The Aeronautical Journal (2025), 1–35
doi:10.1017/aer.2025.16

ROYAL AERONAUTICAL SOCIETY

RESEARCH ARTICLE

A mathematical model and heuristic approaches for runway rescheduling

G. Hancerliogullari Koksalmis[1,2], G. Rabadi[1], M. Kharbeche[3] and M. Al-Salem[3]

[1]University of Central Florida, Orlando, FL, USA
[2]Istanbul Technical University, Istanbul, Turkiye
[3]Qatar University, Doha, Qatar
Corresponding author: G. Hancerliogullari Koksalmis; Email: gulsah.hancerliogullarikoksalmis@ucf.edu

Received: 8 December 2024; **Revised:** 12 February 2025; **Accepted:** 13 February 2025

Keywords: aircraft operation rescheduling; reactive scheduling; multiple runways; schedule stability; total start times; schedule repair algorithms; disruption management

Abstract

This study addresses the Aircraft Reactive Scheduling Problem (ARSP) on multiple parallel runways in response to operational disruptions. We specifically consider three disruptive event types; flight cancelations, delays and unexpected arrivals. Interruptions to aircraft schedules due to various reasons (e.g. bad weather conditions) may render the initial schedule not optimal or infeasible. In this paper, the ARSP is conceptualised as a multi-objective optimisation problem wherein considerations encompass not only the quality of the schedule but also its stability, defined as its conformity to an initial schedule, are of interest. A mixed-integer linear programming (MILP) model is introduced to obtain optimal solutions under different policies. Repair and regeneration heuristic approaches are developed for larger instances for which optimal solutions are time-consuming to obtain. While prevailing literature tends to concentrate on individual disruption types, our investigation diverges by concurrently addressing diverse disruption types through multiple disruptive events. We introduce alternative reactive scheduling methodologies wherein the model autonomously adapts by dynamically choosing from a range of candidate solution methods, considering conflicting objectives related to both quality and stability. A computational study is conducted, and we compare the solutions of heuristics to optimal solutions or the best solution found within a time limit, and their performances are assessed in terms of schedule stability, solution quality and computational time. We compare the solutions of heuristics and optimal solutions (i.e. the best solution found so far), and their performances are assessed in terms of schedule stability, solution quality and computational time.

Nomenclature

ARSP	aircraft reactive scheduling problem
MILP	mixed-integer linear programming

1.0 Introduction

The aircraft sequencing problem (ASP) has been extensively studied in the literature under various assumptions and settings (e.g. Refs. [1–17]). Beasley et al. [5] considered the problem of scheduling aircraft landings. The authors presented a mixed-integer 0–1 formulation of the problem for the single and multiple runways. They have examined disjunctive formulations which rely on general precedence variables. The main objective function examined was minimising the total weighted tardiness and earliness of aircraft. The authors focused on how their proposed model can be adapted to incorporate both arrivals and departures, and to address alternating objective functions; for instance, minimising the

maximum start-time of operations or minimising the average landing time. In order to enhance the model, preprocessing routines and valid inequalities were proposed. Alternatively, a time-indexed formulation was developed. They further proposed an effective heuristic algorithm for the problem. Beasley et al. [4] defined a displacement problem for dynamic (real-time) aircraft landing problem which includes additional cost for perturbing previous schedules. In order to solve the problem they adapted three solution approaches, one optimal (DALP-OPT) using LP-based tree search and two heuristics (DALP-H1, DALP-H2) given in Ref. [5] for the static aircraf landing problem. Recently, Refs [18–20] have investigated enhanced models and solution approaches for the single- and multiple-runway ASP under mixed-mode operations (i.e. aircraft landings and departures are considered simultaneously). In the ASP, n aircraft have to be scheduled on m parallel runways where each aircraft j has a priority weight w_j, a ready-time r_j before which it cannot be scheduled, a target time δ_j for its operation (a departure or a landing) and a deadline d_j. A separation time denoted as s_{kj}, contingent upon the sequence of aircraft, is implemented to mitigate the potential risks associated with wake-vortex effects when aircraft j follows the operations of aircraft k. The values of s_{kj} are contingent upon the type of aircraft operations, distinguishing between departures and arrivals, as well as the size-class classification of the aircraft. For example, sequencing a small aircraft before a large one requires less separation time than the opposite. Hence, aircraft separation times are considered sequence-dependent and, depending on the standard adopted, can be non-triangular in that the immediate separation of consecutive aircraft does not always guarantee sufficient separation between non-consecutive aircraft (Ref. [19]). The initiation time of the operation pertaining to aircraft j is represented by t_j, and the lateness is quantified as $T_j = \max\left(t_j - \delta_j, 0\right)$. Deviating from the stipulated time for aircraft j is acceptable within specified tolerances, incurring a weighted tardiness cost of $w_j T_j$. Conversely, surpassing the deadline is impermissible; if aircraft j exceeds its deadline, it will be precluded from runway assignment and classified as 'unscheduled', leading to an infeasible solution.

In this paper, we extend the ASP to the ARSP in which disruptive events may occur and require an initial schedule to be reactively revised using two dynamic strategies: repair or reschedule. When a schedule is repaired, local changes are introduced in order to recover a feasible, and possibly attractive, schedule; however when it is rescheduled, the entire schedule is regenerated. In doing so, we consider two objectives that may be antithetical, namely, the schedule quality with respect to the original objective function and the schedule stability (or conformity to the original schedule). Minimising the impact of disruptions on the quality of the schedule (such as meeting target times) is very important to maximise passenger satisfaction. On the other hand, it is important to maintain a reasonable level of schedule stability in order to reduce changes that are introduced in the original schedule. As such, the ARSP is framed in this paper as a multi-objective optimisation problem that minimises the total weighted start time, start time deviation and runway deviation. A mathematical model is introduced to find optimal solutions under three types of disruptive events, namely, cancelations, delays and new aircraft arrival. Since the total weighted start time minimisation even for a single machine is NP-hard (Ref. [21]), according to the complexity of hierarchy (Ref. [22]), ARSP is also NP-hard and therefore, several schedule repair algorithms are introduced for the problem.

1.1 Aviation in real-life

Aircraft are directed from departure to arrival and across the orderly airspace by air traffic controllers whose main responsibility is guaranteeing a secure separation of aircraft. In adherence to safety regulations, it is mandated that a singular aircraft utilise the runway at any given time, the landings and take-offs are sequenced based on the requirements for instance smallest deviation from the initial schedule and volume of the runway. The goal of this subsection is to characterise the essential notions identified with the runway activities for sequencing aircraft including the decision-support tools, that help air traffic controllers to sequence and manage aircraft are discussed.

https://doi.org/10.1017/aer.2025.16 Published online by Cambridge University Press

1.1.1 Decision-support tool

According to the Federal Aviation Administration (FAA), the key duty of air traffic controllers is ensuring the adequate spacing of aircraft. Controllers connect with pilots, informing them of traffic or weather in their region. Pilots rely upon the guidelines they get from air traffic control to securely and proficiently make a trip from their origin to their destinations. Interestingly, traffic handling encourage a 'system approach' to overseeing traffic that takes into account the effect of individual activities all in all. Handling disruptions in airspace capacity (caused for instance by terrible weather, traffic loads or emegencies) requires attention of who or what might be affected by the disruption, and a corresponding justification power to guarantee safety, security and efficency. Without an organised reaction, nearby flight delays because of little interruptions can rapidly swell over the whole U.S., causing rerouting, flight cancelations and delays [23].

The vast majority of air terminals utilises the Next Generation Air Transportation System (NextGen) which is the FAA-drove modernisation of America's air transportation framework to make flying significantly more secure, more proficient and more unsurprising. NextGen mechanisation frameworks compose and increment the perceivability of flight data. Three decision support systems enable controllers, traffic managers and different shareholders to rapidly and effectively react to advancing traffic interruptions and climate conditions. The NextGen version of the inheritance Enhanced Traffic Management System (ETMS) can foresee air traffic volume, gaps and floods dependent on current and foreseen airborne airplane at nearby and public levels. It mitigates issues that require proactive arranging, coordination and alterations, for example, bad weather. Such mitigation is expected to decrease negative results of these limitations, which incorporate delays, missing connecting flights, flight cancelations and increase in fuel consumption. At the point when delays are unavoidable, it permits each flight administrator to present the best course alternatives and substitute trips to fulfill business targets for their planned aircraft, for instance, to guarantee more individuals are on time not to miss connecting flights (Ref. [24])

1.1.2 Separation

The duty of air traffic regulators is also to guarantee that a minimum separation between the arrival/departure of one airplane and the arrival/departure of another airplane is kept up, where various versions of separation might be characterised. One of the aspects determining capacity of runway is the separation criteria. The most widely used separation standards are the following: 'radar separation', which is a longitudinal spacing of 5NM 3NM in congested area) and a vertical spacing of 1,000 ft, 'wake turbulance separation', which is a time spacing that relies upon the dimensions and weight of the aircraft. The International Civil Aviation Organization (ICAO) characterises three classes of wake turbulence, to be specific Light (L), Medium (M) and Heavy (H). Besides, several elements are considered in the use of separation between aircraft such as sequence, inbound and outbound routes, the aircraft types, the weather conditions and the wake turbulence category (Ref. [25]). The following minima is applied to aircraft landing behind a Heavy or a Medium aircraft: 2 minutes if Medium behind Heavy aircraft; 3-minutes if Light behind a Heavy or Medium aircraft. Moreover, a minimum separation of 2 minutes is applied between a Light or Medium aircraft departing behind a Heavy or a Light aircraft taking off behind a Medium aircraft (Ref. [26]).

On the other hand, since ICAO's separation are old-fashioned the FAA agreed a recategorisation (RECAT) of wake turbulence separation minima to RECAT Phase I. This agreement was based on long periods of joint innovative work by the FAA, Eurocontrol, scientists studying on wake and specialists in security and risk analysis. Classifications are dependent on weight, certificated approach speeds and wing features, RECAT places airplane into six (6) classes (marked A-F) for both take-off and landing separation(Ref. [27]). Our suggested model is considering real world separation constraints (between 30 and 200 seconds that depends mainly on the size of the aircraft) and the used approach is scalable to any airport. Regarding the time window, we generated it based on some real data collected from Doha International Airport. Specific separation times used in this paper, which are similar to those in Refs [3, 20, 28].

https://doi.org/10.1017/aer.2025.16 Published online by Cambridge University Press

1.1.3 Disruptions

A flight delay is the point at which an aircraft arrives and/or departs later than its planned time. FAA believes a trip to be delayed when it is 15 minutes after than its planned time. A cancelation happens when the aircraft doesn't work the trip at all for a certain reason(Ref. [29]). Since June 2003, the carriers report on-time information additionally report the reasons for delays and cancelations to the Bureau of Transportation Statistics. The carriers report the reasons for delay in general classifications that were made by the Air Carrier On-Time Reporting Advisory Committee. The classes are Air Carrier, National Aviation System, Weather, Late-Arriving Aircraft and Security. The reasons for cancelation are similar, aside from there is no late-showing up airplane classification. Air Carrier: The reason for the cancelation or delay was because of conditions inside the aircraft's control (for example maintenance or crew issues, airplane cleaning, baggage loading, fueling and so forth). Weather: Significant meteorological conditions (real or estimated) that, in the judgement of the carrier, delays or forestalls the activity of a flight, for example, tornado, blizzard or hurricane. National Aviation System (NAS): Delays and cancelations owing to the public flight framework that allude to a wide arrangement of conditions, for example, non-extreme weather conditions, air terminal activities and substantial traffic volume. Late-arriving aircraft: A past trip with same airplane showed up after the expected time, making the current flight leave late. Security: Delays or cancelations brought about by evacuation of a terminal or concourse, re-boarding of airplane in view of security break, broken screening gear as well as long queues more than 29 minutes at screening areas (Ref. [30]).

Furthermore, any arrival made under some level of pressure, on or off an air terminal, that is made essential by the incapacity to proceed with further flight is by and large viewed as a forced landing or unexpected new aircraft arrival. Regularly, this is on the grounds that a pilot depleted his fuel, starved the motor by endeavoring trip with a tank that had zero remaining, or fumbled the mixture (Ref. [31]). Hospital airplane may carry a sick or seriously injured individual requiring critical medical care. VIP airplanes are given with a high priority. The presidential flight is treated as VIP flight where exceptional handling with is worked out, for example, shutting the aerodrome for any period of time (Ref. [32]).

The subsequent sections of this manuscript are structured as outlined below. Section 2.0 furnishes a synopsis of the existing body of literature. The rest of this study is structured as follows. Section 2.0 discusses the related literature. Section 3.0 formally introduces our notation along with the proposed multi-objective MILP. It also discusses the scalarisation of the objective function. Section 4.0 presents alternative heuristic approaches that are compared against optimal solutions in our computational study in Section 5.0. Section 6.0 concludes this work with a summary of our findings.

2.0 Literature Review

There exists an elaborate literature on aircraft scheduling and runway optimisation problems. The reader is referred to Refs [20, 33–35] for discussions of popular exact and heuristic solution methods for aircraft landing and/or departure problems. Less research is available on aircraft reactive scheduling as compared to aircraft scheduling. A survey by Ref. [36] identifies the following categorising attributes of the rescheduling problem: rescheduling environments (static or dynamic), rescheduling strategies (periodic, event-driven or hybrid), and methods (partial rescheduling or complete regeneration). The studies related to rescheduling have concentrated on various job-related disruptions such as new job arrival (Refs [37–42]), order cancelations (Refs [43, 44]) and changes in due-date (Ref. [45]), and machine-related disruptions such as machine breakdowns (Refs [46, 47]).

