Proceedings of the 2018 Winter Simulation Conference
M. Rabe, A. A. Juan, N. Mustafee, A. Skoogh, S. Jain, and B. Johansson, eds.

MULTI-FIDELITY SIMULATION OPTIMISATION FOR AIRLINE DISRUPTION MANAGEMENT

Luke Rhodes-Leader
David J. Worthington
Barry L. Nelson

STOR-i CDT
Lancaster University
Lancaster, LA1 4YR, UK

Bhakti Stephan Onggo

Trinity Business School
Trinity College Dublin
Dublin 2, IRELAND

ABSTRACT

The airline industry faces many causes of disruption. To minimise financial and reputational impact, the airline must adapt its schedules. Due to the complexity of the environment, simulation is a natural modelling approach. However, the large solution space, time constraints and system constraints make the search for revised schedules difficult. This paper presents a method for the aircraft recovery problem that uses multi-fidelity modelling including a trust region simulation optimisation algorithm to mitigate the computational costs of using high-fidelity simulations with its benefits for providing good estimates of the true performance.

1 INTRODUCTION

Disruption to schedules is one of the major issues faced by the airline industry. Airline scheduling itself has seen much focus at the planning stage, allowing for ever more robust and efficient use of the resources available to airlines. A great effort in time and money is spent on producing an optimal schedule. However, due to the complex and stochastic nature of the industry, it is rare that a flight programme is operated as intended. Disruptive events arise due to various circumstances, from weather conditions to aircraft technical failures requiring unplanned maintenance. The impacts of such events can propagate through the system causing further delays and cancellations, particularly when the airline has high aircraft utilisation. In such an eventuality, the Airline Operations Control Centre (AOCC) must alter its schedule to reduce the repercussions for its finances, its reputation and passengers. Such alterations could include delaying or cancelling flights or exchanging aircraft. This paper will focus on the Aircraft Recovery Problem (ARP).

The complexity of the industry can make it difficult to determine what the consequences of any schedule alterations may be. Kohl et al. (2007) suggest that automation could play a large part in the identification and evaluation of potential recovery actions. Some of the complexity has been considered using deterministic models, a number of which have been proposed. However, these models have a limited capability of accounting for the various stochastic elements of the environment. Turn times, airport queueing times, maintenance times and flight durations are all stochastic, which may lead to further disruptions that the AOCC would wish to consider. Probabilistic models that achieve the levels of detail required are unlikely to be analytically tractable. Thus simulation seems to be a natural way to model the airline's operations. However, a high-fidelity simulation model would have a non-negligible computation time. This creates issues for searching through the large solution space. To make the most of the simulation, it must be used selectively on solutions believed to be relevant.

The contribution of this paper is a method to balance the need to use a high-fidelity simulation model to estimate the performance of a recovery option with the computational difficulties associated with simulation

optimisation in this context. These difficulties include a large and complicated solution space and short computation time constraints. Our use of a low-fidelity model reduces the complexity of the solution space and speeds up the search by dealing with the complex constraints of aircraft allocation in a deterministic manner. It then provides the simulation optimisation with a number of good starting solutions and a structure with which to continue the search of its larger solution space. This goes beyond simply testing robustness, and achieves a much more selective use of the simulation than would otherwise be possible.

This paper is organised as follows. In Section 2, we discuss some of the work in airline disruption management. Section 3 describes a small, realistic example, introduces the problem formulation and gives a brief overview of the integer programming (IP) and simulation models used. The optimisation of both models in combination is presented in Sections 4 and 5. Following this, results from the example problem will be discussed in Section 6 and conclusions will be in Section 7.

2 RELATED WORK

There have been many and varied deterministic models and solution methods proposed in the literature for the ARP. Rosenberger et al. (2003) used a problem reduction heuristic combined with a set-packing formulation on the potential aircraft routes. Løve et al. (2005) proposed a Steepest Ascent Local Search heuristic to solve their network model in which aircraft allocations are represented by arcs. Zhu et al. (2015) began to incorporate the inherent uncertainty by using scenarios of times at which aircraft requiring unplanned maintenance become available and solving a stochastic programming problem. The weakness of such an approach is that if it were to be extended to include scenarios with other uncertain elements, the size of the problem would explode and become computationally intractable within the time constraints.

More recent work has focussed upon integrating different aspects of the recovery problem together, such as aircraft and crew as in Zhang et al. (2015). This is an important area of research, particularly as legal constraints on one resource can make an optimal solution for another infeasible in the integrated problem. Whilst integration should lead to improved decisions, the current practise of airlines generally takes a sequential approach with aircraft first. Thus, this paper will focus solely on the aircraft problem.

Simulation has been used to consider disruption handling. Rosenberger et al. (2000) developed software called SimAir to evaluate disruption handling policies. Deterministic functions were applied to identify the best action from a set of heuristics in a given disruption (Rosenberger et al. 2002). Lee et al. (2003) used SimAir to explore the robustness of schedules and recovery policies to disruption. Hutchison and Hill (2001) used a Simultaneous Perturbation Stochastic Approximation in a simulation optimisation process to reduce air delays across a network of airports by adding gate holdings where necessary. Abdelghany et al. (2008) used a deterministic network simulation to predict which flights will be disrupted to reduce the problem size of an IP which is solved on a rolling horizon during the recovery period.

Arias et al. (2013) used a constraint programming approach combined with a simple simulation to account for the stochastic elements of turn times and flight durations. The simulation is used to evaluate the constraint program solution under a variety of scenarios. If the solution is not considered robust across these replications, it is rejected and the process begins again. Guimarans et al. (2015) expanded this approach in two ways. The first was combining the constraint program with a Large Neighbourhood Search Heuristic to propose new solutions. The second was in using the simulation at each proposed solution to evaluate the acceptance criteria (known as *SimLNS*). However, the simulation remains rudimentary, and using a higher fidelity model may harm the performance of the algorithm if the simulation was used at every iteration.

