https://doi.org/10.1016/j.tre.2017.08.006

Robust Aircraft Sequencing and Scheduling Problem with Arrival/Departure Delay using Min-max Regret Approach

K.K.H. NG, C.K.M. LEE*, Felix T.S. CHAN, Yichen QIN

Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University, Hung Hom, Hong Kong, China

* Corresponding author

Address: Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University, Hung Hom, Hong Kong.

Tel.: +852 3400 3899; fax: +852 2362 5267

Email Address: kkh.ng@connect.polyu.hk (K.K.H. NG), ckm.lee@polyu.edu.hk (C.K.M. LEE), f.chan@polyu.edu.hk (Felix T.S. CHAN), Yichen.qin@connect.polyu.hk (Yichen QIN)

Robust aircraft sequencing and scheduling problem with arrival/departure delay using the min-max regret approach

Abstract

This study considers the aircraft sequencing and scheduling problem under the uncertainty of arrival and departure delays for multiple heterogeneous mixed-mode parallel runways. To enhance runway resilience, runway operations should remain robust to mitigate the effects of delay propagation. The main objective of this research was to identify an optimal schedule by evaluating the robustness of feasible solutions under its respective worst-case scenario. A novel artificial bee colony algorithm was developed and verified by experimental results. The proposed efficient artificial bee colony algorithm can obtain close-to-optimal results with less computational effort in regard to a one-hour flight traffic planning horizon.

Keywords: Robust scheduling, Min-max regret approach, Mixed-mode parallel runways, Swarm Intelligence, Artificial bee colony algorithm

1. Introduction

1.1. Problem description

With the introduction of low-cost carriers in Western countries and the remodelling of airport-airline relationships, air transport demand has been significantly increased due to the capacity bottlenecks (Francis et al., 2004; Gelhausen et al., 2013). The performance of the aircraft turnaround process and the airport-airline relationship affects the decision-making of the objective function in the aircraft sequencing and scheduling problem (ASSP) model. The common First-Come-First-Served (FCFS) approach creates unnecessary spare capacity in the ASSP model (Bennell et al., 2017; Ng and Lee, 2016a). The most relevant operational problem in the aircraft scheduling literature often considered the global optimality in practice within a reasonable computation time (Samà et al., 2015). It is frequently observed that real-time aircraft re-scheduling in runway operation occurs. Air Traffic Control (ATC) obtains the latest information on flights to determine a schedule. The ATC workload has dramatically increased due to rising air transport demand. The estimated time of arrival/departure may not be close to the true operation time, which may lead to disruption of planned flight schedules and subsequent ground operation schedules (Sinclair et al., 2014). In fact, the introduction of a robustness optimisation technique in the ASSP problem improves the resilience at busy airports and leverages the possible workload of re-scheduling effort.

This paper aims to improve the runway operations by considering the solution quality, computation time, the resilience level of runway operation and the degree of robustness in makespan optimisation of aircraft scheduling in hedging uncertainties. In this model, we considered the aircraft sequencing and scheduling problem with multiple heterogeneous mixed-mode parallel runways. In mixed-mode runway operation, the runway is used for both aircraft landings and take-offs. The arrival and departure rate in an airport usually does not have a stationary distribution subjected to the landing/take-off demand patterns, which implies inefficient runway capacity usage, solely for landing or take-off in independent runway operation (Jacquillat and Odoni, 2015a, b; Jacquillat et al., 2017). Mixed-mode parallel runway operations further enhance the capacity to handle airborne and airport traffic, but also increase the degree of ATC workload. Uncontrollable delays also increase the vulnerability to disruptions. To enhance the robustness of a flight schedule and reduce the possibility of a re-scheduling effort by ATC, the objective of this research was to minimise the maximum makespan deviation from optimality over all worst-case scenarios using the min-max regret approach.