Both exact and heuristic algorithms have been proposed for reactive scheduling problems to repair the original schedule. Alagöz & Azizolu and Arnaout & Rabadi [48, 49] for example considered machine-related disruption. Curry and Peters [50] developed branch-and-price algorithm to minimise tardiness and reassignment costs when the schedule is disrupted by an arrival of new job. Further, heuristic algorithms including dispatching rules have been developed with the objective of achieving high solution quality and schedule stability. Bean et al. [13] proposed a match-up partial repairing strategy for the rescheduling problem for minimising the total tardiness. Abumaizar and Svestka [51] developed the AP

https://doi.org/10.1017/aer.2025.16 Published online by Cambridge University Press

algorithm for the job shop rescheduling problem. Yang et al. [42] proposed first-in-first-out dispatching rule for the parallel machine rescheduling problem when there is a job arrival in uncertainty. Some others addressed the job shop rescheduling problem (Refs [43–45, 51–58]).

Reactive scheduling problems should consider not only solution quality but also schedule stability. Deviations between the new start times and the initial or planned start times have been used as a measure of stability (Refs [45, 57]). Some other schedule stability measures are proposed in Ref. [51]. Being related to airline operations disturbances, recently, Ref. [59] conducted an extended review of airline disruption management between 2010–2024. Santana et al. [60] presented a systematic literature review for aircraft recovery problem for a time span of 1984–2022. Hassan et al. [61] provided a critical review and classification of the literature between 2009 and 2018 regarding airline disruption management, including aircraft, crew,passenger, and integrated recovery. Similarly, Ref. [62] reviewed the literature on recovery of airlines and airports from schedule perturbations. However, there is no study provided in these review papers related to runway rescheduling problem. The work by Ref. [63] delved into advancements in disruption management, elucidating their application within Operations Research across diverse sectors such as telecommunications, ship-building and the airline industry. Recently, Ref. [64] developed an integer linear programming formulation to minimise the number of stranded passengers for flight rescheduling during airport access mode disruptions. In a distinct study, Ref. [65] addressed the perturbation problem in airline schedules, specifically induced by ground delay programmes. They formulated an integer programme with the aim of minimising the maximum delay among outbound flights. Additionally, Ref. [66] examined scenarios involving one or more incapacitated aircraft, employing the branch-and-bound technique to minimise overall passenger delay while seeking the most cost-effective aircraft routings. Argüello et al. [67] proposed Greedy Randomized Adaptive Search Procedure for aircraft routing in response to groundings and delays. Petersen et al. [68] presented an optimisation-based approach to solve the integrated airline recovery problem to repair the flight schedule, aircraft rotations, crew schedule and passenger itineraries. Artigues et al. [69] reviewed the most prominent methods proposed by the candidates of ROADEF (the French society of Operational Research and Decision Making) challenge on disruption management for commercial aviation posed a large-scale integrated aircraft and passenger scheduling problem. Bisaillon et al. [70] introduced a large neighbourhood search heuristic for an airline recovery problem combining fleet assignment, aircraft routing and passenger assignment, which was ranked first in the ROADEF Challenge. Jozefowiez et al. [71] considered the integrated problem of aircraft rotation and passenger itinerary recovery, and developed a heuristic that took into account passengers and aircraft with the same priority. Brunner [72] addressed the rescheduling problem by an airline when a ground delay programme is issued, and presented a linear integer model in order to minimise delay measures, cost for crew and passenger misconnections, and cost of flight cancelations. Aktürk et al. [73] focused on airline recovery optimisation model along with the environmental constraints and costs, and included the cruise speed as a decision variable. The authors utilised conic mixed-integer programming and implemented conic quadratic optimisation approach to solve the problem optimally. Castro et al. [74] proposed a new multi-agent system approach to Airline Disruption Management based on intelligent agents taking into account aircraft, crew and passengers. In their research, Ref. [75] investigated a complex aircraft recovery problem characterised by multiple objectives. The primary goals included minimising the overall deviation from the original flight schedules, limiting the maximum flight delay time, and optimising the number of aircraft involved in swapping.

Recently, Ref. [76] proposed an automated approach based on ant colony optimisation to solve both aircraft assignment and aircraft recovering problems while considering disrupted passengers as part of the cost function when there is an airline disruption. Niendorf et al. [77] focused on the problem of stability analysis for a runway schedule with respect to delays of aircraft. In particular, they investigated whether the landing sequence of aircraft remains optimal after an arbitrary number of aircraft in that sequence are delayed by an arbitrary amount of time on a single runway. Sama et al. [78] proposed mixed integer linear programming formulations to evaluate the trade-off between various performance indicators of practical interest. The results of the optimal solutions were compared with a commonly used

https://doi.org/10.1017/aer.2025.16 Published online by Cambridge University Press

scheduling rule, first come first served (FCFS). Sama et al. [79] presented a number of algorithms and proposed metaheuristics in order to solve aircraft scheduling and re-routing problems at busy terminal control areas under various types of disturbances, including multiple aircraft delays and a temporarily disrupted runway. Nisse et al. [80] studied recovery of disrupted airline operations using k-maximum matchings in graphs when the landing of the aircraft is delayed due to bad weather conditions, late aircraft arrivals or if other aircraft have to land first. Similarly, Ref. [81] focused on rescheduling and cost allocation mechanism for a sequence of arrivals that are subject to a delay event at a common destination. Kjenstad et al. [82] presented an integrated approach to departure management and surface routing in airports for dynamic rescheduling in real-time environment. Rodríguez-Díaz et al. [83] proposed simulated annealing to minimise delays in the scheduled times of arrival and departure flights in an airport with a mixed-operation runway, under wake vortex separation and constrained position shifting restrictions. Ng et al. [84] developed artificial bee colony algorithm for aircraft sequencing and scheduling problem under the uncertainty of arrival and departure delays for mixed-mode operations. Kammoun and Rezg [85] focused on aircraft routing and rescheduling problem under airspace capacities uncertainty due to unexpected weather conditions. A hybrid approach and a genetic algorithm were proposed to solve the problem. Lin and Wang [86] proposed a fast variable neighbourhood search-based algorithm for flight rescheduling after airport closure. Ali and Nidhal [87] applied genetic algorithm for continuous flight rescheduling problem. Erkan et al. [88] proposed a generic mathematical model for rescheduling of arrivals and departures to minimise total average delay, maximum delay and delay differences among aircraft. Ng et al. [89] implemented a two-stage robust optimisation approach in terminal traffic flow in order to minimise delay throughout the air traffic flow network and the vulnerability to disruption. They checked the performance of the variants of pareto-optimal cut and the dynamic core point selection scheme using simulated annealing algorithm. Recently,Ref. [90] proposed a mathematical model, genetic algorithm and imperialistic competitive algorithm to minimise total damage due to delays in aircraft operations.

Aircraft rescheduling problems in the literature mainly focuses on flight rescheduling and passenger recovery problems. However, different from the literature, in our paper paper, we focus on runway operations; specifically, we are developing models and heuristics for runway rescheduling. To the best of our knowledge, there is no other study in the literature studied our problem that addresses different types of disruptions (flight cancelations, aircraft delays and unexpected, new aircraft arrivals) with multiple disruptive events on single- and multiple-runway under mixed-mode operations. Our aim is to maximise runway throughput and minimise deviation from the given initial schedule. Therefore, research on runway reactive scheduling problems due to multiple disruption types with multi-objectives remains limited, and adopting and implementing the existing methods presented in the literature would not work as our problem definition is different. To fill the literature gap, the present study introduces a MILP model featuring a normalised objective function to identify optimal solutions. Additionally, heuristic algorithms are presented to efficiently derive near-optimal schedules. The evaluation of the trade-off among the objective terms is emphasised, wherein the total weighted start time component signifies solution quality. Meanwhile, the total weighted start time deviation and total weighted runway reassignment are indicative of solution stability. In contrast to prevalent literature, which predominantly concentrates on singular disruption types, our investigation considers diverse disruption types concurrently, accommodating variations in both type and frequency.

3.0 MILP

The MILP introduced here allows (with penalty) runway re-assignment and constrained position shifting (CPS) from the initial schedule. Initial runway assignments and start times for the original schedule are used as input to the MILP, which produces a new schedule at the minimum cost of schedule quality and stability. It is assumed that only a subset of aircraft is eligible for re-scheduling, because some of the earlier aircraft operations had already taken place and some others are about to take place and cannot, for practical reasons, undergo last minute changes.

https://doi.org/10.1017/aer.2025.16 Published online by Cambridge University Press

3.1 Notation and formulation

This section presents the proposed optimisation model. To this end, we introduce the following notation:

3.1.1 Index sets and parameters

- $M = \{1, \ldots, m\}$: Set of m parallel, independent runways.
- $\bar{J} = \{1, \ldots, n\}$: Set of n aircraft (i.e., landing or departing) originally scheduled.
- $\mathcal{D} \subseteq \bar{J}$: Set of delayed aircraft.
- $\mathcal{E} \subseteq \bar{J} = \{1, \ldots, n\}$: Set of canceled aircraft.
- $\mathcal{A}$: Set of new aircraft arrivals.
- $J = \left(\bar{J} - \mathcal{E}\right) \cup \mathcal{A}$. Set of aircraft that are considered for rescheduling.
- r_j: Ready time for aircraft j to take-off or land, $\forall j \in J$.
- δ_j: Target time for aircraft j to take-off or land, $\forall j \in J$.
- d_j: Deadline for aircraft j to take-off or land, $\forall j \in J$.
- O_j: Operation type of aircraft j, being a landing or a departure, $\forall j \in J$.
- C_j: Weight class of aircraft j, being heavy, large, or small, $\forall j \in J$.
- w_j: Objective coefficient or weight associated with the importance of aircraft j based on its operation type and weight class, $\forall j \in J$. Note that higher priority is assigned to landings over departures and to heavy aircraft over large and small ones.
- s_{kj}: Minimum separation time required between aircraft k and j if they are respectively the leading and the following aircraft, $\forall j, k \in J, j \neq k$.
- α_j: Penalty cost of deviation from the initial start time of aircraft j, $\forall j \in \bar{J} - (\mathcal{D} \cup \mathcal{E})$.
- β_j: Penalty cost of deviation from the initial runway assignment of the aircraft j, $\forall j \in \bar{J} - (\mathcal{D} \cup \mathcal{E})$.
- π_1: Penalty cost for the total weighted start time deviation (TWSD) from the initial start times for all aircraft that are being rescheduled but not experiencing any disruption.
- π_2: Penalty cost for the total weighted runway deviation (TWRD) from the initial runway assignment for all aircraft that are being rescheduled but not experiencing any disruption.
- π_3: Penalty cost for the sum of the total weighted start times (TWS) of all aircraft that are being rescheduled.

Furthermore, the solution associated with the initial schedule is summarised as follows:

- $\bar{t}_j$: Start time of the operation for aircraft j in the initial schedule, $\forall j \in \bar{J}$.
- $\bar{z}_{ij} = 1$ aircraft j had been assigned to runway i in the initial schedule, $\forall i \in \bar{M}$ and $\forall j \in \bar{J}$.

3.1.2 Decision variables

- t_j: the start time of aircraft j (i.e. the time for departure or landing), $\forall j \in J$.
- $z_{ij} = 1$ if aircraft j is assigned to runway i, $\forall i \in M, j \in J$.
- $y_{kj} = 1$ if both aircraft k and j are assigned to the same runway and k precedes j $t_j > t_k$, $\forall k, j \in J, k \neq j$.
- g_j, q_j, u_j, and $o_j \geq 0$ $\forall j \in \bar{J}$.

https://doi.org/10.1017/aer.2025.16 Published online by Cambridge University Press

The MILP formulation is stated as follows:

$$\text{Minimise} \quad \pi_1 \sum_{j\in\bar{J}-(\mathcal{D}\cup\mathcal{E})} \alpha_j\left(g_j+q_j\right)+\pi_2 \sum_{j\in\bar{J}-(\mathcal{D}\cup\mathcal{E})} \beta_j\left(u_j+o_j\right)+\pi_3 \sum_{j\in J} w_j t_j \tag{1}$$

$$\sum_{i\in M} z_{ij}=1, \quad \forall j\in J \tag{2}$$

$$r_j \le t_j \le d_j, \quad \forall j\in J \tag{3}$$

$$t_j \ge t_k + s_{kj} - \left(1-y_{kj}\right)\left(d_k - r_j + s_{kj}\right), \quad \forall j_1\in J, j_2\in J, j_1\ne j_2 \tag{4}$$

$$y_{kj}+y_{jk} \ge z_{ik}+z_{ij}-1, \quad \forall i\in M, j_1\in J, j_2\in J, j_1<j_2 \tag{5}$$

$$t_j+g_j-q_j=\bar{t}_j, \quad \forall j\in\bar{J}-(\mathcal{D}\cup\mathcal{E}) \tag{6}$$

$$\sum_{i\in M} i z_{ij}+u_j-o_j=\sum_{i\in M} i\bar{z}_{ij}, \quad \forall j\in\bar{J}-(\mathcal{D}\cup\mathcal{E}) \tag{7}$$

$$y, z \text{ binary}, \;\; g, q, u, o \ge 0. \tag{8}$$

The objective function (1) minimises the total cost which comprises the total weighted deviation from the initial start times, the total weighted deviation from the initial runway assignments for all rescheduled aircraft that have not experienced any disruption, and the total weighted start times of all aircraft where $\pi_1+\pi_2+\pi_3=1$. Constraint (2) ensures that every aircraft is assigned to exactly one of the m runways. Constraint (3) specifies a time-window for each aircraft. Constraint (4) enforces minimal separation times between aircraft that are assigned to the same runway. Constraint (5) ensures that if two aircraft are assigned to the same runway, then one must operate before the other. Constraint (6) captures the deviation from initial start times where (g_j+q_j) will measure this deviation. Constraint (7) accounts for the deviation from initial runway assignments where (u_j+o_j) will measure this deviation. Constraint (8) introduces binary restrictions on runway assignment and aircraft sequencing variables as well as nonnegativity restrictions on deviation variables.

