for every crew member to operate the originally planned flight strings as expected. Hence, the crew specific recovery policies of deadheading and the use of reserve crew are employed. Crew deadheading transports 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. In addition, 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 crew deadheading 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. Hence, reserve crew are employed to operate crew duties from each crew base. Reserve crew are a limited, costly resource, as such their use in the solution to the IRP is minimised by the addition of a penalty to the cost c_p^k for each duty they perform.

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,

$$\min \quad \sum_{k\in K}\sum_{p\in P^k} c_p^k x_p^k + \sum_{j\in N^D}\sum_{v\in U_j} g^{DHD}\kappa_j^{v+} + \sum_{k\in K} g^{DHB}\nu_k + \sum_{r\in R}\sum_{p\in P^r} c_p^r y_p^r + \sum_{j\in N^D} d_j z_j, \tag{1}$$

$$\text{s.t.} \quad \sum_{k\in K}\sum_{p\in P^k} a_{jp}^k x_p^k - \sum_{v\in U_j} \kappa_j^{v+} + z_j = 1 \quad \forall j \in N^D, \tag{2}$$

$$\sum_{k\in K}\sum_{p\in P^k} a_{jp}^k x_p^k = 1 \quad \forall j \in N_{out}^K, \tag{3}$$

$$\sum_{r\in R}\sum_{p\in P^r} a_{jp}^r y_p^r + z_j = 1 \quad \forall j \in N^D, \tag{4}$$

$$\sum_{r\in R}\sum_{p\in P^r} a_{jp}^r y_p^r = 1 \quad \forall j \in N_{out}^R, \tag{5}$$

$$\sum_{r\in R}\sum_{p\in P^r} o_{tp}^r y_p^r \geq \sum_{r\in R} \bar{o}_t^r \quad \forall t \in T, \tag{6}$$

$$\sum_{k\in K}\sum_{p\in P^k} a_{jp}^{kv} x_p^k - \kappa_j^{v+} + \kappa_j^{v-} = 1 \quad \forall j \in N^D, \forall v \in U_j, \tag{7}$$

(IRP)

$$\sum_{k\in K}\sum_{p\in P^k} b_{i_u j_v p}^k x_p^k - \sum_{r\in R}\sum_{p\in P^r} b_{i_u j_v p}^r y_p^r \leq 0 \quad \forall (i_u, j_v) \in E^D, \tag{8}$$

$$\sum_{k\in K}\sum_{p\in P^k} a_{jp}^{kv} x_p^k - \kappa_j^{v+} - \sum_{r\in R}\sum_{p\in P^r} a_{jp}^{rv} y_p^r = 0 \quad \forall j \in N^D, \forall v \in U_j, \tag{9}$$

$$\sum_{p\in P^k} x_p^k + \nu_k = 1 \quad \forall k \in K \backslash K^{res}, \tag{10}$$

$$\sum_{p\in P^k} x_p^k \leq 1 \quad \forall k \in K^{res}, \tag{11}$$

$$\sum_{p\in P^r} y_p^r \leq 1 \quad \forall r \in R, \tag{12}$$

$$x_p^k \in \{0,1\},\ \forall k \in K, \forall p \in P^k, \quad y_p^r \in \{0,1\},\ \forall r \in R, \forall p \in P^r, \tag{13}$$

$$z_j \in \{0,1\},\ \forall j \in N^D, \quad \nu_k \in \{0,1\},\ \forall k \in K, \tag{14}$$

$$\kappa_j^{v+} \geq 0, \kappa_j^{v-} \geq 0, \quad \forall j \in N^D, \forall v \in U_j. \tag{15}$$

The IRP is defined by equations (1)-(14) with the objective 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 (2) and (4). Additionally, the constraints (3) and (5) ensure that each carry-out flight is serviced by crew and aircraft in the recovered solution. Respecting the carry-out flight coverage ensures that the solution to the IRP positions the crew and aircraft to continue the activities as planned, following the end of the recovery window.

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 (6).

The recovery policy of crew deadheading within a duty is implemented through the surplus crew count constraints (7). This set of constraints ensure that crew deadheading is only permitted on flight-copy pairs that are operated by at least one crew.

In the IRP, the integration between the crew and aircraft variables is described by the short connection and delay consistency constraints, equations (8) and (9) respectively. The short connection constraints (8) ensure that crew are only permitted to use connection $(i_u, j_v) \in E^D$ only if an aircraft is also using the same connection. The delay consistency constraints (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. Since there exists one delay consistency constraint for each flight-copy pair, this set of constraints grows very quickly with an increase in 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, captured by constraints (10). This is not true for the reserve crew since they former is not required to perform any duties during recovery, which is captured by the inequality in constraints (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 (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 a common objective to quickly return operations to plan. Returning operations to plan is a consequence of the carry-out activity coverage at the conclusion of the recovery window. Hence, the shorter the recovery window, the quicker operations are returned 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 sophisticated solution techniques.

3 Column-and-Row Generation

This paper demonstrates the improvements in the solution runtimes and quality for the IRP achieved by employing column-and-row generation. As discussed in Section 1.2, there have been a number of generic column-and-row generation approaches that have been developed [18,26,33]. Unfortunately, these generic approaches do not directly apply to the IRP, as such an alternative framework will be presented in the following sections. The framework developed in this paper is an extension of Muter *et al.* [26] and is presented as an alternative solution approach to Benders' decomposition. The key contributions of this framework are i) the extension of [26] to consider multiple sets of secondary variables and linking constraints, ii) the application of column-and-row generation to a problem structure that is commonly solved by Benders' decomposition, and iii) the explicit evaluation against a standard column generation approach, identifying various enhancement techniques.

3.1 Features of the Column-and-Row Generation Framework

The solution approach presented by Muter *et al.* [26] 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 with 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.

Both the RMP and SRMP are solved by column generation and the identical subproblems are used in the solution process for the two problems. However, the SRMP is formed by eliminating structural constraints from the RMP, so variable fixings in the column generation subproblems are used to restrict the set of all possible columns. By applying the variable fixing to the column generation subproblem, all feasible solutions to the SRMP are feasible for the RMP and the original problem.

Column-and-row generation is a solution approach proposed to solve large-scale linear programs. The implementation of column-and-row generation in a branch-and-price framework to solve mixed-integer programs has not been previously published. As a contribution of this paper, an efficient method for applying column-and-row generation at each node in the branch-and-bound tree is explicitly discussed.

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 3.1.1 and 3.1.2 respectively. Finally, a general algorithm for the row generation procedure will be presented in Section 3.1.3.

3.1.1 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.* [26].