The iterative relaxation framework is the standard procedure to solve the min-max regret problem (Aissi et al., 2009). Aissi et al. (2009) addressed that the computational complexity for min-max regret optimisation is a great challenge in the field. Meta-heuristics in min-max regret optimisation have been successfully applied to parallel machine scheduling, job shop scheduling and other related

problems (Feng et al., 2016; Hu et al., 2016; Xu et al., 2013). Therefore, we propose the Efficient Artificial Bee Colony (EABC) algorithm to enhance the computational efficiency for obtaining a robust schedule with close-to-optimal condition.

1.2. Literature review

Runway capacity is the major bottleneck in air traffic management (Balakrishnan and Chandran, 2010; Ghoniem et al., 2014). In order to maintain a smooth airport operation, managing aircraft landing and take-off procedures has become a key component in air transport systems. Runway operation includes the flight approach operation/Aircraft Landing Problem (ALP) (see, e.g., Beasley et al. (2001); Bencheikh et al. (2009); Caprì and Ignaccolo (2004); Hancerliogullari et al. (2013); Hansen (2004); Liu (2011); (Ng and Lee, 2017); Pinol and Beasley (2006); Salehipour et al. (2013); Vadlamani and Hosseini (2014)), departing operation/Aircraft Take-off Problem (ATP) (see, e.g., Atkin et al. (2008); Hancerliogullari et al. (2013)) and mixed-mode parallel operation/ASSP (see, e.g., Bennell et al. (2011); Lieder and Stolletz (2016)). Mixed-mode parallel operation allows simultaneous runway operations for a pair of flights on different runways (Beasley et al., 2000).

Regarding the objectives in formulating the ASSP model, various objective functions can be found in the literature. These include: minimising the makespan (Balakrishnan and Chandran, 2010; Harikiopoulo and Neogi, 2011; Ma et al., 2014; Ng and Lee, 2016a), minimising total/weighted tardiness of all flights (Ng and Lee, 2016b; Pinol and Beasley, 2006; Sabar and Kendall, 2015; Salehipour et al., 2013), and minimising total/average/weighted delay of all flights (Lieder and Stolletz, 2016; Liu, 2011; Samà et al., 2015). The FCFS approach is a policy that maintains fairness among flights in a schedule (Farhadi et al., 2014). Dear and Sherif (1989) revealed that the FCFS approach is undesirable as the ASSP solution must be updated promptly to cope with real-time needs. Beasley et al. (2004) proposed a displacement rule for ASSP to absorb the perturbations in a predefined schedule. Farhadi et al. (2014) introduced Constrained Position Shifting (CPS) based on the FCFS schedule in mixed-mode parallel runway operation. Soomer and Koole (2008) evaluated the trade-off between total cost, delay and fairness to obtain a schedule that compromises different stakeholders' interests.

Microscopic air traffic flow modelling enhances the level of practical usage and robustness of the solution, which provides detailed control of the practice of air traffic control, including air segments, holding patterns, runway operation and ground operation. Detailed characteristics of airport infrastructure and flight paths facilitate the modelling accuracy regarding the on-time coordinates, status and speed profile (Samà et al., 2017). Bianco et al. (1997) formulated microscopic modelling in runway scheduling with blocking and a no-wait version of job-shop scheduling using an alternative graph for managing the streaming of Terminal Manoeuvring Area (TMA) operations. Regarding airport layout and the structure of air segments, sophisticated characteristics and real-world constraints have been proposed in runway scheduling using the extensive versions of microscopic modelling (D'Ariano et al., 2015; D'Ariano et al., 2012; Samà et al., 2017; Samà et al., 2014; Samà et al., 2013). Artiouchine et al. (2008) and Eun et al. (2010) considered the discrete holding patterns and airborne delays to maintain a smooth landing schedules. Providing the latest information assists ATC to resolve the potential conflict detection and collusion-free guidance within the TMA. The current research progress is still mired in the static approach. The aforementioned literature is static in nature, which means that all variables and information are known in advance.