3.2 Objective function normalisation

The objective function in the proposed model has a composite nature. The first two terms address deviations from start-times (in seconds typically) and runway assignments, whereas the third deals with the overall weighted start times of aircraft. Due to this heterogeneity and the varying orders of magnitude in the objective terms, it is pertinent to transform the original objective functions. The use of scalarisation methods is particularly useful in settings such as ours where scaling of the objective penalties may be challenging. Here, the objective function can be normalised for its terms to become dimensionless, whereby the transformed objective function minimises a convex combination of normalised instability and total weighted start times terms. To this end, the following scalarised objective function (similar to Ref. [91]) is used for ARSP:

$$\begin{aligned}\min \mathrm{Z} = \; & \pi_1 \left(\frac{\sum_{j\in\bar{J}-(D\cup E)} \alpha_j\left(g_j+q_j\right) - f_{\widetilde{TWSD}}\left(x^*\right)}{f^{\widetilde{max}}_{TWSD} - f_{\widetilde{TWSD}}\left(x^*\right)} \right) + \pi_2 \left(\frac{\sum_{j\in\bar{J}-(D\cup E)} \beta_j\left(u_j+o_j\right) - f_{\widetilde{TWRD}}\left(x^*\right)}{f^{\widetilde{max}}_{TWSD} - f_{\widetilde{TWRD}}\left(x^*\right)} \right) \\ & + \pi_3 \left(\frac{\sum_{j\in J} w_j t_j - f_{\widetilde{TWS}}\left(x^*\right)}{f^{\widetilde{max}}_{TWS} - f_{\widetilde{TWS}}\left(x^*\right)} \right)\end{aligned} \tag{9}$$

where $f_{\widetilde{TWSD}}\left(x^*\right)$, $f_{\widetilde{TWSD}}\left(x^*\right)$, $f_{\widetilde{TWS}}\left(x^*\right)$, $f^{\widetilde{max}}_{TWSD}$, $f^{\widetilde{max}}_{TWRD}$, and $f^{\widetilde{max}}_{TWS}$ are estimates of the optimal and worst values of the individual objective components of weighted start time deviations, weighted runway assignment deviations, and weighted start times, respectively. Each represents a theoretical optimistic or pessimistic objective value for an individual objective term. To estimate these values, it is necessary to run the MILP model three times for different combinations of π_1, π_2, π_3. For instance, $f_{\widetilde{TWSD}}\left(x^*\right)$ is the

https://doi.org/10.1017/aer.2025.16 Published online by Cambridge University Press

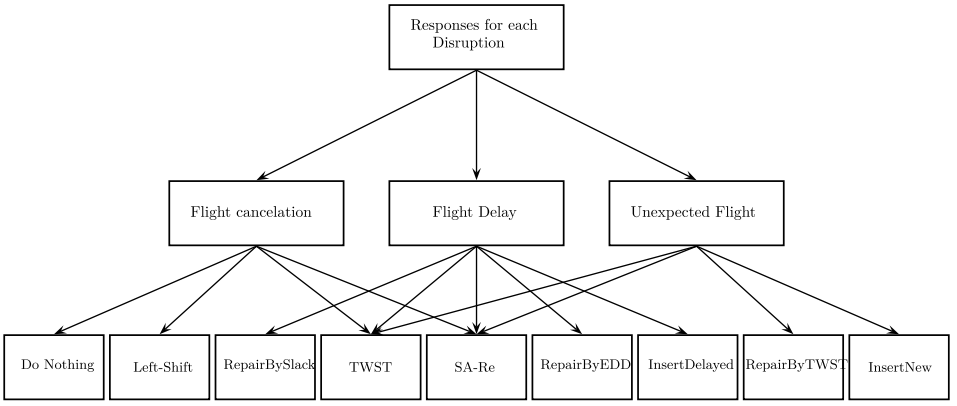

Figure 1. *Reactive scheduling algoritms for each disruption type.*

estimate of the minimum total weighted start time deviation for a given instance, and can be estimated after the first MILP run by setting $\pi_1 = 1 - \pi_2 - \pi_3$, with $\pi_2 = \pi_3 = \varepsilon$, where ε is a very small positive real number. In order to clarify the methodology of estimating the minimum and maximum values of objective function terms, sample data of 15 aircraft and 2 runways are provided and discussed in detail in Appendix.

4.0 Heuristic Algorithms

Unlike most studies in the literature, which focus on one type of disruption at a time, we consider different types and frequency of disruptions simultaneously. We follow a sequential evaluation method in which we start with flight cancelation followed by flight delays and finally new unexpected flights. Although disruptions can be treated in a different sequence, our preliminary experiments showed that better results are obtained by following this sequence of cancelations, delays, and then new flights. Moreover, in practice, addressing these disruption types in this order is more meaningful. At each stage, we evaluate the new schedule with respect to the normalised combined objective function value when different heuristic algorithms are applied to the disruption and the best is selected. Figure 1 illustrates reactive scheduling algorithms that are developed for the corresponding events. As the time horizon under consideration becomes longer, the uncertainty increases and more disruptive events will be more likely to occur. Our approach deals with smaller time blocks and to handle long time horizons (e.g., a whole day), the horizon can be divided into multiple smaller time blocks and the algorithms can be re-run sequentially for the blocks.

These response strategies require an initialisation stage to determine the rescheduling point and the set of aircraft that are affected by the disruption. It is assumed that the initial schedule is given, and at the beginning of every time period, the disruption information is updated. In the following subsections, we introduce a suite of heuristic algorithms to repair or completely reschedule the aircraft schedule when such disruptions occur.

4.1 TWST Algorithm

TWST (Total Weighted Start Time Algorithm) is a complete regeneration algorithm that reschedules all aircraft from scratch and minimises the total weighted start times. This algorithm will be appropriate when the start times are more heavily weighted compared to the instability objective components. The procedure for TWST algorithm is as follows:

https://doi.org/10.1017/aer.2025.16 Published online by Cambridge University Press

Algorithm 4.1 *TWST Algorithm*

1: Let J be a set of unscheduled aircraft.
2: Get initial schedule (schedule before any disruption).
3: Check the disruptions information (cancelation, delay, unexpected flight).
4: Update the input data (i.e., ready time, target time, etc.) according to the following disruption types:
5: – For canceled flight: remove the canceled flight and its parameters from the input data.
6: – For delayed flight: update the ready time, target time and deadline.
7: – For new flight: insert new flight and its parameters into the input data.
8: $\forall j \in \{J \cup A\}$, Calculate the ratio $w_j / \left(r_j + s_{kj}\right)$.
9: Assign the aircraft with the largest ratio in Step 8 to a runway on which its operation can start earliest and remove it from set J.
10: Update the makespan of the runway on which the aircraft is assigned in Step 9.
11: Go to Step 8 and continue until all aircraft are scheduled.
12: Calculate the normalised objective function given in Equation 9.

The rationale of using the ratio in Step 8 is to give higher priority to more important flights and/or to those that are available earlier with short separation times. In essence, this algorithm schedules aircraft with smaller start times first.

4.2 SA-Re algorithm

SA-Re is an algorithm that reschedules all aircraft from scratch. It is similar to the Simulated Annealing (SA) metaheuristic which was successfully implemented for the scheduling problem in Ref. [20], but modified for the rescheduling problem. SA-Re is introduced for the problem and attempts to improve initially constructed solutions obtained by using the TWST algorithm to instances with disruptions. The SA parameters in SA-Re were fine-tuned using the Taguchi Experimental Design approach followed in Ref. [20]. A metaheuristic like SA-Re is very useful in turning an infeasible schedule, which can be produced by a greedy algorithm such as TWST by scheduling some operations' start times past their deadline, into a feasible schedule. The procedure for SA-Re is given as follows.

- S: search area
- θ: current solution
- θ': neighbour solution of the current solution
- θ^*: best solution
- $f(\theta)$: objective function value of the current solution using Equation 9
- $N(\theta)$: neighbourhood of θ
- M: memory set of current best solution and objective function value
- i: inner loop iteration counter
- i_{max}: max number of inner loop iterations
- c: iteration counter
- t_{max}: max number of iterations
- T: temperature
- k: initial temperature coefficient
- α: temperature cooling coefficient

https://doi.org/10.1017/aer.2025.16 Published online by Cambridge University Press

Algorithm 4.2 *SA-Re Algorithm*

1: Get the initial solution from TWST Algorithm as an initial solution θ from S.
2: Update the objective function value, calculate $f(\theta)$.
3: Initialise memory, *Memory* $M = \{(\theta, f(\theta))\}$.
4: Set iteration counters $i = 0, c = 0$.
5: Set the initial temperature as a function of the current objective function of the current objective function value, $T = k.f(\theta)$.
6: **while** $c < t_{max}$ **do.**
7: **while** $i < i_{max}$ **do.**
8: Generate a new solution in the neighbourhood of the initial solution.
9: Choose $\theta \in N(\theta) \subseteq S$ where $M = \{\theta,\ f(\theta')\}$, do neighbourhood search algorithm.
10: **if** there is at least one unscheduled aircraft **then.**
11: use *Aircraft exchange_1*.
12: **else.**
13: use *Aircraft exchange_2*.
14: **end if.**
15: Calculate $f(\theta')$.
16: **if** $f(\theta') - f(\theta) \leq 0$ or $rand[0, 1] \leq e^{-\frac{\Delta\theta}{T}}$, where $\Delta\theta = f(\theta') - f(\theta)$ **then**
$M = \{(\theta,\ f(\theta'))\}$ 17:
18: **end if**
$i = i + 1$ 19:
$c = c + 1$ 20:
21: **end while.**
22: Cool down/update temperature $T = \alpha.T$.
23: **end while.**
24: Output θ^* and $f(\theta^*)$.

Aircraft exchange_1 and *Aircraft exchange_2* functions are used to perturb the current solution locally.

Aircraft exchange_1: When there is a randomly selected unscheduled aircraft j, it is exchanged with another randomly selected aircraft i such that $r_j < r_i$ and $d_j < d_i$.

Aircraft exchange_2: If there are no unscheduled aircraft, then randomly selected aircraft j is exchanged with randomly selected aircraft i. This neighbourhood is applied to all aircraft across the runways.

The difference between both is that *Aircraft exchange_1* is applied when there is an unscheduled aircraft (i.e.,$d_j < t_j$); otherwise *Aircraft exchange_2* is executed.

4.3 Do-Nothing algorithm

Do-Nothing strategy is a type of response that is applied when flight cancelation disruption occurs. After taking out an aircraft from its position on a runway, no corrective action is taken for the remaining aircraft assigned to that runway. Therefore, Do-Nothing keeps the initial runway assignments, flights sequence on each runway and start times as they are. The rationale behind this response is to generate a stable schedule that does not deviate much from the initial schedule after a flight is canceled. The procedure for the Do-Nothing strategy is given as follows:

https://doi.org/10.1017/aer.2025.16 Published online by Cambridge University Press

Algorithm 4.3 *Do-Nothing Algorithm*

1: Get initial schedule (schedule before any disruption).
2: Get the canceled flight j $\forall j \in \mathcal{E}$.
3: Take out flight j from the initial schedule (remove j from $\bar{J}$).
4: Recalculate the normalised total weighted start time and update total objective function value excluding the removed flight from the calculation of the objective function value.

Algorithm 4.4 *Left-Shift Algorithm*

1: Get the initial schedule before disruptions.
2: Get the canceled flight j ($\forall j \in \mathcal{E}$).
3: Determine the runway i for canceled flight j.
4: Remove aircraft j from the initial schedule (remove j from $\bar{J}$).
5: Shift the aircraft immediately succeeding j on runway i to the to the most left without violating the separation requirement with the preceding aircraft.
6: Update start times.
7: Repeat steps 4,5 and 6 until $\mathcal{E} = \{\ \}$.
8: Calculate the combined objective function value in Equation 9.

4.4 Left-shift algorithm

Left-shift strategy is a partial repair algorithm geared towards schedule repair after flight cancelation. It removes the aircraft with the canceled flight and keeps the runway assignment and aircraft sequence for the remaining aircraft on each runway unchanged. It then left-shifts the start times of the subsequent aircraft, which generally results in smaller start times than the initial ones. However, due to the existence of sequence-dependent separation times, this may not always be the case. The rationale behind this algorithm is to minimise the weighted start times after canceled flights without increasing the runway instability component. The procedure can be summarised as follows:

4.5 RepairBySlack algorithm

RepairBySlack is a partial repair algorithm that is applied to the runway on which an aircraft is delayed. Runway reassignment is not performed in this algorithm and aircraft that are considered for repair are the ones that are delayed and their successors on the same runway. Regarding stability, the algorithm attempts to preserve the original sequence and the initial start times on each runway. At time t, the algorithm assigns the aircraft with the minimum start time slack $\left(\bar{t}_j - t\right)$ first, which helps to reduce the difference from their initial start times. The procedure for RepairBySlack algorithm is given as follows.