Multi-fidelity modelling makes use of simple approximations to a problem to identify candidate 'best' solutions which can then be simulated more thoroughly. This is prevalent in "simheuristics" which often use deterministic models to search the solution space for promising solutions, *SimLNS* and the stochastic combinatorial problem approach of Juan et al. (2015) being examples. However, Xu et al. (2016) point out that a low-fidelity model may be a poor performance predictor due to unknown bias and criticise this naive approach of only simulating the best. Their proposed method (MO^{2}TOS) continues to sample from all solutions, weighted by their low-fidelity rankings, thus potentially exploring the whole space. Unfortunately,

this work assumes that all options are exhaustively searched and evaluated with the low-fidelity model and the output has no noise. Neither of these assumptions are valid here.

3 MULTI-FIDELITY MODELS

The following example is based on a flight schedule extracted from an open source data set (Flightradar24 AB 2017) obtained using the python package pyflightdata (Allamraju 2017). It consists of a homogeneous fleet of 8 aircraft with no spare aircraft operating over a hub-and-spoke network of 15 airports in Western Europe. The hubs of the airline are Manchester and Birmingham Airports (U.K.), where all aircraft are meant to spend the night. At the beginning of the day, it is discovered that one aircraft at Birmingham Airport, $A1$, will not be fit to fly its first flight. The expected ready time is 3 hours after that, at 8:50. The AOCC must now reschedule the next day of operations, involving 54 short-haul flights using the available aircraft. Its options include delaying flights, cancelling flights and exchanging aircraft.

Let A be the set of aircraft involved and F be the set of flights in the programme, with $n = |F|$ being the number of flights. Each flight $f \in F$ will require an aircraft, $a \in A$, and a planned delay time, d^f, or a cancellation. Let $\mathbf{x}$ be the aircraft allocation and $\mathbf{d} = (d^f : f \in F)$ be the vector of planned delays. The AOCC has the multi-objective aim of rescheduling to minimise its costs, delays and the number of alterations to the original schedule. Suppose that the cost of delaying is a linear function with a cost α per minute and that there is a further penalty of β per minute if your actual delay exceeds d^f. Passenger compensation for significant delays of flight f is given by a function of the delay, P^f (for example, see Civil Aviation Authority (2015)). The cancellation cost of flight f is C^f.

3.1 Low Fidelity Integer Program

The low-fidelity model used in this paper is an IP adapted from the aircraft recovery model of Zhang et al. (2015). The primary simplification in the model is the removal of all stochastic elements. The model aims to allocate aircraft to flights at a minimum cost and allows delays, re-assigning aircraft and cancellations. The model uses a time-space network in which each node, $v \in N$, represents an airport at a potential arrival or departure time. The potential delay times are discretised in steps m from the original departure time. A smaller step size leads to better solutions due to a less constrained problem, but also increases the problem size substantially.

Each potential delay of flight $f \in F$ is represented by a flight delay arc f_δ, where δ belongs to the set of potential delay options, $\{0, m, 2m, ..., M\}$, where M is the maximum allowable delay. The set of flight delay arcs is L, while the set of flight delay arcs associated with flight f is L^f. Between flight delay arcs, aircraft take ground arcs, $\gamma \in G$, which connect nodes at the same airport. Each γ must be of length at least $t_{\min}$, ensuring that an aircraft has sufficient turn time between flights. A larger choice for $t_{\min}$ is a higher quantile of the turn time distribution, leading to more robust solutions. Let $o_a^f \in \{0,1\}$ indicate whether aircraft $a \in A$ was assigned to flight f before the disruption occurred.

Each aircraft a has an input node, $i(a) \in N_I$, representing its current location. No flight delay arcs exit this node and the ground arcs only connect to nodes at which a is available. In our example, the earliest node that $i(A1)$ can connect to would be (Birmingham, 8:50). Furthermore, all aircraft will end at a return node, $r \in N_R$. Some aircraft may have a specified return node, denoted $r(a)$, for example an aircraft may be required at an airport for its scheduled maintenance. In this case, a constraint is imposed to make the new schedule maintenance feasible. An important part of a recovery is to ensure that the schedule beyond the end of the recovery window is feasible with little or no disruption. In our example, each airport will require sufficiently many aircraft at the end of the day to fly tomorrow's schedule. This is known as aircraft balance, with r_A being the number of aircraft required by return node r.

To consider the continuous flow of aircraft through the network, we define the following sets. Let L_{in}^v and G_{in}^v be the sets of flight delay arcs and ground arcs incident on $v \in N$, and L_{out}^v and G_{out}^v be the sets of flight delay arcs and ground arcs exiting $v \in N$. An airline may only be allowed to use an airport runway

during particular time slots. We list these slot constraints by the set S. Let K_s be the maximum number of aircraft movements allowed by the airline in slot s and L_s be all flight delay arcs impacting slot s. This mechanism can also be used to deal with curfew times at airports, for example, to prevent an aircraft taking off during the night. It is assumed that the costs of delaying flight f by δ minutes is $c^{f\delta} = \alpha\delta + P^f(\delta)$.

The decision variables for the IP are all binary variables. Let $x_a^{f\delta} = 1$ when aircraft a is assigned to flight delay arc f_δ, $y^f = 1$ if flight f is cancelled, and $z_a^\gamma = 1$ if aircraft a uses ground arc γ. The complete formulation follows.

$$\min \quad \sum_{a\in A}\sum_{f_\delta\in L} c^{f\delta}x_a^{f\delta} + \sum_{f\in F} C^f y^f \tag{1}$$

$$\min \quad \sum_{a\in A}\sum_{f_\delta\in L} \delta x_a^{f\delta} \tag{2}$$

$$\min \quad \sum_{a\in A}\sum_{f_\delta\in L} (1-o_a^f)x_a^{f\delta} \tag{3}$$

subject to

$$\sum_{a\in A}\sum_{f_\delta\in L^f} x_a^{f\delta} + y^f = 1 \qquad \forall f \in F \tag{4}$$

$$\sum_{a\in A}\sum_{f_\delta\in L_s} x_a^{f\delta} \le K_s \qquad \forall s \in S \tag{5}$$