This section focuses on problems solved by column generation, as such 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_x^{iL} and A_y^{iL} 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}, resulting in 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 columns 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

$$\min \quad c^x x + \sum_i c^i y^i, \tag{16}$$

$$\text{s.t.} \quad A^x x = b, \tag{17}$$

$$A^i y^i = b^i \ \ \forall i, \tag{18}$$

$$A_x^{iL} x - A_y^{iL} y^i = 0 \ \ \forall i, \tag{19}$$

$$x \geq 0, \ \ y^i \geq 0. \tag{20}$$

SRMP

$$\min \quad \bar{c}^x \bar{x} + \sum_i \bar{c}^i \bar{y}^i, \tag{21}$$

$$\text{s.t.} \quad \bar{A}^x \bar{x} = b, \tag{22}$$

$$\bar{A}^i \bar{y}^i = b^i \ \ \forall i, \tag{23}$$

$$\bar{A}_x^{iL} \bar{x} - \bar{A}_y^{iL} \bar{y}^i = 0 \ \ \forall i, \tag{24}$$

$$\bar{x} \geq 0, \ \ \bar{y}^i \geq 0. \tag{25}$$

3.1.2 Calculation of dual solutions

To demonstrate the correctness of the column-and-row generation approach 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 on the construction of $\tilde{y}^i$ that is exploited in Theorem 3.1.1. The dummy variables contained in $\tilde{y}^i$ also have non-zero elements in the rows of (18), thus the matrices $\tilde{A}^i$ must be introduced. Therefore, the matrix representation of the RMP$'$ is given by,

$$\text{(RMP}') \quad \begin{aligned} \min \quad & \bar{c}\bar{x} + \sum_i \{\bar{c}^i\bar{y}^i + \tilde{c}^i\tilde{y}^i\}, && (26)\\ \text{s.t.} \quad & \bar{A}\bar{x} = b, && (27)\\ & \bar{A}^i\bar{y}^i + \tilde{A}^i\tilde{y}^i = b^i \ \ \forall i, && (28)\\ & \bar{A}_x^{iL}\bar{x} - \bar{A}_y^{iL}\bar{y}^i = 0 \ \ \forall i, && (29)\\ & -\tilde{A}_y^{iL}\tilde{y}^i = 0 \ \ \forall i, && (30)\\ & \bar{x} \geq 0, \ \ \bar{y}^i \geq 0, \ \ \tilde{y}^i \geq 0. && (31) \end{aligned}$$

For each row in equations (27)-(30) there is an equivalent row in equations (17)-(19). It is assumed that in this formulation the optimal solution to the RMP$'$ is not the optimal solution to the original problem. Thus, additional columns with a negative reduced cost can be found by solving the column generation subproblems for the primary and secondary variables.

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 optimal.

Lemma 3.1.1. *The optimal primal solution to the SRMP is an optimal primal solution to the RMP$'$.*

Proof. The constraints (30) 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 (30) 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. □

There are two major steps in the procedure to calculate an optimal dual solution for the RMP$'$ using the solution to the SRMP. Firstly, the constraints (22)-(24) in the SRMP are identical to the constraints (27)-(29), 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 (30), 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 Φ_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 $\bar{\Phi}_i$ and all eliminated rows are contained in $\Phi_i \backslash \bar{\Phi}_i$. Finally, the dual variables for each row in A_y^{iL} is given by $\boldsymbol{\theta}^i = \{\theta_u^i | u \in \Phi_i\}$. 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 θ_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 1.

Algorithm 1 Computing a feasible dual solution

1: Assume that $\theta_u^i = 0$ 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 $v \in \Phi_i \backslash \bar{\Phi}_i, v \neq u$.
2: Solve the column generation subproblem to identify variable $\hat{y}$ that has the minimum reduced cost $\hat{c}^i$.
3: Set $\theta_u^i = -\hat{c}^i$.

This calculation procedure relies on the structure of the RMP$'$ and the form of the dummy variables that populate the rows $u \in \Phi_i \backslash \bar{\Phi}_i$. The reasoning provided here draws upon the discussion presented in the column-and-row generation framework by Muter *et al.* [26]. For $\hat{y}$, found by Algorithm 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 the rows of $\tilde{A}_y^{iL}$, the construction of the RMP$'$ forces $\hat{y} = 0$ upon 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 1 is at least zero. This requirement ensures that the dual solutions that are computed for $\boldsymbol{\theta}^i$ are feasible for the RMP$'$. The following theorem will prove the feasibility of the computed values for $\boldsymbol{\theta}^i$ and that the resulting feasible dual solution is also optimal.

Theorem 3.1.1. *The dual solutions computed for $\boldsymbol{\theta}^i$ 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 $\boldsymbol{\theta}^i$ using Algorithm 1 are feasible for the RMP$'$. For this proof the variable $\bar{\theta}_u^i$ is assumed to have a value that satisfies all dual constraints of the RMP$'$. Additionally, the reduced cost of variable $y \in \tilde{y}^i$ is given by $\bar{c}_y^i$.

Algorithm 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 $\theta_u^i = 0$. 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 $\hat{y}$, $\hat{c}^i = \bar{c}_y^i - \bar{\theta}_u^i$. In Algorithm 1, the value of θ_u^i is set to $-\hat{c}^i$, hence $\bar{c}_y^i - \bar{\theta}_u^i + \theta_u^i = 0$. Therefore, setting $\theta_u^i = -\hat{c}^i$ 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 θ_u^i exists in at least one dual constraint. Lets assume that setting $\theta_u^i = -\hat{c}^i$ violates a constraint in the dual of the RMP$'$. This implies that $\bar{c}_y^i$ calculated using $\theta_u^i = -\hat{c}^i$ in place of $\bar{\theta}_u^i$ is negative, i.e. $\bar{c}_y^i - \bar{\theta}_u^i + \theta_u^i < 0$. Since $\hat{c}^i + \theta_u^i = 0$, then $\hat{c}^i > \bar{c}_y^i - \bar{\theta}_u^i$. Now, step 2 of Algorithm 1 identifies $\hat{y}$ that has the minimum reduced cost, so $\hat{c}^i > \bar{c}_y^i - \bar{\theta}_u^i$ is a contradiction. Therefore, $\bar{c}_y^i - \bar{\theta}_u^i + \theta_u^i \geq 0$ must be true, hence the computed value for θ_u^i satisfies all dual constraints.