4.6 RepairByEDD algorithm

RepairByEDD is a partial repair algorithm that works similar to RepairBySlack, except that it sorts the flights by deadline, and assigns the aircraft with the earliest deadline (d_j) first. Runway re-assignment is not permitted in this method. Obtaining a feasible schedule after disruptions is one of the targets of the rescheduling algorithms. Since the deadline constraint affects the schedule feasibility, this algorithm attempts to obtain a feasible schedule. Regarding stability, it attempts to preserve the initial schedule and start times by repairing only the aircraft that can potentially be affected by the disruptions. This procedure is detailed as follows:

https://doi.org/10.1017/aer.2025.16 Published online by Cambridge University Press

Algorithm 4.5 *RepairBySlack Algorithm*

1: Get initial schedule (schedule after cancelation).
2: Get the delayed aircraft j ($\forall j \in D$).
3: Determine the runway i of the delayed aircraft j.
4: **for** each delayed aircraft j and its successors on runway i **do.**
5: Set current time t as the start time of the delayed aircraft j in the current schedule.
6: **while** the deadline constraint satisfies $t < D_j$ **do.**
7: Calculate the $\left(\bar{t}_j - t\right)$.
8: Find $j = \{j \in J\text{:}\ \min_j \left(\bar{t}_j - t\right)\}$.
9: **end while.**
10: Update the start time.
11: **end for.**
12: Repeat Step 4 until $D = \{\}$.
13: Calculate the combined objective function value using Equation 9.

Algorithm 4.6 *RepairByEDD Algorithm*

1: Get initial schedule (schedule after cancelation).
2: Get all delayed aircraft ($\forall j \in D$).
3: Determine the runway i of the delayed aircraft j.
4: **for** the delayed aircraft j and its subsequent aircraft on runway i **do.**
5: **while** $t < d_j$ **do.**
6: Find $j = \left\{j \in J : \min_j \left\{d_j\right\}\right\}$ and assign j to the runway i.
7: Update the start time.
8: **end while.**
9: **end for.**
10: Repeat Step 4 until $D = \{\}$.
11: Calculate the combined objective function value using Equation 9.

4.7 InsertDelayed algorithm

InsertDelayed is a partial and right-shift repair algorithm that is considered only for the particular runway to which the delayed aircraft was initially assigned. The algorithm inserts every delayed aircraft in all possible subsequent positions on the same runway to find the best position with the minimum value of the normalised objective function. This algorithm emphasises both efficiency and stability. After inserting an aircraft into a certain position on a runway, start times of all the remaining aircraft assigned to the same runway are shifted if necessary. *InsertDelayed* attempts to keep the sequence of the aircraft as much as possible on the corresponding runway unchanged. If there are more than one delayed aircraft on the same runway, the insertion starts with the flight that is positioned earlier. The procedure is given as follows:

4.8 RepairByTWST algorithm

RepairByTWST is a partial repair algorithm that resembles RepairBySlack and RepairByEDD algorithms in the sense that it repairs the schedule rather than rescheduling from scratch, and it resembles the TWST algorithm in the sense that the affected aircraft are assigned to the runways by the largest $w_j / \left(r_j + s_{kj}\right)$ ratio. The rationale here is to give more priority to aircraft that become available earlier and require less separation times and/or with to those that have higher weight. Aircraft that are affected and considered for repair consist of newly arriving aircraft and scheduled aircraft whose start times are greater than the minimum ready times values of these newly arriving aircraft. The approach aims at preserving the initial schedule as much as possible while keeping the total weighted start times at

https://doi.org/10.1017/aer.2025.16 Published online by Cambridge University Press

Algorithm 4.7 *InsertDelayed Algorithm*

1: Get initial schedule (schedule after cancelation).
2: Get every delayed aircraft j ($\forall j \in D$).
3: Determine the position p_j for delayed aircraft j on runway i.
4: Determine the position p_{last} of the last aircraft scheduled on runway i.
5: Increase the ready time, target time and deadline of the delayed aircraft j by the amount of delay.
6: For the delayed aircraft j and its subsequent aircraft on runway i, construct an insertion set I that consists of $p_j, p_{j+1}, p_{j+2}, \ldots, p_{last}$.
7: **while** $I \neq \{\ \}$ **do.**
8: Insert the delayed aircraft into position a, $\forall a \in I$.
9: Update all aircraft start times.
10: Calculate the combined objective function value Z using Equation 9.
11: If Z value after insertion into a is better than the best objective function value so far.
12: Update the best objective function value and the corresponding schedule.
13: **end while.**
14: Display the best combined objective function value and the corresponding schedule.

Algorithm 4.8 *RepairByTWST Algorithm*

1: Get the initial schedule (schedule after delay).
2: Get the information for every new aircraft j ($\forall j \in A$).
3: **for** each new aircraft j and aircraft whose start times are greater than the minimum ready times values of the new aircraft: **do.**
4: Calculate the ratio using $\frac{w_j}{(r_j + s_{kj})}$.
5: Assign the aircraft with the largest ratio to a runway on which its operation can start earlier.
6: Remove aircraft j from set J.
7: Update the makespan of the runway to which aircraft j is assigned.
8: Repeat Step 3 until all aircraft are scheduled.
9: **end for.**
10: Calculate the combined objective function value according to Equation 9.

minimum. Hence, the algorithm focuses on both stability and efficiency simultaneously. The procedure for *RepairByTWST* is given as follows.

4.9 InsertNew algorithm

InsertNew is a partial and right-shift repair algorithm similar to the *InsertDelayed* algorithm. The algorithm tries aircraft insertion alternatives of the new aircraft to find the best insertion with the minimum increase to the normalised objective function value. The set of aircraft that are affected and considered for repair consists of the new aircraft and the aircraft whose start times are greater than the minimum ready times values of the new aircraft. *InsertNew* emphasises both efficiency and stability objectives. After inserting an aircraft into a position on a runway, start times of all subsequent aircraft assigned to that runway are updated. If there are more than one new aircraft, the insertion starts with the aircraft whose deadline is the earliest. The procedure for *InsertNew* algorithm is given as follows.

5.0 Computational Study

In order to measure the effectiveness of the reactive scheduling algorithms introduced in this paper, solutions are compared to the optimal solutions or best solution obtained within a time limit using the MILP introduced earlier. The solution quality and stability of the reactive scheduling algorithms are evaluated

https://doi.org/10.1017/aer.2025.16 Published online by Cambridge University Press

Algorithm 4.9 *InsertNew Algorithm*

1: Get initial schedule (schedule after delay).
2: Get the information for every new aircraft j ($\forall j \in A$).
3: **for** every new aircraft j and the aircraft whose start times are greater than the minimum ready times values of the new aircraft: **do.**
4: Denote the earliest position of the aircraft whose start times are greater than the minimum ready times values of the new aircraft on runway i as x_j ($\forall i \in M$).
5: Denote the last position of an aircraft on runway as x_{last}.
6: Construct an insertion set I which consists of x_j, x_{j+1}, x_{j+2},. . ., x_{last}.
7: **end for.**
8: **while** $I \neq \{\ \}$ **do.**
9: Insert the delayed aircraft into position a, $\forall a \in I$.
10: Update the start time.
11: Calculate the combined objective function value Z according to Equation 9.
12: **if** Z after the insertion into position a is better than the best objective function value so far **then.**
13: Update the best objective function value and the corresponding schedule.
14: **end if.**
15: **end while.**
16: Report the best combined objective function value and schedule.

for problems with a number of aircraft $n = 15,\ 20,\ 25$ and number of runways $m = 2,\ 3,\ 4,\ 5$. For each combination of n and m, 5 instances were generated totaling 60 problem instances available at www.SchedulingResearch.com. Then, for each algorithm, these 60 unique problem instances with 13 different scenarios of objective weight coefficient levels were solved (for (π_1, π_2,π_3) to be (0,0,1), (0,0.25,0.75), (0,0.5,0.5), (0,0.75,0.25), (0,1,0), (0.25,0,0.75), (0.25,0.75,0), (0.33,0.33,0.33), (0.5,0,0.5), (0.5,0.5,0), (0.75,0,0.25), (0.75,0.25,0), (1,0,0)), totaling 780 instances. All MILP instances were solved using CPLEX 20.1.0.0 and Gurobi 9.1 via AMPL. For the mathematical model, the experiments were carried out on a computer with Intel (R) Xeon (R) CPU. 2.60 GHz with 64.00 GB of RAM. The proposed algorithms were implemented in C, and the experiments were carried out on a computer with Intel Core 2 Duo 2.10 GHz CPU with 4.00 GB of RAM laptop.

5.1 Data generation

The rescheduling data consists of certain percentages of delayed, new, and canceled flights along with their corresponding penalty parameters. It is also assumed that a feasible initial aircraft schedule before any disruption is given. Similar to Refs [19, 20] each aircraft is characterised by its operation type (i.e, arrival or departure), weight-class (i.e, heavy, medium or light), priority (aircraft tardiness penalty), ready time, target time, deadline and sequence-dependent separation times. Regarding the time-window, it was generated based on some real data collected from international airports including Doha International Airport. Data were generated as in Ref. [18] as follows:

The specifics of data generation are as follows:

1. Aircraft operation types were randomly generated as 0 or 1 to represent an arrival or departure respectively.
2. Aircraft weight classes were randomly generated as 1, 2, 3 to represent heavy, medium or light aircraft respectively.

https://doi.org/10.1017/aer.2025.16 Published online by Cambridge University Press

3. The aircraft start time tardiness penalty (priority) w_j varies between 1 and 6 and was introduced as a function of the aircraft weight class and its operation type, where the least weight of 1 was assigned to small departures and the greatest weight of 6 was given to heavy arrivals.
4. The ready-times r_j were randomly generated using a discrete uniform distribution over the interval $(0, \gamma \frac{n}{m})$, where γ is a parameter that was randomly selected between 30 and 90.
5. Every aircraft was prescribed a time-window of 600 seconds. Therefore, deadlines d_j were calculated by $r_j + 600$.
6. Target times δ_j were calculated by $r_j + 60$.
7. The minimum separation times s_{kj} were given and range between 30 and 200 seconds depending on aircraft type (small, large or heavy), and the type of operation (landing vs. departure) as enforced by aviation authorities. We used the FAA mandated minimum separation times specified in Table 1 for all the test instances in our test-bed similar to Refs [19, 20]. It has been noted in these studies that they do not assume the triangle inequality condition holds for minimal separations.
8. The aircraft penalty cost of deviation from the initial start time α_j, $\forall j \in \bar{J} - (\mathcal{D} \cup \mathcal{E})$ were randomly generated between 1 and 5.
9. The aircraft penalty cost of deviation from the initial runway assignment β_j, $\forall j \in \bar{J} - (\mathcal{D} \cup \mathcal{E})$ were randomly generated between 5 and 10.
10. The number of canceled flights was randomly generated between 5% and 10 % of the number of aircraft.
11. The number of delayed flights was randomly generated between 10% and 40% of the number of aircraft.
12. The amount of delay was half of the difference between maximum and minimum ready time values of the delayed flight(s).
13. The number of new flights was randomly generated between 5% and 15% of the total number of aircraft.

5.2 Performance of the MILP model

We solved 780 instances after setting the maximum CPU time limit to 1800 s. Since the performance of Gurobi was better than CPLEX, we included the Gurobi results in Table 2. In some cases, the solver was unable to find the optimal solution and stopped after hitting the limit of 1800 seconds. In such cases, we compare the best solution found within this time limit to the objective function value of the algorithm. The column headings are as follows: n: number of aircraft; m: number of runways; π_i: the different scenarios of the objective weight coefficients.

Interestingly and according to Table 2, we can see that for 46% of the coefficient levels, the commercial solver requires in average more than 110s. As expected, the CPU time increases as the problem size increases, which justifies the use of heuristics especially for larger instances. In some cases, the solver was unable to find the optimal solution and stopped for hitting the limit of 1800 seconds.

5.3 Effectiveness of the repair algorithms for flight cancelation

The performance of the reactive scheduling algorithms is measured by computing the error between the normalised combined objective function value of the algorithm (f_{ALG}) and the normalised optimal solution value $\left(f_{Optimal}\right)$ as given in Equation 10. When no optimal solutions are reached within the permitted time, the quality of the solution is measured by relative difference from the best performing algorithm.

$$Error = \left(f_{ALG} - f_{Optimal}\right)/f_{Optimal} \tag{10}$$

https://doi.org/10.1017/aer.2025.16 Published online by Cambridge University Press

Table 1. *Minimum separation times (seconds)*

Departure → Departure Case			
Leading \ Following	Heavy	Large	Small
Heavy	60	90	120
Large	60	60	90
Small	60	60	60
Departure → Arrival Case			
Leading \ Following	Heavy	Large	Small
Heavy	50	53	65
Large	50	53	65
Small	50	53	65
Arrival → Departure Case			
Leading \ Following	Heavy	Large	Small
Heavy	40	40	40
Large	35	35	35
Small	30	30	30
Arrival → Arrival Case			
Leading \ Following	Heavy	Large	Small
Heavy	99	133	196
Large	74	107	131
Small	74	80	98

Reactive scheduling strategies to repair flight cancelations are evaluated under different objective weight coefficient levels (π_1, π_2, π_3). The performance values of the repair algorithms *Do-Nothing* and *Left-Shift* are compared in terms of average relative error and CPU time (in seconds) over level of factors as given in Table 3. For each algorithm, 60 unique problem instances with 13 different scenarios of objective weight coefficient levels were solved totaling 780 instances.