$$\sum_{f_\delta\in L_{\text{in}}^v} x_a^{f\delta} - \sum_{\gamma\in G_{\text{out}}^v} z_a^\gamma = 0 \qquad \forall a \in A, \forall v \in N\backslash(N_I \cup N_R) \tag{6}$$

$$\sum_{\gamma\in G_{\text{in}}^v} z_a^\gamma - \sum_{f_\delta\in L_{\text{out}}^v} x_a^{f\delta} = 0 \qquad \forall a \in A, \forall v \in N\backslash(N_I \cup N_R) \tag{7}$$

$$\sum_{\gamma\in G_{\text{out}}^i} z_a^\gamma = 1_{\{i=i(a)\}} \qquad \forall a \in A, \forall i \in N_I \tag{8}$$

$$\sum_{f_\delta\in L_{\text{in}}^{r(a)}} x_a^{f\delta} + \sum_{\gamma\in G_{\text{in}}^{r(a)}} z_a^\gamma = 1 \qquad \forall a \in \{a \in A : r(a) \in N_R\} \tag{9}$$

$$\sum_{a\in A}\sum_{f_\delta\in L_{\text{in}}^r} x_a^{f\delta} + \sum_{a\in A}\sum_{\gamma\in G_{\text{in}}^r} z_a^\gamma \ge r_A \qquad \forall r \in N_R \tag{10}$$

$$x_a^{f\delta}, z_a^\gamma, y^f \in \{0,1\} \qquad \forall a \in A, f \in F, f_\delta \in L, \gamma \in G \tag{11}$$

Objective (1) is the cost of the recovery action, (2) is the total planned delay and (3) is the number of changes made to the aircraft allocation. Constraint (4) ensures that each flight is either flown once or cancelled. Constraint (5) are the slot constraints. Constraints (6) and (7) are flow constraints for normal nodes, whilst (8) and (9) are analogous for input nodes and return nodes when specified, respectively. Constraint (10) ensures aircraft balance at the end of the recovery period. Additional constraints and costs could extend this model to, for example, penalise inconvenient solutions from a crewing perspective.

3.2 High Fidelity Simulation

The input schedule for the simulation is given by an aircraft allocation $\mathbf{x}$ and a set of planned delays, $\mathbf{d}$. The elements of $\mathbf{x}$ and $\mathbf{d}$ are linked to the IP variables via the following relationships:

$$x_a^f = \sum_{f_\delta\in L^f} x_a^{f\delta}, \tag{12}$$

$$d^f = \sum_{a\in A}\sum_{f_\delta\in L^f} \delta x_a^{f\delta}, \tag{13}$$

whilst a cancellation of flight f can be inferred when $\sum_{a\in A} x_a^f = 0$.

Let $D^f \sim H(\mathbf{x},\mathbf{d})$ be the random variable stating the true delay of flight f. The objective function is

$$g(\mathbf{x},\mathbf{d}) = \mathbb{E}\left[\sum_{f\in F}\left(\alpha D^f + \beta(D^f - d^f)^+ + P^f(D^f)\right)\right] + C(\mathbf{x}). \tag{14}$$

Here, $P^f(D^f)$ represents the compensation associated with passenger delays of flight f and $C(\mathbf{x})$ is the cost of the cancellations in $\mathbf{x}$.

The simulation model is built within AnyLogic 8.2.3 (The AnyLogic Company 2017) and described in Rhodes-Leader et al. (2018). It simulates a sub-fleet of homogeneous aircraft operating the recovery action $(\mathbf{x},\mathbf{d})$ over a set of airports. Each aircraft follows its assignment of the schedule, subject to stochastic flight durations, turn times, queueing times and maintenance. Its general framework is largely based on the SimAir simulation (Lee et al. 2003). However, it does not consider crew members or passengers.

The flight durations for each flight are modelled using the available data. The turn times are assumed to follow a truncated normal distribution. The mean and variance can vary between airports, whilst the minimum could be taken from minimum turn times of the aircraft. These assumptions are made as our current data source does not track this information. The gate departure time of the aircraft is the maximum of the ready time (after the turn time) and the planned departure time according to the input schedule $(\mathbf{x},\mathbf{d})$. When this time occurs, the aircraft joins the take-off queue.

The arrival process to the landing and take-off queues is a deviation-from-schedule model based on the published arrivals and departures schedule and the deviation distributions are estimated from the observed data for the airport. The service time represents the spacing required between aircraft and is assumed deterministic given the weather conditions and time of day. The weather conditions follow a step-function forecast at each airport, with poor weather conditions leading to increased aircraft spacing. Similarly, the aircraft separation may increase late at night or early morning due to noise pollution constraints. It is assumed that each airport has one runway for landings and one runway for departures and that the two queues operate independently.

Each aircraft is given a time to failure based on its flying time since its last maintenance. We only consider faults that would lead to an aircraft being grounded and assume that the time to failure is Weibull distributed. Once an aircraft's flying time has exceeded this time to failure, it enters the maintenance hangar at the next airport where it lands. All unplanned maintenance is assumed to come from a Gamma distribution and must be completed before the next flight. The parameters vary from airport to airport depending on the facilities available. For example, the expected time for unplanned maintenance at Birmingham Airport (a hub of the airline) should be shorter than that of Stuttgart Airport. Scheduled maintenance is also part of the schedule and is assumed to take a fixed amount of time.