The first step of this proof has established that the dual solutions calculated for $\boldsymbol{\theta}^i$ using Algorithm 1 are feasible for the RMP$'$. Therefore, the calculated values for $\boldsymbol{\theta}^i$ can be used as the dual solutions for

the constraints (30). A feasible dual solution for the RMP′ is then simply constructed by equating the solutions to the dual variables representing constraints (27)-(29) 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 3.1.1 that the optimal primal solution to the SRMP is also optimal for the RMP′. In addition, the solutions to the dual variables representing constraints (27)-(29) are equated to the dual solutions of the SRMP. Since the right hand side of the constraints represented by equations (30) 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. □

3.1.3 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′. It is likely, due to the eliminated constraints, that the columns identified during this procedure are infeasible for the SRMP. Such columns are identified by displaying at least one non-zero element in the rows contained in $\Phi_i \backslash \bar{\Phi}_i$. If columns infeasible for the SRMP are found, then u must be added to $\bar{\Phi}_i$ 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 procedure is detailed in Algorithm 2.

Algorithm 2 Row generation algorithm

Require: An optimal solution to the SRMP.

1: Set the dual values for rows (27)-(29) to the dual solutions for the rows (22)-(24).
2: **for all** secondary variable sets i **do**
3: **for all** rows u contained in $\tilde{A}_y^{iL}$ **do**
4: Execute Algorithm 1 to compute the value of θ_u^i.
5: **end for**
6: **end for**
7: By Theorem 3.1.1 an optimal dual solution for the RMP′ has been computed.
8: Solve the column generation subproblem for the primary variables to identify variables feasible for the RMP′.
9: **if** a negative reduced cost column has at least one non-zero entry in the rows of $\tilde{A}_y^{iL}$ **then**
10: add the rows with non-zero entries in $\tilde{A}_y^{iL}$ to $\bar{A}_y^{iL}$.
11: **end if**

The column-and-row generation solution approach terminates when no favourable rows are identified by Algorithm 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$'$, and the SRMP, is optimal for the original problem.

3.2 Column-and-Row Generation Solution Algorithm

The column-and-row generation solution algorithm implemented in this paper is developed by combining the fundamental features of the approach developed throughout Section 3.1. The first stage of the solution algorithm involves the formulation of the SRMP, which is detailed in Section 3.1.1. Using the solution to the SRMP, Section 3.1.2 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 3.1.3, executes Algorithm 2 to identify favourable rows for the SRMP. The complete column-and-row generation solution algorithm is given by Algorithm 3.

Algorithm 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 by column generation to optimality.
5: Use Algorithm 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 2
7: The solution to the SRMP is the optimal solution to the original problems

4 Applying the Column-and-Row Generation Framework

The framework presented in Section 3 describes the features of column-and-row generation that extend the standard column generation approach. Column generation is a critical aspect of column-and-row generation, as such its implementation for the IRP will be discussed in Section 4.1. This will be followed by a review of the features presented in Section 3.1, detailing the application to the IRP in Section 4.2.

4.1 Column Generation

The formulation of the IRP presents two sets of variables for which column generation is applied. These variables are related to crew duties and aircraft routes, which are defined as flight strings. While each of the variables types have similar structures, there are specific rules governing their generation. Hence, two individual column generation subproblems are required in the solution process. 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_{IRP}) is defined by including only a subset of all possible columns, $\bar{P}^k \subseteq P^k$ and $\bar{P}^r \subseteq P^r$, 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 (2)-(3) and (4)-(5), respectively. The dual variables for the aircraft end-of-day location constraints (6) are defined by $\boldsymbol{\epsilon} = \{\epsilon_t, \forall t \in T\}$. The dual variables for the surplus crew count constraints (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) and the delay consistency constraints (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, (10)-(11) and (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 $\bar{P}^k$ and $\bar{P}^r$.

4.1.1 Crew Pairing Subproblem

The crew duty subproblem (PSP^k) is solved as a shortest path problem with one source node and multiple sink nodes. The general form of the PSP^k is given by,

$$\hat{c}_p^k = \min_{p \in P^k} \left\{ RecDutyCost(k) - \sum_{j \in N^D \cup N_{out}^K} \alpha_j a_{jp}^k - \sum_{(i_u, j_v) \in E^D} \rho_{i_u j_v} b_{i_u j_v p}^k - \sum_{j \in N^D} \sum_{v \in U_j} \left\{ \eta_j^v + \gamma_j^v \right\} a_{jp}^{kv} - \delta^k \right\} \tag{32}$$

The set P^k is defined as the feasible region to a network flow problem, as such the PSP^k is solved as a resource constrained shortest path problem (RCSPP). The important features of the RCSPP used to solve the PSP^k is the origination of each crew k from a unique carry-in activity and the termination at any carry-out activity. In addition, the crew duty rules considered in this paper are the maximum number of flying and working hours, which are set at 8 and 13 respectively. In the column generation and column-and-row generation solution approaches the PSP^k is solve once for each $k \in K$ in each iteration.

In the objective function of (32), the complex cost structure used for crew remuneration is denoted by $RecDutyCost(k)$, which defines the additional crew cost resulting from recovery actions. The recovery crew cost $RecDutyCost(k)$ is a **max** function related to the number of flying hours, $fly(k)$, the number of working hours, $elapse(k)$, a minimum number of guaranteed hours, $minGuar$, and the originally

planned duty cost, $OrigDutyCost(k)$. As such, the expression for the cost of a duty in the IRP is given by

$$RecDutyCost(k) = \max\{0, \max\{fly(k), f_d \cdot elapse(k), minGuar\} - OrigDutyCost(k)\}, \tag{33}$$

where $minGuar$ is set at 6 hours [10] and f_d is a fraction which is airline specific and is set at $f_d = 5/8$ [10].

In consideration to the resource restrictions and complex cost structure, a multi-label shortest path algorithm is implemented to solve the PSP^k. 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, $H^2_{i_u l}$. Multiple labels are necessary to track any suboptimal paths to a node, based on the path length $\hat{c}_{i_u l}$, that have a favourable resource consumption. While it is possible to store every path that arrives at a node, this would be a very inefficient method to track resource consumption. As such, a dominance condition is used to reduce the number of labels stored at each node by eliminating any suboptimal paths demonstrating unfavourable resource consumption.

Definition 4.1.1. *(Dominance Condition)*

Given two labels at node i_u, ($\hat{c}_{i_u 1}$, $H^1_{i_u 1}$, $H^2_{i_u 1}$) and ($\hat{c}_{i_u 2}$, $H^1_{i_u 2}$, $H^2_{i_u 2}$), that are not equal. Label 1 dominates label 2 if

$$\hat{c}_{i_u 1} \leq \hat{c}_{i_u 2},\ H^1_{i_u 1} \leq H^1_{i_u 2} \text{ and } H^2_{i_u 1} \leq H^2_{i_u 2}.$$

Using Definition 4.1.1, the dominance of any new label arriving at a node is evaluated against all currently stored labels. There are three possible outcomes from the comparison between the new label and all currently stored labels. 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. At the sink node, the label that achieves the lowest cost is selected and the resulting path is the minimum reduced cost path.