For a specific weight coefficient combination, the cases where an algorithm provides the smallest average relative error are highlighted in the table. When stability is more important than the solution quality (e.g., $\pi_1 = 1, \pi_2 = 0, \pi_3 = 0$), *Do-Nothing* algorithm is a preferable strategy to apply. On the other hand, when the solution quality of the schedule is more important than the conformity to the original schedule (e.g., $\pi_1 = 0, \pi_2 = 0, \pi_3 = 1$), *Left-Shift* algorithm has better mean performance than *Do-Nothing* algorithm. The CPU times (in parentheses) in seconds are indifferent and less than 0.001 second, and do not change significantly with the change in π_1, π_2 and π_3.

Before conducting statistical performance analysis, we performed the the Anderson-Darling test to confirm the normality of the average relative error and average CPU times. The performances of the response strategies are analysed statistically by *t*-test using the statistical software programme, Minitab 15.1. The statistical analysis is conducted under two conditions; schedule stability and solution quality. Therefore, the analysis is focused on the weight coefficient values π_1, π_2, π_3 where schedule stability is represented by π_1, π_2 and solution quality by π_3. The runway deviation is not considered in the response strategies for flight cancelations; so π_2 is not the leading element in this analysis. The objective weight coefficients can take values between 0 and 1, and the sum must be 1. In this paper, we conducted our analysis for 13 various combinations of the coefficients.

1. *when the schedule stability is more important (cases with $\pi_1 > \pi_3$)*

For each of the 60 problem instances, there are 5 combinations where $\pi_1 > \pi_3$ are considered, totaling 300 observations. We can conclude from Table 4 that the average relative error of *Do-Nothing* strategy is significantly less than the average relative error of the *Left-Shift* algorithm when stability is of more

https://doi.org/10.1017/aer.2025.16 Published online by Cambridge University Press

Table 2. *Computational time in seconds to solve the MILP*

	π_1	0	0	0	0	0	0.25	0.25	0.33	0.50	0.50	0.75	0.75	1.00
	π_2	0	0.25	0.50	0.75	1.00	0	0.75	0.33	0	0.50	0	0.25	0
	π_3	1.00	0.75	0.50	0.25	0	0.75	0.00	0.33	0.50	0.00	0.25	0.00	0
n	**m**													
15	2	0.60	0.77	0.62	0.53	0.05	0.35	0.23	0.23	0.25	0.24	0.36	0.24	0.26
	3	9.74	31.84	22.12	14.35	0.05	1.50	0.13	0.57	0.41	0.14	0.18	0.14	0.08
	4	5.26	133.41	128.70	18.88	0.05	1.21	0.11	0.69	0.57	0.14	0.20	0.18	0.16
	5	18.22	1498.82	1134.49	205.37	0.05	3.80	0.15	2.34	0.80	0.30	0.30	0.47	0.26
20	2	1.21	0.98	1.48	1.41	0.06	0.99	0.40	1.25	0.85	0.37	0.60	0.45	0.44
	3	13.34	30.51	15.17	7.09	0.05	2.40	0.50	1.17	0.80	0.54	0.44	0.58	0.40
	4	478.80	1289.51	1084.14	921.43	0.07	403.26	0.39	80.20	8.19	0.48	0.47	0.57	0.32
	5	839.73	1800.00	1800.00	1579.06	0.07	65.84	0.21	21.42	2.92	0.28	0.53	0.57	0.43
25	2	0.66	0.67	0.64	0.61	0.06	0.45	0.27	0.33	0.31	0.26	0.28	0.20	0.27
	3	361.85	362.33	362.59	361.97	0.09	361.33	0.88	203.00	24.66	0.90	0.76	0.87	0.58
	4	551.58	1311.14	1033.11	619.20	0.07	172.37	0.43	5.57	1.98	0.51	0.89	0.53	0.30
	5	1645.68	1800.00	1800.00	1759.01	0.09	1112.11	0.41	505.86	15.20	0.38	0.98	0.22	0.38
	Min	0.60	0.67	0.62	0.53	0.05	0.35	0.11	0.23	0.25	0.14	0.18	0.14	0.08
	Max	1645.68	1800.00	1800.00	1759.01	0.09	1112.11	0.88	505.86	24.66	0.90	0.98	0.87	0.58
	Average	**327.22**	**688.33**	**615.25**	**457.41**	**0.06**	**177.13**	**0.34**	**68.55**	**4.75**	**0.38**	**0.50**	**0.42**	**0.32**

Table 3. *Average relative error and (CPU times in seconds) of the algorithms for flight cancelations*

π_1	0	0	0	0	0	0.25	0.25	0.33	0.5	0.5	0.75	0.75	1
π_2	0	0.25	0.5	0.75	1	0	0.75	0.33	0	0.5	0	0.25	0
π_3	1	0.75	0.5	0.25	0	0.75	0	0.33	0.5	0	0.25	0	0
Do-Nothing	0.745	0.512	0.334	0.167	**0.000**	0.400	**0.000**	0.068	0.122	**0.000**	**0.020**	**0.000**	**0.000**
	(0.001)	(0.001)	(0.001)	(0.000)	(0.000)	(0.001)	(0.000)	(0.000)	(0.001)	(0.000)	(0.000)	(0.000)	(0.000)
Left-Shift	**0.628**	**0.425**	**0.275**	**0.139**	**0.000**	**0.333**	0.021	**0.058**	**0.109**	0.042	0.055	0.063	0.085
	(0.001)	(0.001)	(0.001)	(0.000)	(0.000)	(0.001)	(0.000)	(0.000)	(0.001)	(0.000)	(0.000)	(0.000)	(0.000)

Table 4. *Paired T-Test for Do-Nothing vs Left-Shift (with respect to stability)*

	N	Mean	StDev	SE Mean
Do-Nothing	300	0.00409	0.00827	0.00049
Left-Shift	300	0.05341	0.02123	0.00128
Difference	300	−0.04932	0.02242	0.00135

95% CI for mean difference: (−0.05098, −0.04677).
T-Test of mean difference = 0 (vs not = 0): *T*-Value = −36.48 *P*-Value = 0.000.

Table 5. *Paired T-Test: Do-Nothing vs Left-Shift (quality)*

	N	Mean	StDev	SE Mean
Do-Nothing	480	0.2938	0.2366	0.0113
Left-Shift	480	0.2466	0.1987	0.0095
Difference	480	0.0472	0.0378	0.0018

95% CI for mean difference: (0.04353, 0.05042)
T-Test of mean difference = 0 (vs not = 0): *T*-Value = 26.06 *P*-Value = 0.000.

concern since the *p*-value for the difference between them is very small compared to $\alpha = 0.05$. This means that *Do-Nothing* is better to apply in this case.

2. *when the solution quality is more important (cases where $\pi_1 \leq \pi_3$)*

For each of the 60 problem instances, there are 8 combinations where $\pi_1 \leq \pi_3$ are considered, totaling 480 observations. Conducting similar statistical analysis like the previous case, Table 5 shows that the average relative error of Left-Shift strategy is statistically significantly less ($p - value = 0.000 < \alpha = 0.05$) than the average relative error of the Do-Nothing algorithm when solution quality is of more concern, which means the Left-Shift is better to apply in this case.

To summarise, if the weight of the schedule stability is higher than the weight of the solution quality in the objective function, Do-Nothing is preferred to Left-Shift; otherwise, Left-Shift is more preferable.

In practice, the first-come-first-served (FCFS) rule, is the most widely used heuristic in terminal areas for the aircraft sequencing where aircraft are assigned to the runways in increasing order of their ready times. The target of the FCFS is to minimise the delay by minimising makespan or minimising maximum lateness. To show that the two algorithm are different, we have implemented the FCFS heuristic and compared it with Do-Nothing algorithm. In the Do-Nothing algorithm, if there is a flight cancelation, no corrective action is taken for the given initial schedule i.e., remaining aircraft assigned to that runway. Therefore, Do-Nothing keeps the initial runway assignments, flights sequence on each runway and start times as they are. Therefore, our aim is to maximise the throughput of the runways and minimise deviation from the initial given schedule. On the other hand, in the FCFS rule, the target of the FCFS is to minimise the delay by minimising makespan or minimising maximum lateness. For our paper, the only case Do-Nothing algorithm is equivalent to FCFS could be that if the initial schedule (before disruptions) is constructed based on increasing order of ready time for aircraft j to take-off or land. Table 6 shows that even though computational time performance of them are similar to each other, Do-Nothing algorithm beats FCFS rule in terms of objective function values (i.e., average relative error).

5.4 Effectiveness of the repair algorithms for flight delay

Due to the sequential evaluation methodology developed to treat disruptions, once the performance analysis of the repair algorithms for flight cancelations are conducted and the revised schedule is updated as the current schedule, the performance of the repair algorithms for flight delays are evaluated next. The proposed algorithms are tested under various flight delays and objective weight coefficient levels.

https://doi.org/10.1017/aer.2025.16 Published online by Cambridge University Press

Table 6. *Comparison of Do-Nothing and FCFS algorithms in terms of average relative error and (CPU times in seconds)*

π_1	0	0	0	0	0	0.25	0.25	0.33	0.5	0.5	0.75	0.75	1
π_2	0	0.25	0.5	0.75	1	0	0.75	0.33	0	0.5	0	0.25	0
π_3	1	0.75	0.5	0.25	0	0.75	0	0.33	0.5	0	0.25	0	0
Do-Nothing	0.745	0.512	0.334	0.167	**0.000**	0.400	**0.000**	0.068	0.122	**0.000**	**0.020**	**0.000**	**0.000**
	(0.001)	(0.001)	(0.001)	(0.000)	(0.000)	(0.001)	(0.000)	(0.000)	(0.001)	(0.000)	(0.000)	(0.000)	(0.000)
FCFS	0.870	0.598	0.390	0.195	**0.000**	0.467	0.025	0.079	0.143	0.049	0.064	0.074	0.099
	(0.001)	(0.001)	(0.001)	(0.000)	(0.000)	(0.001)	(0.000)	(0.000)	(0.001)	(0.000)	(0.000)	(0.000)	(0.000)

The performances of the RepairBySlack, RepairByEDD and InsertDelayed are compared in terms of the average relative error and CPU times. Equation 10 is used to calculate the error for each test problem.

Comparison results of the repair algorithms for flight delays are provided in Table 7 given that Left-Shift algorithm is applied to update the schedule after flight cancelations. Following similar statistical analysis like before, it is shown that there is a significant difference ($p - value = 0.000 < \alpha = 0.05$) between the algorithms' performance in terms of average relative error where:

1. when the schedule stability is more important, InsertedDelayed performs best, followed by RepairBySlack, and finally followed by RepairByEDD.
2. when the solution quality is more important, RepairByEDD performs best, followed by InsertDelayed and finally followed by RepairBySlack.

Comparison results of the repair algorithms for flight delays given that Do-Nothing algorithm is applied after flight cancelations are provided in Table 8. When the solution quality is more important than the stability, RepairByEDD algorithm is the best strategy to apply. Conversely, when the stability of the schedule is more important, RepairBySlack algorithm provides the lowest average relative error. When the weight coefficient values of all three objective function components are equal, InsertDelayed algorithm performs best.

5.5 Effectiveness of the repair algorithms for arrival of new flight

The repair algorithms evaluated here are RepairByTWST and InsertNew under two scenarios of multiple disruptions: first, assuming that Left-Shift algorithm is applied to flight cancelation and repaired by RepairByEDD for flight delays. Second, assuming that Do-Nothing algorithm is applied for flight cancelation and repaired by RepairBySlack for flight delays. In fact there are more algorithm combinations that can be considered; however, we observed earlier that Left-Shift and RepairByEDD algorithms outperformed the other combinations when the solution quality was more important. On the other hand, Do-Nothing and RepairBySlack algorithms performed better when the schedule stability was more important. The overall results of both scenarios are summarised in Tables 8 and 9, respectively.