Most international airports find that the associated financial costs caused by airborne delays are significant so they try to moderate the cost by reducing the flight delay times (Ball et al., 2010; Hansen and Zou, 2013; Ng and Lee, 2017; Zou and Hansen, 2012). The sensitivity of an airport network is remarkable as all the airport resources are highly linked (Beatty et al., 1999). The delay of a flight leads to delay propagation of various airport activities and scheduling (Campanelli et al., 2016; Churchill et al., 2010; Kafle and Zou, 2016; Pyrgiotis et al., 2013). It is known that aircraft approaching and take-off times are not precisely determined in

advance due to the dynamic changes of the environment. Risk analysis in the ASSP model is lacking in the current literature. According to the decision theory of risk analysis, uncertainties are defined as the outcome of a decision remaining unknown when an alternative is selected (Bell, 1982). Various uncertainties, practical constraints and the changes in the dynamic environment are considered in the model, which leads to an increase in the practical complexities in the decision-making for ATC. There are two types of methods for handling uncertainty: stochastic and robust modelling. In the stochastic approach, the uncertain variables in a model are designed as a known probability distribution by the historical data. Nonetheless, the expected outcome may not be able to be derived from past records in certain situations. Therefore, robustness analysis has become of more interest under the risk-adverse approach than optimal performance. Robustness analysis provides a paradigm supporting the decision-making considering imprecision so as to properly frame a decision based on any possible input data (Aissi et al., 2009). Robust schedule modelling develops a solution by hedging against the worst-case scenarios and is especially applicable to the potential huge loss of disruption afterwards. The possible scenarios in robustness analysis can be classified as interval and discrete scenarios (Kouvelis and Yu, 1997).

There are three robustness criteria for robustness analysis: absolute robustness, robust deviation and relative deviation (Xu et al., 2013). The min-max regret approach was considered in this research and is designed for developing a robust schedule that minimises the maximum makespan deviation from the optimal schedule under the worst-case scenario. The runway is considered as a bottleneck between the airborne and airport traffic. Any delay in air traffic causes significant delays among the airport activities if proper management in the ASSP model is omitted (Rodríguez-Díaz et al., 2017). The motivation for using the min-max regret approach is to neutralise the risk and avoid wrong decisions (e.g., delay/revision in airline schedule, poor gate assignment, customer dissatisfaction and overcharging for the congestion externality) (Basso, 2008). The major drawback of using Mixed Integer Linear Programming (MILP) in the min-max regret approach is that the computation time is significantly lengthened in resolving large-sized instances, and high computational capacity is required. The computation using the exact algorithm in min-max regret optimisation is costly as the complexity of the computation increases along with the number of worst-case scenarios (Feng et al., 2016). Also, the complexity of the computation increases dramatically with the size of the model due to the nature of the non-deterministic polynomial hard (NP-hard) problem (Bianco et al., 1997).

During the development of meta-heuristics in the current literature, the solution quality derived from the meta-heuristics regarding exploitation and exploration has greatly narrowed the research gap for the complex mathematical modelling in real-life applications, especially under the population-based meta-heuristics approach (Bianchi et al., 2009; Glover, 1986). The meta-heuristics approach is a high-level model-free framework to obtain a near optimal solution within a satisfactory calculation time and includes: Stochastic Search, evolutionary algorithms, physics-based algorithms and swarm intelligence. Several meta-heuristic approaches have been recently proposed in the ASSP model, including Simulated Annealing (SA) (Hancerliogullari et al., 2013; Salehipour et al., 2013), Variable Neighbourhood Search (VNS) (Ng and Lee, 2016b; Salehipour et al., 2013; Vadlamani and Hosseini, 2014), Iterative Local Search (ILS) (Sabar and Kendall, 2015), Genetic Algorithm (GA) (Beasley et al., 2001; Pinol and Beasley, 2006), Memetic Algorithm (MA) (Bencheikh et al., 2009), Biogeography-Based Optimisation (BBO) (Dastgerdi et al., 2015), Bat Algorithm (Xie et al., 2013) and Ant Colony Optimisation (ACO) (Jiang et al., 2014; Zhan et al., 2010). Among these four groups, swarm intelligence has been well studied in regard to scheduling problems and is regarded as a promising technique for resolving the NP-hard model (Karaboga and Akay, 2009; Karaboga et al., 2014). The mechanism of the SI algorithm aims to maintain the balance of exploitation and exploration to enhance the time for convergence and optimality (Zhang et al., 2015). The main feature of searching for an optimal value in swarm intelligence relies on the swarm behaviour of self-organisation, decentralisation and collective searching (Jeanne, 1986; Kube and Bonabeau, 2000; Trelea, 2003).