4 SOLVING THE INTEGER PROGRAMMING MODEL

The aim of the IP is to generate a set of promising solutions to be used as starting points in the simulation optimisation process. This will result in a set of fixed aircraft allocations. As the overall problem has multiple objectives, the $\varepsilon-$constraint method is used to produce the solutions. The objectives of minimising total delay (2) and minimising the number of aircraft reassignments (3) are added to the IP as constraints:

$$\sum_{a\in A}\sum_{f_\delta\in L}\delta x_{f_\delta}^a \in [\varepsilon_l^d, \varepsilon_u^d]$$
$$\sum_{a\in A}\sum_{f_\delta\in L}(1-o_a^f)x_{f_\delta}^a \in [\varepsilon_l^e, \varepsilon_u^e].$$

The parameters $(\varepsilon_l^d, \varepsilon_u^d, \varepsilon_l^e, \varepsilon_u^e)$ are varied using an efficient method proposed by Laumanns et al. (2006) to search for solutions on the efficient frontier. Initially, the IP is solved without (2) and (3) constraining the

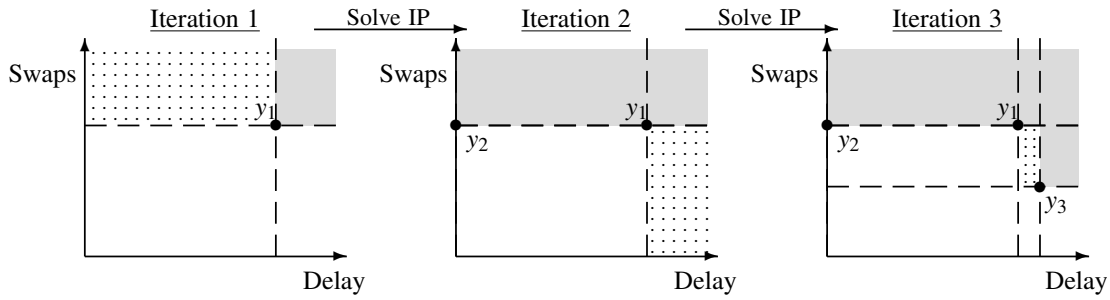

Figure 1: An example of the ε-Constraint approach used. Each plot shows the result of one iteration, each time generating a new solution, y_1, y_2 and y_3. Shaded areas are considered searched, dotted areas indicate the next region to be searched. Dashed lines show the region divisions.

problem. Due to the nature of the quantities, this is equivalent to defining $\varepsilon_l^d = \varepsilon_l^e = 0$ (that is, no delays and no exchanges) and $\varepsilon_l^d = nM$ (all flights delayed by the maximum amount) and $\varepsilon_l^e = n$ (no flights operated using their original aircraft). Once the initial solution is found, the values of (2) and (3), $(\varepsilon_1^d, \varepsilon_1^e)$ are used to split the objective space into four sectors: $(0,\varepsilon_1^d) \times (0,\varepsilon_1^e)$, $(0,\varepsilon_1^d) \times (\varepsilon_1^e, n)$, $(\varepsilon_1^d, nM) \times (0,\varepsilon_1^e)$ and $(\varepsilon_1^d, nM) \times (\varepsilon_1^e, n)$. The first solution is considered the best in the region $(\varepsilon_1^d, nM) \times (\varepsilon_1^e, n)$ and so this region is considered searched. The algorithm now chooses one of the not searched sectors to define $(\varepsilon_l^d, \varepsilon_u^d, \varepsilon_l^e, \varepsilon_u^e)$. If this new region provides a solution, $(\varepsilon_2^d, \varepsilon_u^d) \times (\varepsilon_2^e, \varepsilon_u^e)$ is added to the searched regions. If $(\varepsilon_l^d, \varepsilon_u^d, \varepsilon_l^e, \varepsilon_u^e)$ leads to an infeasible IP, $(\varepsilon_l^d, \varepsilon_u^d) \times (\varepsilon_l^e, \varepsilon_u^e)$ is considered searched. This process is repeated, splitting the sectors further and blocking off searched regions. Given sufficient time, this process will eventually search the whole space and find all solutions on the efficient frontier. In our application, however, this process stops after a time limit. An example of this process can be seen in Figure 1. For each combination of limits, the IP is solved to optimality using the Gurobi Optimizer 7.0.2 (Gurobi Optimization 2017).

The result is a set of solutions, $\mathscr{X}$, each with an aircraft allocation, a set of cancelled flights, and a delay value for all flights in the programme.

5 SIMULATION OPTIMISATION AROUND IP SOLUTIONS

Each revised schedule $(\mathbf{x}, \mathbf{d}_0) \in \mathscr{X}$ is used as the starting point for a simulation optimisation process, searching for improvement around this solution. To reduce the complexity of the solution space, the flight allocation $\mathbf{x}$ is fixed during the optimisation. This leaves a continuous, ordered solution space of planned delays, $\mathbf{d} \in \mathscr{D} \subset \mathbb{R}^n$, with simple bound constraints on each variable. For further reduction, only the n' flights with non-zero delay variables in $\mathbf{d}_0$, $\{d^f : f \in F^+\}$, are used in the optimisation. The solution space is now greatly simplified from the original. This gives the simulation optimisation problem:

$$\min_{\mathbf{d}} \; g(\mathbf{x}, \mathbf{d}) \tag{15}$$

$$\text{subect to} \quad d^f \in [l^f, u^f], \qquad \forall f \in F^+ \tag{16}$$

$$d^f = 0, \qquad \forall f \notin F^+ \tag{17}$$

where g is defined in (14), searching for a local optimum conditional on $\mathbf{x}$. For our example, we take $l^f = 0$ and $u^f = \infty$ for all $f \in F$. We estimate the performance of $(\mathbf{x}, \mathbf{d})$ by the mean of multiple replications, denoted $\hat{g}(\mathbf{x}, \mathbf{d})$.

To solve the problem, an adapted version of the STRONG algorithm (Chang et al. 2013) is applied to find a local improvement starting at $(\mathbf{x}, \mathbf{d}_0)$. STRONG is a simulation optimisation algorithm based on the ideas of trust region optimisation. At iteration j, a linear or quadratic response surface model, $r_j(\mathbf{d}_j + \mathbf{p})$, is built on a hyper-sphere of radius Δ_j (the trust region) centred at the current solution, $\mathbf{d}_j$, where $\mathbf{p}$ is a deviation from $\mathbf{d}_j$. This meta-model can be built using 2^k-factorial designs and Central Composite

Designs (CCD) for the linear and quadratic versions respectively with n_d replications of each design point, as suggested by Chang et al. (2013). The quadratic model is chosen whenever the trust region becomes sufficiently small, $\Delta_j < \tilde{\Delta}$. This creates a trust region sub-problem:

$$\begin{aligned} \min_{\mathbf{p}} \quad & r_j(\mathbf{d}_j + \mathbf{p}) \\ \text{subject to} \quad & ||\mathbf{p}|| \leq \Delta_j. \end{aligned}$$