Applying statistical testing to the results in Table 9 shows that there is a significant difference ($p - value = 0.000 < \alpha = 0.05$) between the mean values of average relative error for the algorithms where:

1. *when the schedule stability is more important ($\pi_1 > \pi_3, \pi_2 > \pi_3$), RepairByTWST has better mean performance than InsertNew algorithm.*
2. *when the solution quality is more important ($\pi_1 \leq \pi_3, \pi_2 \leq \pi_3$), RepairByTWST has better mean performance than InsertNew algorithm.*

Similar statistical analysis to the results in Table 10 leads to conclude that:

1. *when the schedule stability is more important ($\pi_1 > \pi_3, \pi_2 > \pi_3$), InsertNew outperforms the RepairByTWST.*
2. *when the solution quality is more important ($\pi_1 \leq \pi_3, \pi_2 \leq \pi_3$), RepairByTWST has better mean performance than InsertNew algorithm.*

Inspecting Tables 9 and 10 for the best average relative error values and comparing them for each π_1, π_2, π_3 combination, we can see that Do-Nothing, RepairBySlack and InsertNew combination provides the lowest average relative error when the schedule stability is more important ($\pi_1 > \pi_3$), while Left-Shift, RepairByEDD, RepairByTWST combination performs better when the solution quality is more important ($\pi_1 < \pi_3$).

https://doi.org/10.1017/aer.2025.16 Published online by Cambridge University Press

Table 7. *Average relative errors and (CPU times in seconds) of the algorithms for flight delays if the Left-Shift algorithm is applied for flight cancelations*

π_1	0	0	0	0	0	0.25	0.25	0.33	0.5	0.5	0.75	0.75	1
π_2	0	0.25	0.5	0.75	1	0	0.75	0.33	0	0.5	0	0.25	0
π_3	1	0.75	0.5	0.25	0	0.75	0	0.33	0.5	0	0.25	0	0
RepairByEDD	**0.643**	**0.431**	**0.275**	**0.144**	(0.000)	**0.534**	0.056	**0.173**	**0.297**	0.112	0.192	0.168	0.224
	(0.001)	(0.001)	(0.001)	(0.000)	(0.000)	(0.001)	(0.001)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)
RepairBySlack	0.877	0.608	0.387	0.199	(0.000)	0.687	**0.040**	0.226	0.379	**0.080**	0.200	**0.121**	**0.161**
	(0.001)	(0.001)	(0.001)	(0.000)	(0.000)	(0.001)	(0.001)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)
InsertDelayed	0.745	0.509	0.322	0.166	(0.000)	0.591	0.042	0.184	0.317	0.084	**0.172**	0.126	0.168
	(0.002)	(0.004)	(0.004)	(0.002)	(0.002)	(0.004)	(0.002)	(0.002)	(0.004)	(0.002)	(0.002)	(0.002)	(0.002)

Table 8. *Average relative error and (CPU times in seconds) of the algorithms for flight delays when Do-Nothing algorithm is applied for flight cancelations*

π_1	0	0	0	0	0	0.25	0.25	0.33	0.5	0.5	0.75	0.75	1
π_2	0	0.25	0.5	0.75	1	0	0.75	0.33	0	0.5	0	0.25	0
π_3	1	0.75	0.5	0.25	0	0.75	0	0.33	0.5	0	0.25	0	0
RepairByEDD	**0.672**	**0.510**	**0.345**	**0.173**	**(0.000)**	**0.575**	0.057	**0.305**	0.460	0.114	0.344	0.170	0.227
	(0.001)	(0.001)	(0.001)	(0.000)	(0.000)	(0.001)	(0.001)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)
RepairBySlack	0.916	0.687	0.456	0.229	**(0.000)**	0.702	**0.015**	0.326	0.489	**0.031**	0.275	**0.046**	**0.061**
	(0.001)	(0.001)	(0.001)	(0.000)	(0.000)	(0.001)	(0.001)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)
InsertDelayed	0.784	0.588	0.392	0.196	**(0.000)**	0.609	0.021	**0.290**	**0.435**	0.043	**0.260**	0.064	0.086
	(0.002)	(0.004)	(0.004)	(0.002)	(0.002)	(0.004)	(0.002)	(0.002)	(0.004)	(0.002)	(0.002)	(0.002)	(0.002)

Table 9. *Average relative error and (CPU times in seconds) of the algorithms for new flight arrival with Left-Shift for cancelations and RepairByEDD for delays*

π_1	0	0	0	0	0	0.25	0.25	0.33	0.5	0.5	0.75	0.75	1
π_2	0	0.25	0.5	0.75	1	0	0.75	0.33	0	0.5	0	0.25	0
π_3	1	0.75	0.5	0.25	0	0.75	0	0.33	0.5	0	0.25	0	0
RepairByTWST	**0.022**	**0.137**	**0.254**	**0.369**	**0.485**	**0.069**	**0.421**	**0.244**	0.162	**0.357**	0.226	**0.293**	**0.229**
	(0.001)	(0.001)	(0.001)	(0.000)	(0.000)	(0.001)	(0.001)	(0.000)	(0.001)	(0.000)	(0.000)	(0.000)	(0.000)
InsertNew	0.034	0.209	0.381	0.556	0.730	0.098	0.620	0.351	**0.125**	0.510	**0.176**	0.398	0.289
	(0.002)	(0.004)	(0.004)	(0.002)	(0.002)	(0.004)	(0.002)	(0.002)	(0.004)	(0.002)	(0.002)	(0.002)	(0.002)

Table 10. *Average relative error and (CPU times in seconds) of the algorithms for new flight arrival with the Do-Nothing for flight cancelations and the RepairBySlack for delays*

π_1	0	0	0	0	0	0.25	0.25	0.33	0.5	0.5	0.75	0.75	1
π_2	0	0.25	0.5	0.75	1	0	0.75	0.33	0	0.5	0	0.25	0
π_3	1	0.75	0.5	0.25	0	0.75	0	0.33	0.5	0	0.25	0	0
RepairByTWST	**0.140**	**0.258**	**0.366**	**0.426**	**0.485**	**0.156**	0.506	**0.283**	**0.171**	0.405	0.177	0.304	0.203
	(0.001)	(0.001)	(0.001)	(0.000)	(0.000)	(0.001)	(0.001)	(0.000)	(0.001)	(0.000)	(0.000)	(0.000)	(0.000)
InsertNew	0.248	0.308	0.373	0.491	0.606	0.215	**0.391**	0.317	0.182	**0.300**	**0.147**	**0.207**	**0.115**
	(0.002)	(0.004)	(0.004)	(0.002)	(0.002)	(0.004)	(0.002)	(0.002)	(0.004)	(0.002)	(0.002)	(0.002)	(0.002)

Table 11. *Average relative errors and (CPU times in seconds) of the complete regeneration algorithms*

π_1	0	0	0	0	0	0.25	0.25	0.33	0.5	0.5	0.75	0.75	1
π_2	0	0.25	0.5	0.75	1	0	0.75	0.33	0	0.5	0	0.25	0
π_3	1	0.75	0.5	0.25	0	0.75	0	0.33	0.5	0	0.25	0	0
TWST	0.165	0.141	0.126	0.226	0.141	0.154	0.474	0.446	0.474	0.452	0.181	0.212	0.127
	(0.048)	(0.046)	(0.047)	(0.047)	(0.047)	(0.047)	(0.047)	(0.047)	(0.048)	(0.047)	(0.048)	(0.047)	(0.047)
SA-Re	**0.049**	**0.071**	**0.084**	**0.094**	**0.071**	**0.072**	**0.089**	**0.071**	**0.088**	**0.086**	**0.081**	**0.089**	**0.083**
	(21.300)	(22.820)	(21.930)	(21.440)	(21.670)	(21.760)	(21.880)	(21.560)	(22.120)	(22.870)	(22.350)	(21.980)	(21.890)

5.6 Effectiveness of the complete regeneration algorithms

Complete rescheduling algorithms introduced in this paper (SA-Re and TWST algorithms) consider cancelations, delays and new arrivals simultaneously. The proposed algorithms are tested under various disruptive events and objective weight coefficient levels. Table 11 provides the average relative error and CPU times in seconds (in parentheses) of the complete regeneration algorithms for different problem instances. The results show that regardless of the π_1, π_2, π_3 combination, SA-Re performs better than the TWST algorithm in terms of average relative error but it requires longer CPU.

When the average relative error performances of repair algorithms and complete regeneration algorithms after disruptions are compared utilising Tables 9–11, for each combination of π_1, π_2, π_3, the results of SA-Re outperforms the rest.

Table 12 summarises the main findings in terms of average relative error and provides a road map for the most effective solution approach to use with respect to each disturbance type. We follow a sequential evaluation method in which we start with flight cancelation followed by flight delays and finally the arrival of new unexpected flights. With respect to flight cancelations, when schedule stability is of a higher priority, the Do-Nothing algorithm is preferred; while, when the solution quality is higher priority, the Left-Shift algorithm is preferred. Regarding flight delays, when schedule stability has a higher priority, RepairBySlack performs best when Do-Nothing is used for cancelations. InsertDelayed performs best when Left-Shift is used for cancelations. On the other hand, when solution quality is more important, RepairByEDD performs best when Left-Shift or Do-Nothing are used with flight cancelations. As for new flight arrivals, and when schedule stability is of has higher priority, RepairByTWST performs best when either Do-Nothing is used for cancelations and RepairBySlack for delays, or when Left-Shift is used for flight cancelations and RepairByEDD is used for delays. On the other hand, when the solution quality is more important, InsertNew performs best when Do-Nothing is used for cancelations and RepairBySlack for delays; while RepairByTWST algorithm performs best when Left-Shift is used for flight cancelations and RepairByEDD is used for delays. Finally, when it comes to rescheduling from scratch to treat flight cancelations, delays and unexpected arrivals simultaneously regardless of stability and quality priorities, complete regeneration using SA-Re performs best.

6.0 Conclusions and Future Research

Throughout the course of daily operations, air traffic systems frequently encounter various disruptions because of the dynamic environment and unexpected events such as severe weather, aircraft failures or personnel shortages. Therefore, the initial plan may not be executed as designed. This paper addressed the aircraft reactive scheduling problem to update the existing aircraft schedule dynamically. The research considers disruptions including the arrival of new aircraft, flight cancelations and aircraft delays. The objectives consist of minimising the total weighted start times (solution quality), total weighted start time deviation and total weighted runway deviation (instability measures). Our approach is to sequence the aircraft before they go to the holding area while it's still practical to change their positions. The research that addresses multi-objective aircraft rescheduling problem with multiple disruption types is very limited. To fill this research gap, this paper extends a mixed integer linear programming model from the scheduling domain to the rescheduling domain to find optimal solutions. ARSP is formulated as a multi-objective optimisation problem in which both the schedule's quality and stability are of interest. For instances that are too difficult to solve optimally using the MILP within the permitted time, the paper proposes approximate algorithms to obtain near-optimal schedules efficiently. Therefore, sequential evaluation methodology is developed to treat the disruptions and revise the schedules periodically. The performance of the proposed algorithms depend on priority given to the schedule's quality versus stability as well as on the disruption type(s) encountered. Accordingly, the proposed framework can dynamically invoke the best performing algorithms. Repair and complete regeneration approximate algorithms are developed for each type of disruptive events. Do-Nothing and Left-Shift are the repair strategies for the flight cancelations, RepairBySlack, RepairByEDD, InsertDelayed algorithms

https://doi.org/10.1017/aer.2025.16 Published online by Cambridge University Press

Table 12. *Algorithm summary*

		Flight Cancelation	
Repair Algorithms	**Solution Quality**	Left-Shift	
	Schedule Stability	Do-Nothing	
		Flight Delay	
		Flight Delay-Start with Do-Nothing	**Flight Delay-Start with Left-Shift**
	Solution Quality	RepairByEDD	RepairByEDD
	Schedule Stability	RepaiBySlack	InsertDelayed
		Arrival of New Flight	
		Arrival of New Flight-Start with Do-Nothing and RepairBySlack	**Arrival of New Flight-Start with Left-Shift and RepairByEDD**
	Solution Quality	InsertNew	RepairByTWST
	Schedule Stability	RepairByTWST	RepairByTWST
		Flight Cancelation, Flight Delay, Arrival of New Flight	
Complete Regeneration Algorithms	**Solution Quality**	SA-Re	
	Schedule Stability	SA-Re	

are proposed to repair the schedule for flight delays, and RepairByTWST and InsertNew are the repair algorithms for the arrival of new aircraft. Two complete regeneration algorithms, TWST algorithm and SA-Re algorithm are proposed to regenerate schedules from scratch to treat flight cancelations, delays and unexpected arrivals simultaneously. All algorithms were tested against difficult benchmark problems and the solutions were compared to optimal solutions and to each other in terms of solution quality, schedule stability and computational time. Using data pertaining to Doha International Airport (DOH) and a test-bed of simulated instances, our computational results and findings highlight the usefulness of the proposed mathematical formulation and the accompanying heuristic solution procedures. We also scrutinise the practicality of the proposed algorithms by examining the average relative error and CPU times from the base solution produced by the traditional FCFS rule.

A computational study was conducted for the three disruptive event types of ARSP with various values of objective weight coefficients π_1, π_2, π_3. Initially, response strategies to repair flight cancelation disruptions were evaluated.

It was statistically illustrated that when the stability objective has a higher importance $\pi_1 > \pi_3$, Do-Nothing algorithm is preferred. On the other hand, Left-Shift algorithm has significantly performed better when $\pi_1 \leq \pi_3$. Secondly, the repair algorithms for flight delays were tested. Depending on the initially selected strategy in the flight cancelation (i.e. Do-Nothing vs. Left-Shift) and the decision maker's interest in solution quality and stability, the performance of the repair algorithms was evaluated. When the solution quality is more important, RepairByEDD algorithm outperforms RepairBySlack and InsertDelayed. Conversely, InsertDelayed or RepairBySlack can be preferred depending on the repair algorithm used for cancelations when the schedule stability has a higher importance. Then, RepairByTWST and InsertNew repair algorithms were compared when Left-Shift strategy is used for flight cancelation, and RepairByEDD is used for delays. Depending on the importance of solution quality vs. stability, the performance of the repair algorithms was evaluated; when either the solution quality or schedule stability has higher importance, RepairByTWST outperforms the InsertNew. On the other hand, when the initially selected strategy in the flight cancelation is Do-Nothing, and in the flight delay is RepairBySlack, it was statistically shown that InsertNew is a better choice than RepairBySlack when the schedule stability is more important. Finally, the performances of the complete regeneration algorithms were tested. Although SA-Re requires longer CPU time than TWST, it is illustrated that either when the solution quality or schedule stability has higher importance, SA-Re performs better than TWST algorithm in terms of average error. Moreover, the average CPU time does not seem to change significantly with the change in π_1, π_2, and π_3.