1.3. Contribution of the research

The contribution of the research can be summarised as follows: First, we address the mixed-mode parallel operation and the uncertain parameters in the scheduled time of operations in order to enhance the capacity and resilience level. In this research, minimising the maximum regret value in the robust ASSP system under the mixed-mode parallel operation was considered in the problem formulation. The arrival and departure delays fall into an interval case to represent a different level of delays. We believe that the design of robust scheduling considering uncontrollable delays enhances the resilience level of runways and reduces the ATC efforts in runway rescheduling. To achieve the practical applicability of mixed-mode parallel operation, we formulate the landing and take-off time of each arrival or departure flight in a time interval, which is affected by the possible scenarios of different levels of arrival and departure delays. To the best of our knowledge, this is the first attempt to adopt a decision theory for the robust ASSP, extending the model of the robust machine scheduling problem (Hu et al., 2016; Xu et al., 2013). Second, this paper presents a novel swarm intelligence algorithm, which significantly reduces the computational effort in iterative relaxation procedure for min-max regret optimisation. The proposed EABC algorithm enhances the convergence rate and reduces computation time. The proposed EABC algorithm is based on the following modified framework to solve the problem above. The initialisation phase in the EABC algorithm is computed by a proposed constructive heuristic to build a satisfactory solution. Inefficient neighbourhood search, crossover and reverse operators, are eliminated in the employed bee phase to enhance the searching quality, as these two operators work inefficiently during the exploitation phase. In addition, an objective-guided updating mechanism in the scout bee phase is considered. With these proposed improvements of the Artificial Bee Colony (ABC) algorithm, the robust solution is improved regarding the computation time as well as the solution quality in our experiments by comparing with the exact algorithm, biological evolution and swarm intelligence algorithms.

1.4. Organisation of the paper

The organisation of this paper is summarised as follows. After the introduction of the research and literature study on the ASSP model in **Section 1**, **Section 2** describes robust ASSP modelling and problem formulation. **Section 3** presents the solution procedure for the robust ASSP model using the exact method, meta-heuristics and proposed EABC algorithms. Computational experiments are reported in **Section 4**. Finally, the concluding remarks and future work are raised in **Section 5**.

2. Problem formulation

In this study, a robust ASSP with arrival and departure delays was considered in the system, aiming to enhance the robustness of the scheduling with the objectives of makespan minimisation, considering the limited number of runways, resource capacity constraints and ATC regulation. Due to the technological advancement and improvement of radar monitoring systems, air traffic controllers can apply simultaneous operations by following the particular rules and regulations to improve the effectiveness of runway control. The runway configuration of the formulated problem is denoted as a mixed-mode parallel operation, in which the runways are not exclusively for approach or departure only. Compared with independent parallel operation, the mixed-mode parallel operation reduces redundant runway capacity, and enhances the maximum capability of the aerodrome capacity to handle the air traffic incorporating the constraints of the separation requirements. For example, 196 seconds of separation time is required if an approaching large-sized flight is scheduled to land before a small-sized approaching aircraft in a segregated operation model. Under the mixed-mode operation, the trailing departure flight has a 75-second pause until completion of the landing process of a large-sized aircraft. Other considerations may also affect the decision for the selection of parallel operation, such as the physical properties of runways, the minimum distance for parallel operation, the noise abatement procedures for residential areas and the capability of ATC. Other assumptions are considered below in constructing the robust ASSP model.