The Cauchy Point, $\mathbf{p}_c$, (the point that minimises r_j along the direction $-\nabla r_j(\mathbf{d}_j)$ within the feasible trust region) of the meta-model is either accepted or rejected based on a sufficient reduction hypothesis test and a ratio comparison:

$$\frac{\hat{g}(\mathbf{x}, \mathbf{d}_j) - \hat{g}(\mathbf{x}, \mathbf{d}_j + \mathbf{p}_c)}{r_j(\mathbf{d}_j) - r_j(\mathbf{d}_j + \mathbf{p}_c)} \geq \eta > 0 \tag{18}$$

comparing the predicted reduction in g with the simulated reduction in g . The size of the trust region, Δ_j, is altered depending on the outcome of the tests. Chang et al. (2013) showed that under some technical assumptions this algorithm asymptotically converges to a local optimal with probability 1 for unconstrained simulation optimisation.

The simple bound constraints, (16), of the ARP, such as delay being non-negative, mean that a different strategy is required near the boundary to retain feasibility without significantly slowing the optimisation down. Instead of the hyper-sphere constraint on the trust region sub-problem, an ellipsoidal constraint is used, and the objective function modified by an adjustment to the Hessian:

$$\min_{\mathbf{p}} \quad r_j(\mathbf{d}_j + \mathbf{p}) + \frac{1}{2}\mathbf{p}^T C_j \mathbf{p} \tag{19}$$

$$\text{subject to} \quad ||D_j \mathbf{p}|| \leq \Delta_j \tag{20}$$

$$\mathbf{d}_j + \mathbf{p} \in \mathscr{D}. \tag{21}$$

This trust region method for simple bound constrained-problems was proposed by Coleman and Li (1996). To motivate this method, consider the first-order optimality conditions for simple bounded optimisation:

$$\begin{aligned} &\nabla g(\mathbf{d})^f = 0 \quad \text{if } d^f \in (l^f, u^f), \\ &\nabla g(\mathbf{d})^f \geq 0 \quad \text{if } d^f = l^f, \\ &\nabla g(\mathbf{d})^f \leq 0 \quad \text{if } d^f = u^f. \end{aligned} \tag{22}$$

If we define a diagonal scaling matrix, $D(\mathbf{d})$, by $(D(\mathbf{d}))^{ff} = (|v^f(\mathbf{d})|)^{-1/2}$, where

$$v^f(\mathbf{d}) = \begin{cases} d^f - u^f & \text{if } \nabla g(\mathbf{d})^f < 0 \text{ and } u^f < \infty \\ d^f - l^f & \text{if } \nabla g(\mathbf{d})^f \geq 0 \text{ and } l^f > -\infty \\ -1 & \text{if } \nabla g(\mathbf{d})^f < 0 \text{ and } u^f = \infty \\ 1 & \text{if } \nabla g(\mathbf{d})^f \geq 0 \text{ and } l^f = -\infty \end{cases}$$

then (22) is equivalent to

$$D(\mathbf{d})^{-2} \nabla g(\mathbf{d}) = \mathbf{0}. \tag{23}$$

We estimate $D_j = D(\mathbf{d}_j)$ by using the approximation $\nabla g(\mathbf{d}_j) \approx \nabla r_{j-1}(\mathbf{d}_j)$. If one takes a Newton step based on (23) the Hessian term has an adjustment given by $C_j = D_j J_j \text{diag}(\nabla r_{j-1}(\mathbf{d}_j)) D_j$ where J_j is the Jacobian matrix of the vector $v(\mathbf{d}_j)$ and $\text{diag}(\mathbf{y})$ is a diagonal matrix with elements of the vector $\mathbf{y}$. The adjustment aims to prevent movement directly towards the boundary. Any proposed steps that lead to infeasible solutions are truncated to retain strict feasibility. The acceptance condition (18) is modified to

$$\frac{\hat{g}(\mathbf{x}, \mathbf{d}_j) - \hat{g}(\mathbf{x}, \mathbf{d}_j + \mathbf{p}_c) - \frac{1}{2}\mathbf{p}_c^T C_j \mathbf{p}_c}{r_j(\mathbf{d}_j) - r_j(\mathbf{d}_j + \mathbf{p}_c) - \frac{1}{2}\mathbf{p}_c^T C_j \mathbf{p}_c} \geq \eta. \tag{24}$$

Table 1: Solutions produced by the IP in different cases.

Problem	Solution	Cost (€1000)	Delay (minutes)	Exchanges	Cancellations
(30,10)	1	19.5	390	12	0
(30,10)	2	25.0	0	14	2
(30,10)	3	25.0	0	12	2
(30,10)	4	21.0	420	10	0
(30,10)	5	41.5	390	10	2
(25,10)	1	17.5	350	16	0
(25,10)	2	25.0	0	16	2
(25,10)	3	17.5	350	14	0
(25,10)	4	25.0	0	14	2
(25,10)	5	17.5	350	12	0
(25,10)	6	25.0	0	12	2
(25,10)	7	19.5	390	8	0
(25,10)	8	39.5	350	10	2

Coleman and Li (1996) showed that this converges to a local optimal for deterministic optimisation under certain conditions.

It is important to note that this method may still require the simulation of infeasible points to create the meta-model, which may be a problem if such a point cannot be simulated. One possible approach is to use the boundary value as the input to the simulation – maintaining the orthogonal experimental design but violating assumptions that g is smooth; another is to adapt the experimental design to ignore the infeasible region. This is unlikely to retain the orthogonality of 2^k-factorial designs and CCD. Our approach uses the former method as this aids the consistency of the gradient estimates via the experimental design.

Due to the discretisation of time in the IP, some delays may have been added that were not necessary. Thus the solution resulting from the optimisation $(\mathbf{x}, \mathbf{d}^*)$ is compared with $(\mathbf{x}, \mathbf{0})$, that is the choice to not make any delays under the aircraft allocation $\mathbf{x}$. We use a one-sided Welch's t-test to compare:

$$H_0: \ g(\mathbf{x}, \mathbf{0}) \geq g(\mathbf{x}, \mathbf{d}^*), \qquad H_{\text{alt}}: \ g(\mathbf{x}, \mathbf{0}) < g(\mathbf{x}, \mathbf{d}^*).$$

If $(\mathbf{x}, \mathbf{0})$ is found to be an improvement, it is set as the final position.