The connection network described in Section 2 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 $\exists(i_u, j_v) \in C^K$ [4]. Using a topological ordering, the shortest path problem can be solved in $O(ml)$ time in the worst case, where m is the number of arcs in the acyclic directed graph and l is the maximum number of labels. A pulling algorithm is implemented, which solves the shortest path problem by "pulling" labels from previously processed nodes. Such a pulling algorithm for solving the shortest path problem on an acyclic directed graph is presented in Ahuja *et al.* [4]. This algorithm can be easily adapted to solve the PSP^k with multiple labels.

4.1.2 Aircraft Routing Subproblem

The column generation subproblem for the aircraft routing variables solves a shortest path problem from an origination location to one of the permissible termination locations. The general form of the column generation subproblem for the aircraft routing variables (PSP^r) is given by,

$$\hat{c}_p^r = \min_{p \in P^r} \left\{ c_p^r - \sum_{j \in N^D \cup N_{out}^K} \alpha_j a_{jp}^r - \sum_{t \in T} \epsilon_t o_{tp}^r + \sum_{(i_u, j_v) \in E^D} \rho_{i_u j_v} b_{i_u j_v p}^r + \sum_{j \in N^D} \sum_{v \in U_j} \gamma_j^v a_{jp}^{kv} - \delta^k \right\} \quad (34)$$

Similar to P^k, the set P^r is defined as the feasible region of a network flow problem. However, unlike the PSP^k, the PSP^r does not consider any resources in addition to cost, therefore the column generation subproblem is solved as a standard shortest path problem. The key features of the PSP^r is that each flight string must originate from a unique carry-in activity and may terminate at any permissible carry-out activity or overnight airport. In addition, if an aircraft is planned to receive maintenance at the end of the day, the termination locations will ensure that this requirement is met. In the column generation and column-and-row generation solution approaches the PSP^r is solve once for each $r \in R$ in each iteration.

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. As such, the nodes can be listed in a topological order and an efficient pulling algorithm presented in Ahuja *et al.* [4] is implemented to solve the PSP^r.

In a given 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 $\hat{c}_p^R = \min_{r \in R}\{\bar{c}_p^r\}$. However, all connection costs and dual variables, except for $\boldsymbol{\delta}^R = \{\delta^r, \forall r \in R\}$, included in (34) are aircraft independent. Therefore, by setting $\delta^r = \delta^R, \forall r \in R$, where $\delta^R = \max_{r \in R}\{\delta^r\}$ in (34), it is possible to find a lower bound on $\hat{c}_p^R$, labelled as $\bar{c}_p^R$, by solving the aircraft routing shortest path algorithm only once. The aircraft routing subproblem to be solved only once will be labelled PSP^R and will be used as part of the row generation procedure described in Section 4.2.

4.2 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 [34] and discrete flight copies [37], 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, equation (9), capture this, at the expense of adding a large number of constraints to the RMP_{IRP}. 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_{IRP}.

The implementation of delay copies in the IRP provides alternative 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 [41], Thengvall *et al.* [37] and Andersson and Varbrand [5], Bratu and Barnhart [11] state that a number of copies may be dominated by shorter delay options. Further, Petersen *et al.* [30] 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, significantly reducing the number delay consistency constraints (9). While it is possible to implement the recovery flight network reductions as described by [11] and [30], this would simply result in the further enhancement of the column-and-row generation approach.

Comparing the IRP with the RMP in Section 3.1.1, it is clear that the delay consistency constraints (9) describe linking constraints similar to (19). 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 3.1.1 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.

The implementation of the column-and-row generation algorithm, Algorithm 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 paper, the column-and-row generation solution approach developed by Muter *et al.* [26] 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.

4.2.1 Formulation of the restricted problems

The column-and-row generation framework requires the formulation of a RMP_{IRP} and SRMP_{IRP} as restrictions on the original problem. The formulation of the RMP_{IRP} is provided in Section 4.1, including only a subset of all variables. The SRMP_{IRP} is a further restriction on the RMP_{IRP}, initialised with all rows related to constraints (2)-(8) and only a subset of rows for the delay consistency constraints (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$_{IRP}$ is coupled with the fixing of variables in the column generation subproblems. This variable fixing restricts the use of specific flight-copy pairs related to the rows eliminated to form the SRMP$_{IRP}$. Thus, flight strings can only be constructed using the flight-copy pairs $j_v, \forall j \in N^D, \forall v \in \bar{U}_j$. Since the set of columns feasible for the SRMP$_{IRP}$ is restricted, the solution to the SRMP$_{IRP}$ provides an upper bound on the optimal solution of the IRP.

4.2.2 Row generation algorithm

The calculation of the optimal dual solution to the RMP$'$ is a fundamental part of the row generation procedure. By solving the SRMP$_{IRP}$ to optimality using column generation, Theorem 3.1.1 states that the optimal dual solution to the RMP$'$ can be calculated using the solution to the SRMP$_{IRP}$ and Algorithm 1. The dual solutions that must be calculated are related to the rows eliminated to form the SRMP$_{IRP}$, 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 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 uses the optimal dual solution to the RMP$'$ to identify favourable rows to add to the SRMP$_{IRP}$. This process is described by steps 8-11 of Algorithm 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 schedules for each crew k, hence a larger number of favourable rows are 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 achieve an efficient solution approach for the IRP.

The dual variable calculation of the row generation procedure is a feature of column-and-row generation that is identified to be very computationally expensive. Given the set of flight-copy pairs, $\cup_{j \in N^D} U_j \backslash \bar{U}_j$, the PSP^R must 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 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$_{IRP}$ too rapidly. A successful approach involves adding

more rows to the SRMP_{IRP} based upon the value of the calculated dual variables. This is achieved for the IRP by adding a row for every flight-copy pair with a positive dual variable value, as calculated by Algorithm 1. The rows identified by this procedure is in addition to those identified in Algorithm 2. The ideal number of rows to add at each iteration is difficult to determine. However, this approach described here improves the solution runtimes compared to the standard row generation procedure.

4.2.3 Row generation warm-up

The subset of rows initially included in the SRMP_{IRP} greatly affects the efficiency of the column-and-row generation solution process. In Section 4.2.1, the initialisation of the SRMP_{IRP} 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_{IRP} 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_{IRP} with all rows from the original problem but only the columns contained in the initial formulation of the SRMP_{IRP}. The RMP_{IRP} is then solved by column generation and in each iteration of the solution algorithm, flight strings are constructed with no restriction on the permissible flight-copy pairs. The initial set of delay copies for the SRMP_{IRP} 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_{IRP} is formed to contain only the delay consistency constraints (9) described by the sets $\bar{U}_j, \forall j \in N^D$ and all columns in the current formulation of the RMP_{IRP}.