2.1. Assumption of the robust ASSP

Several assumptions were made before the formulation of the robust ASSP model. First, the robust ASSP schedules are limited to the runway decisions without the involvement of the TMA resources, such as ground operations, terminal air traffic control and queue length in the airborne traffic. Second, the length of all the runways is sufficient to perform the mixed-mode operation. Depending on the flight classes and size, adequate length of the runway is required to accommodate the speed reduction for landing operations. Moreover, the minimal length of the runway for landing and take-off are different even for the same classes of flights. Third, the separation requirement caused by the runway's physical properties (i.e. near-parallel runways, the size of the no-transgression zone (NTZ), the terrain constraints surrounding the airport and the noise abatement procedures) are minimal in the instrument landing system. For example, Hong Kong International Airport is not able to operate missed approaches and departure procedures for the south runway due to the terrain constraints of Lantau Island. Fourth, the probabilities of a missed approach and departure, the penetration of NTZ, pilot error, runway incursions and the abnormal operation are negligible. Emergency operations require clearance of deviation procedures and establishing an appropriate separation between flights before the recovery of cleared routes and normal operation. The emergency events in terminal air traffic are ignored in the robust ASSP model. Fifth, the parameters of approaching and departing delays are imprecise concerning interval values. The estimation of the arrival time is not precisely obtained even if flights have entered the near terminal area. The planned schedule may also be disrupted by any small disturbances, including the flight delay, the terminal weather and the level of wind speed, the airborne conflict and disruption on the turnaround procedures. The formulation of imprecision in robust optimisation assists schedulers to create a robust schedule by considering plausible values or time intervals of the operation time.

2.2. The robust ASSP model

In the robust ASSP model, the number of aircraft is defined as n, and each flight is denoted as $i, i = (1,2,\dots,n)$. Any pair of consecutive runway operations is regarded as j and i to denote a pair of arrival/arrival, departure/departure, arrival/departure and departure/arrival flights in a consecutive sequence. The maximum number of runways is m. Each approaching flight can only land after it reaches the tower control region, where the departing flight may execute take-off procedures once the pilot receives permission for take-off from the tower control. The scheduled landing/departing time represents the estimated landing/departing time for an aircraft to land on a corresponding runway under the planning horizon. The assigned time of operation between the two consecutive flights must be larger than the required separation time s_{ji}, according to the flight classes and runway operation (Balakrishnan and Chandran, 2010). The notation and decision variables are shown in **Table 1**.

Table 1

Notation and decision variable

Notations	Explanation
i	Aircraft ID $i \in I,\ (i = 1,2,\dots,n)$
n	The maximum number of aircraft
r	Runway ID $r \in R,\ (r = 1,2,\dots,m), m \geq 2$
m	The maximum number of runway
S_{ij}	The runway operation based separation time between aircraft i and j scheduled on the same runway, $S_{ij} \geq 0$
STO_{ir}	The scheduled landing/take-off time of aircraft i on runway r
$\underline{STO_{ir}}$	The lower bound value of scheduled landing/take-off time of aircraft i on runway r
$\overline{STO_{ir}}$	The upper bound value of scheduled landing/take-off time of aircraft i on runway r
STO_{ir}^{s}	The time interval of landing/take-off time of aircraft i on runway r, $STO_{ir}^{s} \in \left[\underline{STO_{ir}}, \overline{STO_{ir}}\right]$
ε_{ir}	The possible deviation from the predetermined operation time with or without interruption

s	The possible realised operation time in a scenario, $s = (STO_{1r}^{s}, STO_{2r}^{s}, \dots, STO_{(n-1)r}^{s}, STO_{nr}^{s}), s \in \delta$
ω	The set of all feasible schedules
M	Large number associated with the artificial variable