Once this optimisation has been completed, we are left with a set of improved solutions, each one originating from a solution from the IP. The AOCC decision makers can then consider each of these options in combination with crew and passenger recovery to produce a recovery plan that can be submitted to the relevant authorities.

6 NUMERICAL RESULTS AND DISCUSSION

To demonstrate the algorithm and consider its performance we use the example described in Section 3 and compare the results with the 'No Action' option. 'No Action' does not make any changes to the aircraft allocation, the airline simply waits for $A1$ to be repaired and become available to fly. Whilst this is a very unlikely option to take, it provides a benchmark for comparison. The minimum turn time allowed in the original schedule is 30 minutes. Therefore, we solve the IP with $t_{\min} = 30$ minutes and a more optimistic $t_{\min} = 25$ minutes that attempts to exploit the slack built into the system. For each case a time discretisation of $m = 10$ minutes and a maximum delay of $M = 3$ hours were used. In this section, a problem with minimum turn time $t_{\min}$ and discretisation m will be denoted $(t_{\min}, m)$. The IP performance measures are shown in Table 1. Solutions 5 for (30,10) and 8 for (25,10) appear to be poor solutions on multiple counts. They arise from the order in which the algorithm of Section 4 searches the feasible region. Further research could give more direction to this search procedure and adapt it to prioritise certain regions.

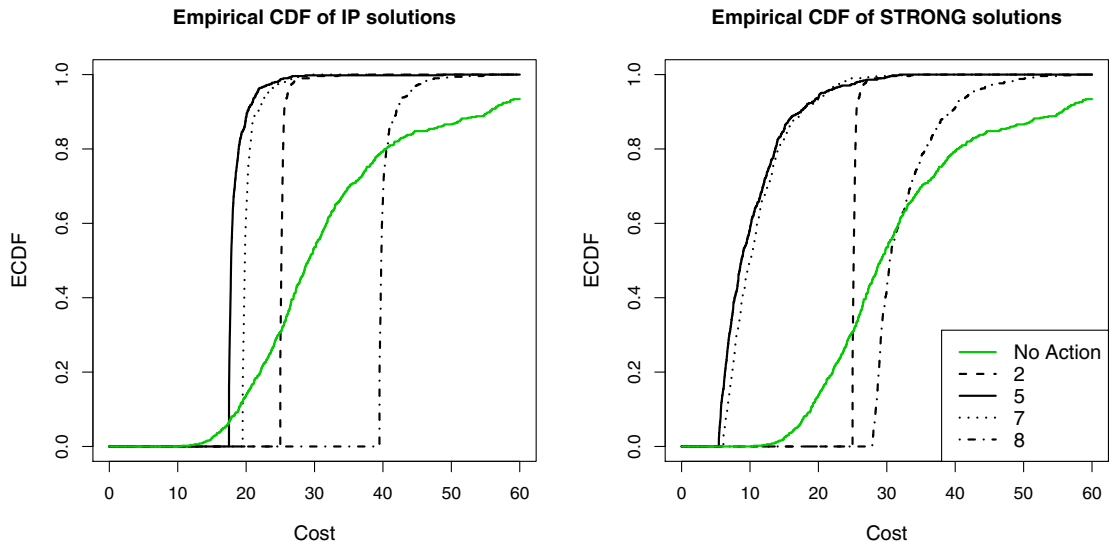

Figure 2: The ECDF of the IP solutions of the (25,10) problem, the simulation optimisation improved solutions and the 'No Action' option with $p = 0.05$ and $\beta =$ €20. Based on over 500 replications.

For further investigation, the penalty value for exceeding a planned delay, β, was varied between €0, €20 and €50 (compared to $\alpha =$ €50) and the probability that the turn time exceeds the 30 minute limit given in the schedule, p, was also varied. The values of p used here were 0.05 and 0.1, and these represent a measure of the level of robustness to minor delays for which the original schedule was planned.

In each case, 2000 simulation replications were allowed for the optimisation procedure, with $n_d = 5$ replications at each design point, and 20 replications used for the acceptance tests (24). The initial trust region size was $\Delta_0 = 5$ minutes and the threshold of switching to the quadratic model was $\tilde{\Delta} = 4$ minutes.

Note that the solutions 2 and 3 for (30,10) and 2, 4, and 6 for (25,10) do not contain any planned delays as the proposed action is cancellations. These solutions do not undergo the simulation optimisation and were just simulated 100 times.

The empirical cumulative distribution functions (ECDF) of the initial and resulting solutions for some of the (25,10) set over 500 replications of the simulation are shown in Figure 2. Solutions 1, 3 and 5 all have the same basic actions regarding the schedule; the same flights are delayed by the same amount in each case but with different aircraft allocations. This leads to very similar performance and so only solution 1 is shown. Similarly for solutions 2, 4 and 6, that all cancel the same two flights, leading to almost identical ECDFs, so only solution 2 is shown. An ideal set of results would see each ECDF moving to the left (reducing the mean) and having a steeper gradient (reducing the standard deviation). The results suggest that the simulation optimisation process does indeed improve the mean performance. However, increases in standard deviation are also seen. Almost all solutions show an improvement in mean and standard deviation over the 'No Action' response. It is possible, however, to get a good outcome from the 'No Action' solution although the probability of doing so is small. Table 2 shows that in this case, an improvement in the 0.9 quantile is seen suggesting that the solutions are robust. Different solutions offer different values. Solution 1 has the lowest mean and 0.9 quantile as well as a reasonably low standard deviation. However, solution 7 has a lower standard deviation and requires fewer changes to the schedule despite its slightly higher mean. Different airlines may have differing priorities on this trade-off.