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 use of the warm-up process is a contribution of this paper to the column-and-row generation solution approach.

The runtime improvements achieved by this approach demonstrate the importance of an intelligent selection of rows in the initial formulation of the SRMP_{IRP}. 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 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 $n = 20$

iterations.

4.3 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. 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 [32]. A description of each rule is provided below in the order of their assigned priority.

A cancellation variable branching rule is implemented for the IRP to force the decision of either covering or cancelling a specific flight. 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 branching on follow-on's. The implementation of follow-on branching for the IRP is similar to that presented in Froyland *et al.* [19]. The aim of this branching rule is to select the most fractional pair of connected flights, flights occurring in succession on a flight string, for either the crew duty or aircraft routing variables. In the implementation of this rule, the multiple copies for each flight are ignored 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 $P_f^k = \{p \in P^k | x_p^k \notin \mathbb{Z}\}$, and similarly the set of fractional variables for aircraft r is defined as, $P_f^r = \{p \in P^r | y_p^r \notin \mathbb{Z}\}$. The fractionality of a connection (i, j) is calculated by,

$$
\begin{aligned}
frac^K_{fOn}(i,j) &= \min\Big\{ \sum_{k\in K} \sum_{\substack{p\in P_f^k \\ |(i,j)\in p}} x_p^k, 1 - \sum_{k\in K} \sum_{\substack{p\in P_f^k \\ |(i,j)\in p}} x_p^k \Big\}, \quad \text{and} \\
frac^R_{fOn}(i,j) &= \min\Big\{ \sum_{r\in R} \sum_{\substack{p\in P_f^r \\ |(i,j)\in p}} y_p^r, 1 - \sum_{r\in R} \sum_{\substack{p\in P_f^r \\ |(i,j)\in p}} y_p^r \Big\},
\end{aligned}
\tag{35}
$$

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. The branching is performed by enforcing the use of connection (i^*, j^*) on a string that contains either flight i^* or j^* on the left branch for the identified variable type. On the right branch, flight j^* must not directly follow flight i^* on any string for the identified variable type. However, on the right branch, flights i^* and j^* are not precluded from existing on the same string, provided that the connections (i^*, k) and (l, j^*) are used, where $k \neq j^*$ and $l \neq i^*$.

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,

$$\begin{aligned} frac^K_{flt}(k, i_u) &= \min\Big\{ \sum_{\substack{p \in P^k_f \\ |i_u \in p}} x^k_p, 1 - \sum_{\substack{p \in P^k_f \\ |i_u \in p}} x^k_p \Big\}, \quad \text{and} \\ frac^R_{flt}(r, i_u) &= \min\Big\{ \sum_{\substack{p \in P^r_f \\ |i_u \in p}} y^r_p, 1 - \sum_{\substack{p \in P^r_f \\ |i_u \in p}} y^r_p \Big\}, \end{aligned} \tag{36}$$

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 (36). 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 approach, 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 potentially overestimate 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.

5 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 provides a comparison between 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 use of recovery policies. These results demonstrate column-and-row generation as a superior alternative to column generation for solving integrated airline optimisation problems for the given flight schedules.

5.1 Description of data and disruption scenarios

The performance of the IRP is evaluated on two different flight schedules that are designed for a point-to-point and hub-and-spoke carriers, labelled as F262-A20 and F441-A36 respectively. The point-to-point schedule consists of 262 flights, operated by 48 aircraft of a single fleet type and 79 crew groups. This schedule services 20 airports; 12 of the airports are overnight bases for aircraft and 4 are crew bases.

The hub-and-spoke schedule consists of 441 flights, operated by 123 aircraft of a single fleet type and 182 crew groups. This schedule services 36 airports; 6 of the airports are hubs, 32 are overnight bases for aircraft and 3 are crew bases. The original duties and routings for both schedules 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 each 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 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 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 3: Scenario numbers. (x,y) indicates scenario x has a closure of 3 hours and scenario y has a closure of 5 hours. 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 by only including those identified as disruptable. These results aim to demonstrate the runtime improvements achieved by implementing column-and-row generation alone. Hence, no approximation of the set of disruptable crew and aircraft is made. It is expected that with approximations of the dataset in practical applications of the algorithm, further runtime improvements will be observed. 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 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.

Scenario	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
F262-A20	150	151	149	150	147	145	150	149	182	183	185	186	184	182	184	183
F441-A36	265	261	246	254	251	251	250	251	315	311	304	310	305	305	292	293

Table 4: The number of disruptable flights for each scenario.

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 [14], 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 2. 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 (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 another flight operated by the airline, ii) rebooked onto another 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 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.

5.2 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 1 compares the runtime required to solve the IRP for the two different flight schedules 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. This maximum runtime is selected to be within the runtime of 30 minutes set by Petersen *et al.* [30] for their evaluation of an integrated airline recovery problem.

Figure 1 shows that in the vast majority of experiments, the optimal solution is found with runtimes much less than 1200 seconds. On average, column-and-row generation solves the IRP in 427 seconds for the F262-A20 schedule and 400 seconds for the F441-A32 schedule. In addition, the results presented in Figure 1 show that there is no scenario solved by column-and-row generation that exceeds the maximum allowable runtime. Therefore, it is possible to reduce the maximum allowable runtime without affecting the optimality of the IRP solved by column-and-row generation.

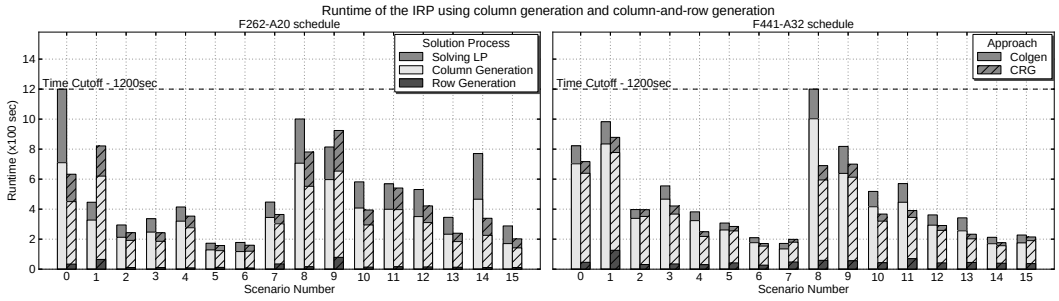

Figure 1: The runtime to solve the IRP for each scenario with a maximum of 1200 seconds. This figure compares the solution approaches of column generation (bars) and column-and-row generation (bars with hatching).