This paper has the following contributions: first, the problem is modeled under a mixed mode of operations where both landing and departure flows are considered simultaneously, contrary to most existing studies that treat departures as separate from landings (i.e. segregated mode). It is to be noted that aircraft landing problems have received greater attention than aircraft departure problems because of their critical nature, and few studies have addressed the joint problem of sequencing both aircraft landings and departures (Refs [3, 6, 25, 92]). Note that the wake vortex separation requirements for departures-only or arrivals-only operations usually satisfy the triangular inequality. However, the triangle inequality does not necessarily hold when both arrivals and departures are scheduled simultaneously. In other words, the separation times do not automatically ensure proper separation between any pair of aircraft having the same operation type that are interspersed with an aircraft operation of the opposite type (e.g. two landings separated by a departure or two departures interspersed with a landing) as was emphasised by several researchers (e.g., Refs [3, 19, 20, 28]). This means that models and algorithms that rely on the triangular inequality in the segregated mode may not automatically be used in the mixed mode making the problem more complex from that perspective. Balakrishnan and Chandran [3] showed that accommodating the triangle inequality adds a complexity to an equivalent problem in which the separation requirements obey the triangle inequality. Moreover according to [92], runway scheduling has three mathematical models of increasing complexity: RSP1 allows only landings or only takeoffs on a single runway; RSP2 allows mixed takeoffs and landings on a single runway; and RSP3 allows multiple runways. Due to the specific separation times, as provided in Table 1, used in this paper, which are

https://doi.org/10.1017/aer.2025.16 Published online by Cambridge University Press

similar to those in Refs [3, 20, 28] it is necessary to ensure the separation of an aircraft between at most four consecutive aircraft which makes our problem more difficult.

Second, the problem of scheduling aircraft arrivals and departures over multiple runways is examined which is much more difficult than single runway problem. When a single runway is considered, although still NP-hard, one has merely to determine the sequence of the aircraft allocated to a runway. On the other hand, scheduling over multiple runways is a two-step process; first, one has to determine the assignment of aircraft to runways, then the sequence of the aircraft on each runway. The features of the problem; unequal ready time, target time, deadline, sequence-dependent separation time, multi-resourced, single and multi-objective structure of the problem is unique, which makes the contribution of the paper significant in our opinion.

Third, the research that address the multi-objective optimisation in aircraft reactive scheduling that deals with flight related disruptions is very limited. To fill this research gap, this paper updated mixed integer linear programming with normalised objective function to find optimal solutions, and proposed approximate algorithms to obtain near-optimal schedules efficiently. The trade-off between the objectives are evaluated; the components of the multi-objective function are the total weighted start time which represents solution quality, and the total weighted start time deviation and total weighted runway deviation which represent solution stability.

Fourth, unlike most studies in the literature, which focus on one disruption type at a time, different types of disruptions with multiple disruptive events are considered simultaneously in this paper. Therefore, the sequential evaluation methodology is developed to treat the disruptions and revise the schedules periodically. Alternative reactive scheduling approaches (Do-Nothing, Left-Shift, RepairByEDD, RepairBySlack, InsertDelayed, RepairByTWST, InsertNew, TWST and SA-Re) for different disruptions are proposed in which the model itself dynamically select the most appropriate from several candidate solution methods with respect to (conflicting) objectives of quality and stability.

The problem can be extended in the future to address the following aspects: firstly, the disruption of the schedules affects the capacity of the airport, causes passenger dissatisfaction and imposes substantial costs. In addition to maintaining conformity to the initial schedule, the convenience of the passengers who have connecting flights can be taken into account. Secondly, case studies and real data analyses based on airport data from around the world would be interesting. Thirdly, the work can be generalised for more complex aviation regulations such as stochastic time-windows (i.e. uncertain ready time, target time, deadline, etc.). Fourthly, reactive scheduling problem can be extended by considering more disruptive events such as runway closure. Fifthly, the problem can be revised in such a way that the impacts of the operational settings (i.e. using the proposed algorithms) on the fuel cost savings and greenhouse gas emission be examined empirically. Lastly, time-indexed formulations may indeed be an efficient way to solve the problem and we can include this as a future research.

Acknowledgements. This publication was made possible by an NPRP award [NPRP09-253-2-103] from the Qatar National Research Fund (a member of the Qatar Foundation). The statements made herein are solely the responsibility of the authors.

References

[1] Abela, J., Abramson, D., Krishnamoorthy, M., De Silva, A. and Mills, G. Computing optimal schedules for landing aircraft., in *Proceedings of the 12th National Conference of the Australian Society for Operations Research*, Adelaide, 1993, pp 71–90.

[2] Avella, P. et al. Time-indexed formulations for the runway scheduling problem, *Transport. Sci.*, 2017, **51**, (4), pp 1196–1209.

[3] Balakrishnan, H. and Chandran, B.G. Algorithms for scheduling runway operations under constrained position shifting, *Oper. Res.*, 2010, **58**, (6), pp 1650–1665.

[4] Beasley, J.E. et al. Displacement problem and dynamically scheduling aircraft landings, *J. Oper. Res. Soc.*, (2004), **55**, (1), pp 54–64.

[5] Beasley, J.E. et al. Scheduling aircraft landings-the static case, *Transport. Sci.*, 2000, **34**, (2), pp 180–197.

[6] Bianco, L., DellOlmo, P. and Giordani, S. Scheduling models for air traffic control in terminal areas, *J. Sched.*, 2006, **9**, (3), pp 223–253.

[7] Cecen, R.K. A stochastic programming model for the aircraft sequencing and scheduling problem considering flight duration uncertainties, *Aeronaut. J.*, 2022, **126**, (1304), pp 1736–1751.

https://doi.org/10.1017/aer.2025.16 Published online by Cambridge University Press

[8] Çeçen, R.K., Cetek, C. and Kaya, O. Aircraft sequencing and scheduling in TMAs under wind direction uncertainties, *Aeronaut. J.*, 2020, **124**, (1282), pp 1896–1912.
[9] D'Ariano, A., Pacciarelli, D., Pistelli, M. and Pranzo, M. Real-time scheduling of aircraft arrivals and departures in a terminal maneuvering area, *Networks*, 2015, **65**, (3), pp 212–227.
[10] De Maere, G., Atkin, J.A.D. and Burke, E.K. Pruning rules for optimal runway sequencing, *Transport. Sci.*, 2018, **52**, (4), pp 898–916.
[11] Dönmez, K. Aircraft sequencing under the uncertainty of the runway occupancy times of arrivals during the backtrack procedure, *Aeronaut. J.*, 2023, **127**, (1310), pp 562–580.
[12] Dönmez, K., Çetek, C. and Kaya, O. Aircraft sequencing and scheduling in parallel-point merge systems for multiple parallel runways, *Transport. Res. Record*, 2022, **2676**, (3), pp 108–124.
[13] Ernst, A.T., Krishnamoorthy, M. and Storer, R.H. Heuristic and exact algorithms for scheduling aircraft landings, *Networks*, 1999, **34**, (3), pp 229–241.
[14] Güven, A., Cetek, F.A. and Cecen, R.K. Runway assignment optimisation model for Istanbul Airport considering multiple parallel runway operations, *Aeronaut. J.*, 2024, pp 1–16.
[15] Kaplan, Z., Çetek, C. and Saraç, T.U.B.A. A multi-objective nonlinear integer programming model for mixed runway operations within the TMAs, *Aeronaut. J.*, 2024, pp 1–31.
[16] Riahi, V. et al. Constraint guided search for aircraft sequencing, *Exp. Syst. Appl.*, 2019, **118**, pp 440–458.
[17] Salehipour, A., Modarres, M. and Naeni, L.M. An efficient hybrid meta-heuristic for aircraft landing problem, *Comput. Oper. Res.*, 2013, **40**, (1), pp 207–213.
[18] Ghoniem, A. and Farhadi, F. A column generation approach for aircraft sequencing problems: a computational study, *J. Oper. Res. Soc.*, 2015, **66**, pp 1717–1729.
[19] Ghoniem, A., Sherali, H.D. and Baik, H. Enhanced models for a mixed arrival-departure aircraft sequencing problem, INFORMS J. Comput., 2014, 26, (3), pp 514–530.
[20] Hancerliogullari, G. et al. Greedy algorithms and metaheuristics for a multiple runway combined arrival-departure aircraft sequencing problem, *J. Air Transport Manag.*, 2013, **32**, pp 39–48.
[21] Lawler, E.L., Lenstra, J.K. and Kan, A.H.G.R. Recent developments in deterministic sequencing and scheduling: a survey, in *Deterministic and Stochastic Scheduling: Proceedings of the NATO Advanced Study and Research Institute on Theoretical Approaches to Scheduling Problems held in Durham, England, July 6–17, 1981*. Dordrecht: Springer Netherlands, 1982, pp. 35–73.
[22] Pinedo, M. *Scheduling: Theory, Algorithms, and Systems Multi-Coloring*, 2008.
[23] FAA. Traffic management overview, *US Department of Transportation*, 2020.
[24] FAA. What is NextGen? *US Department of Transportation*, 2020, https://www.faa.gov/nextgen#:~:text=Through%20NextGen%2C%20the%20FAA%20has,and%20resiliency%20of%20U.S.%20aviation
[25] Furini, F. et al. Improved rolling horizon approaches to the aircraft sequencing problem, J. Sched., 2015, 18, (5), pp 435–447.
[26] International Civil Aviation Organization. *Procedures for Air Navigation Services: Air Traffic Management*, International Civil Aviation Organization, 2020.
[27] FAA Advisory Circular. Circular 90–23G, Aircr. Wake Turbulence, 2014, https://www.faa.gov/regulations_policies/advisory_circulars/index.cfm/go/document.information/documentid/1023467
[28] Sherali, H.D. et al. A combined arrival-departure aircraft sequencing problem, in *Manuscript, Grado Department of Industrial and Systems Engineering (0118)*, Virginia Polytechnic Institute and State University, 250, 2010.
[29] Levin, A. Airline glitches top cause of delays, *USA Today*, 2007.
[30] Bureau of Transportation Statistics. Understanding the reporting of causes of flight delays and cancellations, US Department of Transportation, 2020.
[31] Pardo, J. *Unexpected Arrival*. Aircraft Owners and Pilots Association, 2006.
[32] FAA. FAA Order JO 7110.65Y – Air Traffic Control, *US Department of Transportation*, 2020.
[33] Bennell, J.A., Mesgarpour, M. and Potts, C.N. Airport runway scheduling, *Ann. Oper. Res.*, 2013, **204**, (1), pp 249–270.
[34] Ikli, S. et al. The aircraft runway scheduling problem: a survey, *Comput. Oper. Res.*, 2021, p 105336.
[35] Messaoud, M.B. A thorough review of aircraft landing operation from practical and theoretical standpoints at an airport which may include a single or multiple runways, *Appl. Soft Comput.*, 2020, p 106853.
[36] Vieira, G.E., Herrmann, J.W. and Lin, E. Rescheduling manufacturing systems: a framework of strategies, policies, and methods, *J. Sched.*, 2003, **6**, (1), pp 39–62.
[37] Bean, J.C. et al. Matchup scheduling with multiple resources, release dates and disruptions, *Oper. Res.*, 1991, **39**, (3), pp 470–483.
[38] Cheng, S.-C. et al. Dynamic hard-real-time scheduling using genetic algorithm for multiprocessor task with resource and timing constraints, *Exp. Syst. Appl.*, 2009, **36**, (1), pp 852–860.
[39] Church, L.K. and Uzsoy, R. Analysis of periodic and event-driven rescheduling policies in dynamic shops, *Int. J. Comput. Integr. Manuf.*, 1992, **5**, (3), pp 153–163.
[40] Duenas, A. and Petrovic, D. An approach to predictive-reactive scheduling of parallel machines subject to disruptions, *Ann. Oper. Res.*, 2008, **159**, (1), pp 65–82.
[41] Vieira, G.E., Herrmann, J.W. and Lin, E. Predicting the performance of rescheduling strategies for parallel machine systems, *J. Manuf. Syst.*, 2000, **19**, (4), pp 256–266.
[42] Yang, M.-H., Chung, S.-H. and Kao, C.-K. A comparative study to minimize the makespan of parallel-machine problem with job arrival in uncertainty, in *IIE Annual Conference Proceedings*, Institute of Industrial Engineers-Publisher, 2006, p 1.

https://doi.org/10.1017/aer.2025.16 Published online by Cambridge University Press