As a variation on the problem, a zero penalty situation, that is $\beta =$ €0, was considered. The main observations seen in this case are that each improved ECDF was shifted slightly towards 0 and their standard deviation is decreased. This is also mirrored in the 'No Action', whose standard deviation reduces from €15,400 to €12,600 and mean reduces from €33,200 to €24,600. This is not a surprising result as the lack of a penalty allows the airline to be overly optimistic in terms of the delay it will incur. For example,

Table 2: The estimated means, $\hat{g}(\mathbf{x},\mathbf{d})$, standard deviations, $\hat{\sigma}$ and 0.9 quantile, $\hat{q}_{0.9}$, for the cost (€1000) performance for the (25,10) solutions with $\beta =$ €20 and $p = 0.05$.

Problem	Solution	Initial Solution			Improved Solution		
		$\hat{g}(\mathbf{x},\mathbf{d}_0)$	$\hat{\sigma}$	$\hat{q}_{0.9}$	$\hat{g}(\mathbf{x},\mathbf{d}^*)$	$\hat{\sigma}$	$\hat{q}_{0.9}$
-	No Action	33.2	15.4	55.6			
(25,10)	1	18.4	1.60	20.1	10.15	4.80	16.8
(25,10)	2	25.3	0.74	25.6			
(25,10)	3	18.5	1.68	20.6	10.24	4.83	17.4
(25,10)	4	25.3	0.51	25.8			
(25,10)	5	18.5	2.05	20.2	10.44	5.05	17.4
(25,10)	6	25.3	0.62	25.8			
(25,10)	7	20.3	1.43	21.7	11.21	4.59	17.5
(25,10)	8	40.4	1.76	42.0	32.47	4.91	39.6

the final position of solution 5 changes from $(3,43,50,12)$ minutes to $(0,8,13,7)$ minutes, suggesting the algorithm is happier to reduce delays further. This goes alongside a reduction in mean of €2290 and standard deviation of €860. The opposite results occur when increasing the penalty to $\beta =$ €50. This moves the final position of each solution to the right, increasing the mean as the optimisation is more reticent to decrease delays. This is accompanied by an increase in standard deviation. It is interesting to note that the solutions from the IP remain quite robust to the changes in penalty, with only minor changes seen in the ECDF. This suggests that there is significant slack built into these solutions.

We also increased to $p = 0.1$, suggesting a less robust original schedule, thereby increasing the standard deviation. The results suggest a small drop in the performance of the resulting solutions, with solution 5 having mean €10,800, standard deviation €7,300 and 0.9 quantile €19,800, which is a drop compared to that in Table 2. The ECDFs do not appear to show a large degradation in performance, although this may be due to the relatively robust nature of this schedule. Indeed, the differences between the 'No Action' options are minor, suggesting a greater difference in p would be required to alter performance dramatically.

As mentioned previously, we also solved the problem with $t_{\min} = 30$ minutes. The results are similar to those in the (25,10) case, with a reduction in the expected cost alongside an increase in the standard deviation. However, as the solutions given by the IP are worse, the final positions are similarly worse.

7 CONCLUSIONS AND FURTHER WORK

This paper presents a proof-of-concept multi-fidelity modelling approach to the ARP combining both integer programming and simulation optimisation. The algorithm allows for high-fidelity simulations to be used to evaluate solutions in a highly complex environment whilst balancing this with the time constraints and computational issues associated with simulation optimisation. The results in this paper suggest that this method could be used to provide good solutions to the ARP.

The solution time of the IP depends strongly on the number of flights, aircraft and the time step. Size reduction techniques, such as that by Rosenberger et al. (2003), could help to control this for large problems. The computational cost of the simulation optimization stage depends on the number of delays being altered as this affects the number of distinct simulation runs required, and the simulations dominate the computational cost. However, these simulations could be run in parallel.

There are a number of additions one could add to the problem that are not considered here. Some recovery options, such as higher cruise speeds that can help an airline catch up more effectively, whilst costing more in fuel. If these variables are continuous, they could be incorporated into the current framework.

We do not currently have theory about the behaviour of the simulation optimisation method described in Section 5. Future work may be towards looking at what guarantees there are for the performance of this method in the general setting.

Part of the nature of the ARP is that it is a reoccurring problem for airlines. The use of Symbiotic Simulation as discussed by Aydt et al. (2009), in which the model is allowed to adapt to new information from the system it is modelling, could have great value in the area of airlines operation. Future work will include considering how the repeated nature of the problem could be exploited to improve both the models and the optimisation process in a symbiotic manner. Symbiotic simulation could also be used to give new information on the same problem. For example, if there were changes to the maintenance status of aircraft, this could potentially be used to improve the decisions based on the latest information.

ACKNOWLEDGMENTS

We gratefully acknowledge the financial support of the EPSRC funded EP/L015692/1 STOR-i Centre for Doctoral Training, the NSF Grant CMMI-1068473 and Rolls-Royce Limited. We would also like to thank Nigel Jackson, Richard Standing, Stewart Preston and Mike Chester at Rolls-Royce (R^2 Data Labs) for the original research idea and contextual information.

REFERENCES

Abdelghany, K. F., A. F. Abdelghany, and G. Ekollu. 2008. "An Integrated Decision Support Tool for Airlines Schedule Recovery During Irregular Operations". *European Journal of Operational Research* 185(2):825–848.

Allamraju, H. 2017. "pyflightdata". Accessed March 11th, 2017. https://github.com/supercoderz/pyflightdata.

Arias, P., M. M. Mota, D. Guimarans, and G. Boosten. 2013. "A Methodology Combining Optimization and Simulation for Real Applications of the Stochastic Aircraft Recovery Problem". In *Proceedings - 8th EUROSIM Congress on Modelling and Simulation, EUROSIM 2013*, edited by K. Al-Begain et al., 265–270. Washington, DC, USA: IEEE Computer Society.

Aydt, H., S. J. Turner, W. Cai, and M. Y. H. Low. 2009. "Research Issues in Symbiotic Simulation". In *Proceedings of the 2009 Winter Simulation Conference*, edited by M. D. Rossetti et al., 1213–1222. Piscataway, New Jersey: IEEE.