The results presented in Figure 1 demonstrate that the solution to the IRP using column-and-row generation is achieved much faster than when column generation is used for all but three cases. Two of the cases are for the F262-A20 schedule (scenarios 1 and 9) and the other is for the F441-A32 schedule (scenario 7). For the F262-A20 schedule, the average relative improvement between the two solution methods is 27.12%, with a range of -84.14% (scenario 1) to 127.13% (scenario 14). This level of improvement is also observed for the F441-A32 schedule, with an average improvement in the solution runtimes of 25.18%. However, the range of the results for the F441-A32 schedule is more restricted with a minimum improvement of -15.91% (scenario 7) and a maximum of 74.97% (scenario 8). These improvements demonstrate a significant benefit from 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_{IRP} compared to the RMP_{IRP}. 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 include a column generation process, which involves solving the LP relaxation of the RMP_{IRP}, and SRMP_{IRP}, each time a set of columns is added. Since the RMP_{IRP} and SRMP_{IRP} are formulated as very similar problems, with the latter containing less constraints, solving the LP for the SRMP_{IRP} requires significantly less simplex iterations resulting in faster execution times.

Figure 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 for the F262-A20 schedule. Using scenario 12 for the F262-A20 schedule as an example, solving the LP of the SRMP_{IRP} 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_{IRP}. 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 provides a significant improvement in a solution process which is integral to both approaches.

Another significant improvement in solution runtimes is observed in 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 and 0.6754 seconds quicker for column-and-row generation compared to column generation in the F262-A20 and F441-A32 schedule results respectively. The magnitude of this improvement can be explained using scenario 4 for the F262-A20 schedule 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 result is also observed in scenario 1 for the F441-A32 schedule, where the total execution time of the column generation subproblem is 57.37 seconds less with the column-and-row generation approach. The reduction in column generation subproblems execution times contributes 54.6% of the total runtime reduction. This reduction in runtimes for the column generation subproblems is achievable as a result of eliminating rows to form the SRMP_{IRP}. It is clear from Figure 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.

5.2.1 Effects of problem size

The number of rows initially removed from the RMP_{IRP} to form the SRMP_{IRP} 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_{IRP}

initially defines a problem much smaller than the related RMP_{IRP}, 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 2 displays the relative difference in runtimes between the column generation and column-and-row generation approaches by varying the number of flight copies for the F262-A20 schedule. The results demonstrate that across all experiments performed, column-and-row generation outperforms the column generation approach in most cases, with an average improvement of 27.07%.

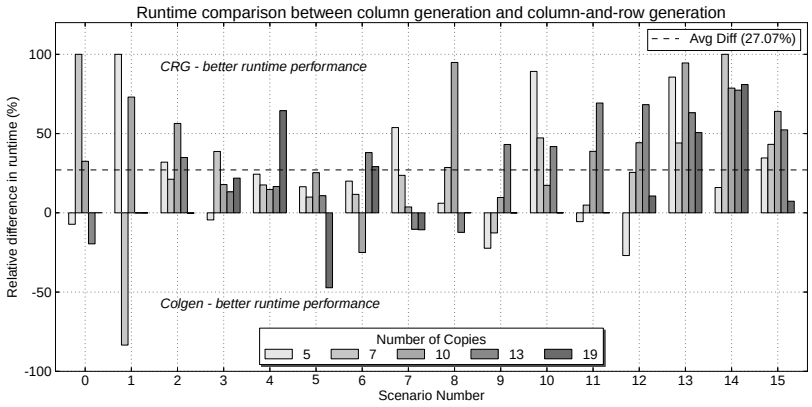

Figure 2: The relative difference in runtimes between the column generation (x) and column-and-row generation (y) approaches with different sets of copies for the F262-A20 schedule. The values in the figure are calculated by $(x - y)/\min\{x, y\}$ with a maximum reported improvement capped at 100%. Note: maximum runtime of 1 hour was used for these results.

Comparing the relative difference between the runtimes of column generation and column-and-row generation in Figure 2 demonstrates a better average performance of the latter approach. 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_{IRP}. The structure of the SRMP_{IRP} appears to affect the efficacy of the branching rules and the performance of the primal heuristics. This is a common observation regarding the implementation of column-and-row generation within the branch-and-price framework.

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

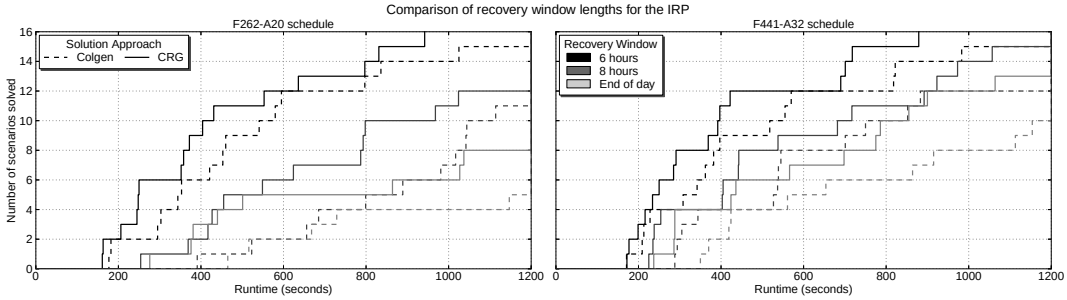

Figure 3: Time required to solve scenarios with different recovery window lengths (maximum runtime of 1200 seconds and 7 flight copies per flight). The recovery windows are set at 6 hours, 8 hours and until the end of the day.

flights N^D has two main effects on the IRP, i) an increase in the number of constraints in both the RMP_{IRP} and SRMP_{IRP}, and ii) an increase in the size of the connection network used in the column generation subproblem.

Figure 3 presents the time required to solve all of the scenarios used in the experiments for both flight schedules with different recovery window lengths, 6 hours, 8 hours and until the end of the day. The results presented for the F262-A20 schedule demonstrate that 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. This result is also observed for the F441-A32 schedule, however the separation of the frontiers is not as pronounced.

It is observed in Figure 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 for both flight schedules. 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.

Comparing the results presented in Figures 1 and 3, it is clear that the improvement in runtimes is more pronounced for the F262-A20 schedule. In addition, the number of scenarios solved for the F262-A20 schedule degrades much quicker with an increase in the recovery window length compared to the F441-A32 schedule. This is an interesting result which can be explained by the percentage increase in the number of included flights. By increasing the recovery window length from 6 hours to the end of the day, the average percentage increase in flights for the F262-A20 schedule is 38.45% compared to 27.13% for the F441-A32 schedule. In addition, the maximum percentage increase in the number of included flights for the F262-A20 schedule is 73.97% compared to 52.11% for the F441-A32 schedule. Since the increase in recovery window lengths results in a comparatively larger increase in the number of included flights, it is expected that a greater degradation of the solution runtimes will be observed

for the F262-A20 schedule.