[43] Jain, A.K. and Elmaraghy, H.A. Production scheduling/rescheduling in flexible manufacturing, *Int. J. Prod. Res.*, 1997, **35**, (1), pp 281–309.
[44] Wang, S.-J., Xi, L.-F. and Zhou, B.-H. Filtered-beam-search-based algorithm for dynamic rescheduling in FMS, *Robot. Comput.-Integr. Manuf.*, 2007, **23**, (4), pp 457–468.
[45] Subramaniam, V. and Raheja, A.S. mAOR: a heuristicbased reactive repair mechanism for job shop schedules, *Int. J. Adv. Manuf. Technol.*, 2003, **22**, (9–10), pp 669–680.
[46] Yin, Y., Cheng, T.C.E. and Wang, D.-J. Rescheduling on identical parallel machines with machine disruptions to minimize total completion time, *Eur. J. Oper. Res.*, 2016, **252**, (3), pp 737–749.
[47] Yin, Y., Wang, Y., Cheng, T.C.E., Liu, W. and Li, J. Parallel-machine scheduling of deteriorating jobs with potential machine disruptions, *Omega*, 2017, **69**, pp 17–28.
[48] Alagöz, O. and Azizolu, M. Rescheduling of identical parallel machines under machine eligibility constraints, *Eur. J. Oper. Res.*, 2003, **149**, (3), pp 523–532 .
[49] Arnaout, J.-P. and Rabadi, G. Rescheduling of unrelated parallel machines under machine breakdowns, *Int. J. Appl. Manag. Sci.*, 2008, **1**, (1), pp 75–89.
[50] Curry, J. and Peters, B. Rescheduling parallel machines with stepwise increasing tardiness and machine assignment stability objectives, *Int. J. Prod. Res.*, 2005, **43**, (15), pp 3231–3246.
[51] Abumaizar, R.J. and Svestka, J.A. Rescheduling job shops under random disruptions, *Int. J. Prod. Res.*, 1997, **35**, (7), pp 2065–2082.
[52] Akturk, M.S. and Gorgulu, E. Match-up scheduling under a machine breakdown, *Eur. J. Oper. Res.*, 1999, **112**, (1), pp 81–97.
[53] Itayef, A.B., Loukil, T. and Teghem, J. Rescheduling a permutation flowshop problems under the arrival a new set of jobs, in *Computers & Industrial Engineering, 2009. CIE 2009. International Conference on*, IEEE, 2009, pp 188–192.
[54] Mason, S.J., Jin, S. and Wessels, C.M. Rescheduling strategies for minimizing total weighted tardiness in complex job shops, *Int. J. Prod. Res.*, 2004, **42**, (3), pp 613–628.
[55] Raheja, A.S. and Subramaniam, V. Reactive recovery of job shop schedules–a review, *Int. J. Adv. Manuf. Technol.*, 2002, **19**, (10), pp 756–763.
[56] Salido, M.A. et al. Rescheduling in job-shop problems for sustainable manufacturing systems, *J. Clean. Prod.*, 2017, **162**, S121–S132.
[57] Subramaniam, V., Raheja, A.S. and Reddy, K.R.B. Reactive repair tool for job shop schedules, *Int. J. Prod. Res.*, 2005, **43**, (1), pp 1–23.
[58] Zhao, Q. and Yuan, J. Rescheduling to minimize the maximum lateness under the sequence disruptions of original jobs, *Asia-Pac. J. Oper. Res.*, 2017, **34**, (05), p 1750024.
[59] Hu, Y. et al. Review of optimization problems, models and methods for airline disruption management from 2010 to 2024, *Digital Transport. Saf.*, 2024, **3**, (4), pp 1–18.
[60] Santana, M. et al. The aircraft recovery problem: a systematic literature review, *EURO J. Transport. Logist.*, 2023, p 100117.
[61] Hassan, L.K., Santos, B.F. and Vink, J. Airline disruption management: a literature review and practical challenges, *Comput. Oper. Res.*, 2021, **127**, pp 1–17.
[62] Filar, J.A., Manyem, P. and White, K. How airlines and airports recover from schedule perturbations: a survey, *Ann. Oper. Res.*, 2001, **108**, pp 315–333.
[63] Clausen, J. et al. Disruption management-operations research between planning and execution, *Or/ms Today*, 2001, **28**, (5), pp 40–43.
[64] Scozzaro, G. et al. An ILP approach for tactical flight rescheduling during airport access mode disruptions, *Int. Trans. Oper. Res.*, 2024.
[65] Luo, S. and Yu, G. On the airline schedule perturbation problem caused by the ground delay program, *Transport. Sci.*, 1997, **31**, (4), pp 298–311.
[66] Teodorovi, D. and Guberini, S. Optimal dispatching strategy on an airline network after a schedule perturbation, *Eur. J. Oper. Res.*, 1984, **15**, (2), pp 178–182.
[67] Argüello, M.F., Bard, J.F. and Yu, G. A GRASP for aircraft routing in response to groundings and delays, *J. Comb. Optim.*, 1997, **1**, (3), pp 211–228.
[68] Petersen, J.D. et al. An optimization approach to airline integrated recovery, *Transport. Sci.*, 2012, **46**, (4), pp 482–500.
[69] Artigues, C. et al. Disruption management for commercial airlines: methods and results for the ROADEF 2009 challenge, *Eur. J. Ind. Eng.*, 2012, **6**, (6), pp 669–689.
[70] Bisaillon, S., Cordeau, J.F., Laporte, G. and Pasin, F. A large neighbourhood search heuristic for the aircraft and passenger recovery problem, *4OR*, 2011, **9**, (2), pp 139–157.
[71] Jozefowiez, N., Mancel, C. and Mora-Camino, F. A heuristic approach based on shortest path problems for integrated flight, aircraft, and passenger rescheduling under disruptions, *J. Oper. Res. Soc.*, 2013, **64**, (3), pp 384–395.
[72] Brunner, J.O. Rescheduling of flights during ground delay programs with consideration of passenger and crew connections, *Transport. Res. Part E: Logist. Transport. Rev.*, 2014, **72**, pp 236–252.
[73] Aktürk, M.S., Atamtürk, A. and Gürel, S. Aircraft rescheduling with cruise speed control, *Oper. Res.*, 2014, **62**, (4), pp 829–845.
[74] Castro, A.J.M., Rocha, A.P. and Oliveira, E. *A New Approach for Disruption Management in Airline Operations Control* (Vol. **562**, No. 3, pp 125–139). Heidelberg: Springer, 2014.
[75] Hu, Y., Liao, H., Zhang, S. and Song, Y. Multiple objective solution approaches for aircraft rerouting under the disruption of multi-aircraft, *Exp. Syst. Appl.*, 2017, **83**, pp 283–299.

https://doi.org/10.1017/aer.2025.16 Published online by Cambridge University Press

[76] Sousa, H., Teixeira, R., Cardoso, H.L. and Oliveira, E. Airline disruption management dynamic aircraft scheduling with ant colony optimization, in *Proceedings of the International Conference on Agents and Artificial Intelligence*, 2015, pp 398–405.
[77] Niendorf, M., Kabamba, P.T. and Girard, A.R. Stability analysis of optimal runway schedules, in *American Control Conference (ACC), 2015*, IEEE, 2015, pp 4616–4621.
[78] Sama, M., D'Ariano, A., D'Ariano, P. and Pacciarelli, D. Scheduling models for optimal aircraft traffic control at busy airports: tardiness, priorities, equity and violations considerations, *Omega*, 2017, **67**, pp 81–98.
[79] Sama, M., D'Ariano, A., Corman, F. and Pacciarelli, D. Metaheuristics for efficient aircraft scheduling and rerouting at busy terminal control areas, *Transport. Res. Part C: Emerg. Technol.*, 2017, **80**, pp 485–511.
[80] Nisse, N., Salch, A. and Weber, V. Recovery of disrupted airline operations using k-Maximum Matching in Graphs, PhD thesis. Inria Sophia Antipolis, 2015.
[81] Rosenthal, E.C. and Eisenstein, E.M. A rescheduling and cost allocation mechanism for delayed arrivals, *Comput. Oper. Res.*, 2016, **66**, pp 20–28.
[82] Kjenstad, D. et al. Integrated surface and departure management at airports by optimization, in *Modeling, Simulation and Applied Optimization (ICMSAO), 2013 5th International Conference on*, IEEE, 2013, pp 1–5.
[83] Rodríguez-Díaz, A., Adenso-Díaz, B. and González-Torre, P.L. Minimizing deviation from scheduled times in a single mixed-operation runway, *Comput. Oper. Res.*, 2017, **78**, pp 193–202.
[84] Ng, K.K.H. et al. Robust aircraft sequencing and scheduling problem with arrival/departure delay using the min-max regret approach, *Transport. Res. Part E: Logist. Transport. Rev.*, 2017, **106**, pp 115–136.
[85] Kammoun, M.A. and Rezg, N. An efficient hybrid approach for resolving the aircraft routing and rescheduling problem, *J. Air Transport Manag.*, 2018, **71**, pp 73–87.
[86] Lin, H. and Wang, Z. Fast variable neighborhood search for flight rescheduling after airport closure, *IEEE Access*, 2018, **6**, pp 50901–50909.
[87] Ali, K.M. and Nidhal, R. Continuous flight rescheduling problem resolution based on genetic algorithms, in *2019 IEEE 6th International Conference on Industrial Engineering and Applications (ICIEA)*, IEEE, 2019, pp 791–795.
[88] Erkan, H., Erkip, N.K. and Afak, Ö. Collaborative decision making for air traffic management: a generic mathematical program for the rescheduling problem, *Comput. Ind. Eng.*, 2019, **137**, p 106016.
[89] Ng, K.K.H. et al. A two-stage robust optimisation for terminal traffic flow problem, *Appl. Soft Comput.*, 2020, **89**, p 106048.
[90] Nasab, P.T., Ghasemi, M.V. and Tohidi, G. A mathematical model for robust landing and take-off scheduling at an airport considering runway disturbances, *Int. J. Res. Ind. Eng.*, 2023, **12**, (1), pp 21–42.
[91] Rangaiah, G.P. *Multi-Objective Optimization: Techniques and Applications in Chemical Engineering*, Vol. 1. World Scientific, 2008.
[92] Trivizas, D.A. Optimal scheduling with maximum position shift (MPS) constraints: a runway scheduling application, *J. Navigat.*, 1998, **51**, (2), pp 250–266.

Appendix

Aircraft = [1–15]

Ready Times = [35,51,57,76,109,122,165,203,218,237,252,264,276,354,441]

Target Times = [95,111,117,136,169,182,225,263,278,297,312,324,336,414,501]

Deadlines = [635,651,657,676,709,722,765,803,818,837,852,864,876,954,1041]

Weights = [1,1,2,2,2,2,3,3–5,5,5,6,6,6]

For the given problem instance, the best total weighted start time value is 177, which is obtained by solving the MILP presented earlier. The initial schedule is shown in Table 13 with the following operation start times and runway assignments.

Start times = [35,57,75,107,135,147,207,231,261,264,299,311,371,406,478]

Runway Assignment = [(1, 1), (2, 1), (3, 2), (4, 1), (5, 2), (6, 2), (7, 1), (8, 2), (9, 2), (10, 2), (11, 1), (12, 1), (13, 2), (14, 2), (15, 2))

where (j, i) indicates that aircraft j is assigned to runway i.

Suppose that the flight associated with aircraft 3 is canceled, and only the aircraft affected by the disruption are to be repaired. Since the canceled aircraft was scheduled on runway 2, the aircraft on runway 1 are not affected by the cancelation. Moreover, because aircraft 8 is scheduled prior to 3, it is not affected by this cancelation. The set of aircraft affected by the disruption and considered for rescheduling is {9, 10, 6, 5, 15, 13, 14}.

The steps to calculate $\widetilde{f_{TWS}}(x^*)$, $\widetilde{f_{TWRD}}(x^*)$, $\widetilde{f_{TWS}}(x^*)$, $\widetilde{f^{max}_{TWSD}}$, $\widetilde{f^{max}_{TWRD}}$ and $\widetilde{f^{max}_{TWS}}$ are summarised as follows:

https://doi.org/10.1017/aer.2025.16 Published online by Cambridge University Press

Table 13. *Initial schedule before disruption*

Runway #1	1	2	11	4	7	12			
Runway #2	8	3	9	10	6	5	15	13	14

Table 14. *Optimal solution after disruption*

Runway #1	1	2	11	4	7	12			
Runway #2	8		9	10	6	5	15	13	14

1. The index sets and parameter values are initialised based on the disruption information (i.e. the parameter values of aircraft 3 are omitted since it is canceled).
2. Set $\pi_1 = 1 - \pi_2 - \pi_3$ when $\pi_2 = \pi_3 = \varepsilon$ and run the MILP to estimate the minimum value of total weighted start time deviation $\widetilde{f_{TWSD}}(x^*)$, and a worst value of the total weighted start time $\widetilde{f^{max}_{TWS}}$.
3. Set $\pi_2 = 1 - \pi_1 - \pi_3$ when $\pi_1 = \pi_3 = \varepsilon$ and run the MILP to estimate the minimum value of total weighted runway deviation, $\widetilde{f_{TWRD}}(x^*)$.
4. Set $\pi_3 = 1 - \pi_1 - \pi_2$ when $\pi_1 = \pi_2 = \varepsilon$ and run the MILP to estimate the minimum value of total weighted start time, $\widetilde{f_{TWS}}(x^*)$ and the maximum values of the total weighted start time deviation $\widetilde{f^{max}_{TWSD}}$, and total weighted runway deviation f^{max}_{TWRD}.

After following these steps, it is estimated that for this example the best (optimistic) values for $[\widetilde{f_{TWSD}}(x^*),\ \widetilde{f_{TWRD}}(x^*),\ \widetilde{f_{TWS}}(x^*)] = [0,\ 0,\ 10752]$ and the worst (pessimistic) values for $[\widetilde{f^{max}_{TWSD}}, \widetilde{f^{max}_{TWRD}}, \widetilde{f^{max}_{TWS}}] = [10752,\ 32,\ 21314]$.

Accordingly, the objective function for this problem is updated as follows:

$$\text{Min}\ \ \pi_1 \left(\frac{f_{TWSD}(x) - 0}{10752 - 0}\right) + \pi_2 \left(\frac{f_{TWRD}(x) - 0}{32 - 0}\right) + \ \pi_3 \left(\frac{f_{TWS}(x) - 10752}{21314 - 10752}\right) \tag{11}$$

The new optimal schedule for this instance is obtained by minimising the objective function in 9 which takes into account both schedule stability and solution quality. If the coefficient values are set for example to $\pi_1 = \pi_2 = 0.5, \quad \pi_3 = 0$, the optimal solution obtained will be as shown in Table 14 and will have an objective value of 0 with TWSD = 0, TWRD = 0, and TWS = 11,430. In this example, since the runway and start times deviations are equally weighted to 0.5 while the start times are neglected, the optimal solution will keep all operations as they are to avoid instability penalties. When the values of π are different, the solution may be different. In other words, if the stability measure is much more important than the solution quality (i.e. start times), the new schedule tends not to change aircraft positions to keep the schedule stable.

Cite this article: Hancerliogullari Koksalmis G., Rabadi G., Kharbeche M. and Al-Salem M. A mathematical model and heuristic approaches for runway rescheduling. *The Aeronautical Journal*, https://doi.org/10.1017/aer.2025.16