Chang, K.-H., L. J. Hong, and H. Wan. 2013. "Stochastic Trust-Region Response-Surface Method (STRONG) - A New Response-Surface Framework for Simulation Optimization". *INFORMS Journal on Computing* 25(2):230–243.

Civil Aviation Authority 2015. "Your Rights When You Fly". Accessed July 23rd, 2018. https://www.caa.co.uk/Passengers/Resolving-travel-problems/Delays-cancellations/Your-rights/Your-rights-when-you-fly/.

Coleman, T. F., and Y. Li. 1996. "An Interior Trust Region Approach for Nonlinear Minimization Subject to Bounds". *SIAM Journal on Optimization* 6(2):418–445.

Flightradar24 AB 2017. "Flightradar24". Accessed March 7th, 2017. https://www.flightradar24.com.

Guimarans, D., P. Arias, and M. M. Mota. 2015. "Large Neighbourhood Search and Simulation for Disruption Management in the Airline Industry". In *Applied Simulation and Optimization: In Logistics, Industrial and Aeronautical Practice*, edited by M. M. Mota et al., 169–201. Cham: Springer International Publishing.

Gurobi Optimization, LLC. 2017. "Gurobi Optimizer Reference Manual". Accessed July 23rd, 2018. http://www.gurobi.com.

Hutchison, D. W., and S. D. Hill. 2001. "Simulation Optimization of Airline Delay with Constraints". In *Proceedings of the 2001 Winter Simulation Conference*, edited by B. A. Peters et al., 1017–1022. Piscataway, New Jersey: IEEE.

Juan, A. A., J. Faulin, S. E. Grasman, M. Rabe, and G. Figueira. 2015. "A Review of Simheuristics: Extending Metaheuristics to Deal with Stochastic Combinatorial Optimization Problems". *Operations Research Perspectives* 2:62–72.

Kohl, N., A. Larsen, J. Larsen, A. Ross, and S. Tiourine. 2007. "Airline Disruption Management-Perspectives, Experiences and Outlook". *Journal of Air Transport Management* 13(3):149–162.

Laumanns, M., L. Thiele, and E. Zitzler. 2006. "An Efficient, Adaptive Parameter Variation Scheme for Metaheuristics Based on the Epsilon-Constraint Method". *European Journal of Operational Research* 169(3):932–942.

Lee, L. H., H. C. Huang, C. Lee, E. P. Chew, J. Wikrom, Y. Y. Yong, Z. Liang, C. H. Leong, Y. P. Tan, K. Namburi, E. Johnson, and J. Banks. 2003. "Discrete Event Simulation Model for Airline Operations: SIMAIR". In *Proceedings of the 2003 Winter Simulation Conference*, edited by S. Chick et al., 1656–1662. Piscataway, New Jersey: IEEE.

Løve, M., K. R. Sørensen, J. Larsen, and J. Clausen. 2005. "Using Heuristics to Solve the Dedicated Aircraft Recovery Problem". *Central European Journal of Operations Research* 13(2):189–207.

Rhodes-Leader, L., B. S. Onggo, D. J. Worthington, and B. L. Nelson. 2018. "Airline Disruption Recovery Using Symbiotic Simulation and Multi-fidelity Modelling". In *Proceedings of the Operational Research Society Simulation Workshop 2018 (SW18)*, edited by A. Anagnostou et al., 146–155. Birmingham, UK: The Operational Research Society.

Rosenberger, J. M., A. J. Schaefer, D. Goldsman, E. L. Johnson, A. J. Kleywegt, and G. L. Nemhauser. 2000. "SIMAIR: A Stochastic Model of Airline Operations". In *Proceedings of the 2000 Winter Simulation Conference*, edited by J. A. Joines et al., 1118–1122. Piscataway, New Jersey: IEEE.

Rosenberger, J. M., A. J. Schaefer, D. Goldsman, E. L. Johnson, A. J. Kleywegt, and G. L. Nemhauser. 2002. "A Stochastic Model of Airline Operations". *Transportation Science* 36(4):357–377.

Rosenberger, J. M., E. L. Johnson, and G. L. Nemhauser. 2003. "Rerouting Aircraft for Airline Recovery". *Transportation Science* 37(4):408–421.

The AnyLogic Company 2017. "AnyLogic 8.2.3". Accessed July 23rd, 2018. http://www.anylogic.com.

Xu, J., S. Zhang, E. Huang, C.-H. Chen, L. H. Lee, and N. Celik. 2016. "MO^{2}TOS : Multi-Fidelity Optimization with Ordinal Transformation and Optimal Sampling". *Asia-Pacific Journal of Operational Research* 33(3):1650017.

Zhang, D., H. Y. K. Henry Lau, and C. Yu. 2015. "A Two Stage Heuristic Algorithm for the Integrated Aircraft and Crew Schedule Recovery Problems". *Computers and Industrial Engineering* 87:436–453.

Zhu, B., J.-F. Zhu, and Q. Gao. 2015. "A Stochastic Programming Approach on Aircraft Recovery Problem". *Mathematical Problems in Engineering* 2015:680609.

AUTHOR BIOGRAPHIES

LUKE RHODES-LEADER is a Ph.D. student of the Statistics and Operational Research with Industry Centre for Doctoral Training at Lancaster University. His email address is l.rhodes-leader@lancaster.ac.uk.

BHAKTI STEPHAN ONGGO is an Associate Professor of Data Analytics at Trinity Business School, Trinity College Dublin, Ireland. His research interests lie in the areas of predictive analytics using simulation. He is the associate editor for the *Journal of Simulation*. His email address is stephan.onggo@tcd.ie.

DAVID J. WORTHINGTON is a senior lecturer in Operational Research in the Department of Management Science in Lancaster University Management School. He researches the modelling and management of time-dependent queueing systems, and applications of Management Science in healthcare. These research interests often coincide. His email address is d.worthington@lancaster.ac.uk.

BARRY L. NELSON is the Walter P. Murphy Professor in the Department of Industrial Engineering and Management Sciences at Northwestern University and a Distinguished Visiting Scholar in the Lancaster University Management School. His research centers on the design and analysis of computer simulation experiments. His e-mail address is nelsonb@northwestern.edu.