5.2.2 Analysing enhancement techniques

A contribution of this paper is the explicit evaluation of the column-and-row generation approach against column generation and the development of enhancement techniques. While the techniques introduced in Section 4.2 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_{IRP} with a meaningful set of delay options. Using the F262-A20 schedule, Figure 4 presents the relative difference in solution runtimes for the column-and-row generation approach with different enhancements compared to the standard column generation approach.

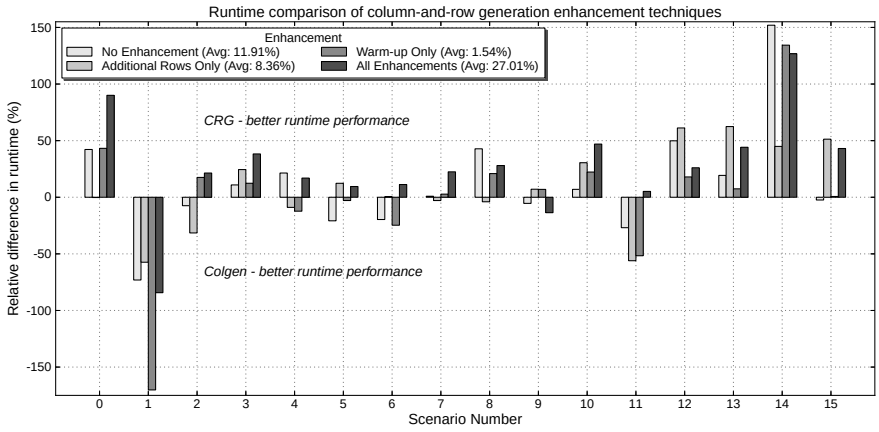

Figure 4: The relative difference in the solution runtimes from applying different enhancement techniques for the F262-A20 schedule. The values in the figure are calculated by $(x - y)/\min\{x, y\}$.

The strength of the column-and-row generation approach to improve upon solution runtimes of column generation is evident in Figure 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 can be explained by the nature of the two enhancement approaches. The row warm-up technique slows the execution of the initial iterations of the column generation stage by permitting the use of all flight-copies in the subproblem. This coupled with only a small set of rows added in each iteration of the row generation procedure does not provide much reduction in the solution runtimes. By constrast, the additional rows enhancement rapidly increases the size of the SRMP_{IRP}. As a result, very little decrease in the LP solving time is observed and on average more time is required per iteration to solve the column generation subproblem. Finally, the use of no enhancements achieves a smaller optimality gap

at the root node compared to the two individual enhancement approaches in most scenarios, requiring less nodes to find the integer optimal solution.

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 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 provides a runtime improvement of 15.88%. This is exhibits a significant runtime improvement from the use of enhancement techniques in the implementation of column-and-row generation.

The results presented 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 4, where 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 solving time for the root node, with scenario 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.

5.3 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. Figures 5 and 6 demonstrate the use of each of these recovery policies in the solutions achieved by column generation and column-and-row generation for the two considered flight schedules.

Since both the column generation and column-and-row generation solution approaches are solved to within an optimality gap of 1% for most scenarios, very little difference between the solutions is observed. This is demonstrated in Figures 5 and 6 with only subtle variations in the use of the reported recovery policies. The largest difference is observed for Scenario 8 in Figure 6, which is a consequence of column generation being unable to solve the IRP to integral optimality within the maximum runtime. It is also important to note that scenario 0 in Figure 5 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

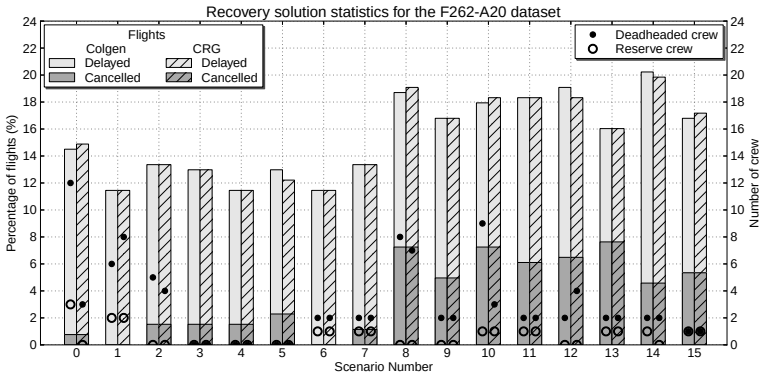

Figure 5: Comparison of recovery solution statistics from the column generation and column-and-row generation solutions using the F262-A20 schedule. The left axis is related to the flight statistics and the right axis is related to crew specific recovery policies.

recovered crew duties and aircraft routes.

The most significant difference in the solutions presented in Figure 5 is given by the number of deadheaded crew. The column generation solution requires more deadheading crew than the 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_{IRP} and RMP_{IRP}. Since the SRMP_{IRP} contains less rows than the RMP_{IRP} throughout the solution process, there are many columns that are initially not permissible in the column-and-row generation approach. This results in feasible integer solutions with minimal delay, hence requiring the use of alternative recovery policies, being added to the solution pool in the column-and-row generation approach. 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.

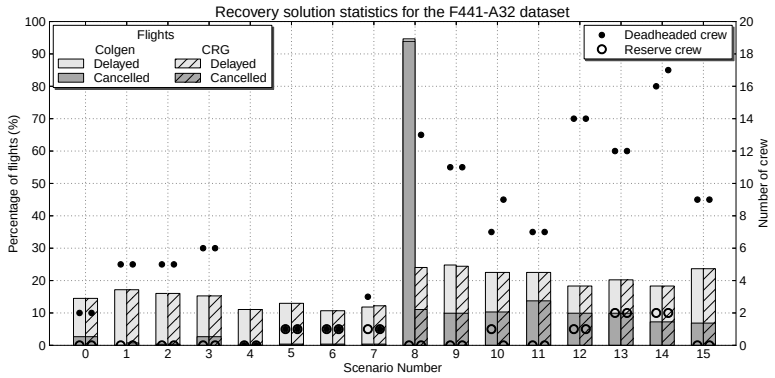

Figure 6: Comparison of recovery solution statistics from the column generation and column-and-row generation solutions using the F441-A32 schedule. The left axis is related to the flight statistics and the right axis is related to crew specific recovery policies.

Figure 6 greatly emphasises the similarities between the solutions achieved by column generation and

column-and-row generation. This little difference is the result of the IRP being solved to a 0% optimality gap at the root node in the majority of experiments for the F441-A32 schedule. Both column generation and column-and-row generation are exact solution approaches, so the results presented in Figure 6 are not surprising. The most interesting result is in Scenario 8, where the recovery statistics for the column generation result exhibits a large percentage of cancelled flights. Since column-and-row generation solves the IRP faster than standard column generation on average, the former solution approach is expected to provide a better solution quality in shorter runtimes.

Finally, reserve crew are a valuable resource for an airline, however there is a limitation to their use across a full scheduling period. It is important to manage the number of reserve crew that are employed to operate a disrupted schedule to ensure their availability for subsequent days. Figures 5 and 6 demonstrate that the modelling approach used for the IRP results in the very little use of reserve crew during recovery. This is an important result that allows the practical management of the reserve crew resource. Additionally, it is trivial to modify the model to limit the number of reserve crew that are available for each disruption.

6 Conclusions

This paper presents an alternative approach to solving the integrated airline recovery problem by implementing column-and-row generation. The integrated recovery problem is a very large and computationally difficult problem for which there have been many attempts to develop efficient solution methods. A general framework for column-and-row generation has been developed as an alternative exact solution method to Benders' decomposition and approximation techniques. The use of column-and-row generation to solve the IRP has been demonstrated here as a superior approach in regards to solution runtime and quality compared to column generation.

The column-and-row generation framework developed in this paper is an extension of Muter *et al.* [26]. The extensions of this framework are the consideration of multiple secondary variables and linking constraints, providing a wider applicability of the solution approach. Further extending the analysis of the framework by Muter *et al.* [26], an explicit evaluation of column-and-row generation against a standard column generation approach is performed. As a result of this evaluation, a number of enhancement techniques have been developed that may be applied to general column-and-row generation implementations. 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. The evaluation performed in this paper demonstrates the ability of column-and-row generation to improve upon the solution runtimes achieved by column generation. Finally, this paper details the integration of column-and-row generation and branch-and-price, which has not previously been published.

A motivation for applying column-and-row generation to solve the IRP is to reduce solution runtimes. The results in Section 5 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.

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.

There are two directions for future work on the integrated recovery problem solved using column-and-row generation. Firstly, the integrated recovery problem presented here integrates the schedule, aircraft and crew recovery problems, however more detailed solutions for the complete recovery problem can be achieved through further integration. The integration of passenger considerations in the IRP will aid in reducing the impact of recovery on passengers. Secondly, while the use of column-and-row generation improves upon the solution quality and runtime when compared to the standard column generation approach, it is likely that additional enhancements can ameliorate these even further. Research into the selection of rows during the row generation procedure is expected to aid in reducing the runtime for the column-and-row generation solution approach.

Acknowledgements

SJM would like to thank Gary Froyland for discussing concepts presented in this paper and providing insight to the problem.

References

[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] R. Ahuja, T. Magnanti, and J. Orlin. *Network flows: theory, algorithms, and applications*. Prentice Hall, 1993.

[5] T. Andersson and P. Varbrand. The flight perturbation problem. *Transportation Planning and Technology*, 27(2):91–117, 2004.

[6] 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.

[7] 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.

[8] J. Bard, G. Yu, and M. Argüello. Optimizing aircraft routings in response to groundings and delays. *IIE Transactions*, 33(10):931–947, 2001.

[9] 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.

[10] 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.

[11] S. Bratu and C. Barnhart. Flight operations recovery: new approaches considering passenger recovery. *Journal of Scheduling*, 9(3):279–298, 2006.

[12] 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.

[13] 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.

[14] A. Cook and G. Tanner. European airline delay cost reference values. *URL: http://www.eurocontrol.int/documents/european-airline-delay-cost-reference-values*, 2011.

[15] 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.

[16] N. Eggenberg, M. Bierlaire, and M. Salani. A column generation algorithm for disrupted airline schedules. Technical report, Ecole Polytechnique Federale de Lausanne, 2007.

[17] A. Frangioni and B. Gendron. 0-1 reformulations of the multicommodity capacitated network design problem. *Discrete Applied Mathematics*, 157(6):1229–1241, 2009.

[18] A. Frangioni and B. Gendron. A stabilized structured Dantzig-Wolfe decomposition method. *Mathematical Programming*, 140(1):45–76, 2013.

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

[20] 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.

[21] 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.

[22] L. Lettovsky. *Airline operations recovery: an optimization approach*. PhD thesis, Georgia Institute of Technology, 1997.

[23] L. Lettovsky, E. Johnson, and G. Nemhauser. Airline crew recovery. *Transportation Science*, 34(4):337–348, 2000.

[24] C. P. Medard and N. Sawhney. Airline crew scheduling from planning to operations. *European Journal of Operational Research*, 183(3):1013–1027, 2007.

[25] 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.

[26] 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.

[27] 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.

[28] R. Nissen and K. Haase. Duty-period-based network model for crew rescheduling in European airlines. *Journal of Scheduling*, 9(3):255–278, 2006.

[29] N. Papadakos. Integrated airline scheduling. *Computers & Operations Research*, 36(1):176–195, 2009.

[30] 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.

[31] J. M. Rosenberger, E. L. Johnson, and G. L. Nemhauser. Rerouting aircraft for airline recovery. *Transportation Science*, 37(4):408–421, 2003.

[32] 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.

[33] R. Sadykov and F. Vanderbeck. Column generation for extended formulations. *EURO Journal on Computational Optimization*, 1(1-2):81–115, 2013.

[34] M. Stojković and F. Soumis. An optimization model for the simultaneous operational flight and pilot scheduling problem. *Management Science*, 47(9):1290–1305, 2001.

[35] M. Stojković and F. Soumis. The operational flight and multi-crew scheduling problem. *Yugoslav Journal of Operations Research*, 15(1):25–48, 2005.

[36] M. Stojković, F. Soumis, and J. Desrosiers. The operational airline crew scheduling problem. *Transportation Science*, 32(3):232–245, 1998.

[37] 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.

[38] G. Wang and L. Tang. A row-and-column generation method to a batch machine scheduling problem. In *Proceedings of the Ninth International Symposium on Operations Research and Its Applications (ISORA-10)*, pages 301–308, 2010.

[39] 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.

[40] S. Yan and D.-H. Yang. A decision support framework for handling schedule perturbation. *Transportation Research Part B: Methodological*, 30(6):405–419, 1996.

[41] 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.

[42] E. J. Zak. Row and column generation technique for a multistage cutting stock problem. *Computers & Operations Research*, 29(9):1143–1156, 2002.