Decision variables	Explanation
X	A schedule X is constructed by x_{ir}, y_{jir} and $T_{ir}^{s}(x)$.
x_{ir}	1, if aircraft i is assigned to runway r; 0, otherwise
y_{jir}	1, if aircraft j is before aircraft i on the same runway r (not necessarily immediately); 0, otherwise
$T_{ir}^{s}(X)$	The assigned operation time for aircraft *i* on the runway r in schedule X under scenario s, $T_{ir} \geq 0$
C_{r}^{s}(X)	The makespan of schedule *X* under scenario s, $C_r \geq 0$

x_{ir} is the decision variable to identify the runway assignment of each flight i, while y_{jir} defines the landing/take-off sequence if flight j lands before flight i on runway r (not necessarily immediately). Given that the scheduled operation times are heterogeneous on different runways for the same flight i, the scenario $s = (STO_{1r}^{s}, STO_{2r}^{s}, \dots, STO_{(n-1)r}^{s}, STO_{nr}^{s}), s \in \delta$ is a set of scheduled times of operation and S denotes a set of all possible scenarios. A schedule is represented as X. ε_{ir} is the deviation between the lower bound and upper bound value of scheduled landing/take-off time of flight i on runway r to represent the possible operation time given an unknown distribution. We assume that the possible operation time of all flights are different and follow into an interval case.

2.3. The robust ASSP formulation

The min-max regret approach seeks to undertake all the scenarios with uncertainties in the aircraft landing and take-off schedules to conduct the robust scheduling. Accordingly, the model aims to discover a robust ASSP schedule by minimising the maximum makespan deviation from the optimal schedule under the worst-case scenarios. C_r^s is the completion time of runway r under scenario s. The makespan $F(X,s)$ of a schedule X under scenario s is determined as Equation (1), while the minimal makespan of a schedule is denoted as the optimal schedule X_s^* under scenario s by Equation (2). Given a feasible solution $X \in \omega$, its regret value $Regret(X,s)$ under scenario $s \in \delta$ is defined as the difference between the makespan of schedule X and the optimal makespan under this scenario by Equation (3), while the maximal regret is interpreted as Equation (4). Feasible solution is defined as a solution that satisfies all the constraints from the mathematical formulation. Subsequently, the model establishes the robust scheduling by minimising the deviation from the optimal solution under the worst-case scenarios by Equation (5).

$$F(X,s) = \max_{r \in R}(C_r^s(X)) \tag{1}$$

$$F_s^* = F(X_s^*, s) = \min_{X \in \omega} F(X,s) \tag{2}$$

$$Regret(X,s) = F(X,s) - F_s^* \tag{3}$$

$$Regret_{max}(X) = \max_{s \in \delta} Regret(X,s) \tag{4}$$

$$\min_{X \in \omega} Regret_{max}(X) = \min_{X \in \omega} \max_{s \in \delta}(F(X,s) - F_s^*) \tag{5}$$

The makespan of runway r in schedule X under scenario s is calculated by Equation (6) and must be equal to the assigned landing time of the last flight on the runway system for a finite time horizon. Equations (7) and (8) compute the assigned landing/take-off time of each flight $i \in I$, where I is the set of flights, taking into account the separation time S_{ji} between two flights j and i, and its scheduled operation time. The assigned operation time of all flights denotes the time of runway operation in TMA. ATC will assign an appropriate landing time based on the corresponding scheduled time of operation (STO), including the landing and take-off, with the consideration of the runway availability and the air traffic. The configuration and physical properties of the runways may vary, which leads to an unrelated STO.

$$C_r^s \geq T_{ir}^s(X) - M(1 - x_{ir}), \forall i, r, s \quad (6)$$

$$T_{ir}^s(\mathrm{X}) - T_{jr}^s(X) \geq S_{ji} - M\left(1 - y_{jir}\right), \forall i, j, i \neq j, r, s \quad (7)$$

$$T_{ir}^s(\mathrm{X}) \geq STO_{ir}^s - M(1 - x_{ir}), \forall i, r, s \quad (8)$$

The robust ASSP is a kind of scheduling problem that evaluates the optimal robust scheduling that can perform well under all the possible scenarios. The critical runway is defined as a runway with the longest completion time under scenario s in an aircraft schedule X. Equation (9) minimises the maximum deviation of makespan across all the scenarios $s \in \delta$. The completed mathematical formation of the robust ASSP model is shown below:

$$(\mathrm{RobustASSP}) \min_X (\max_{s \in \delta} [F(X, s) - F_s^*]) \quad (9)$$

$$s.t.$$

$$x_{ir} + x_{jr} \leq 1 + y_{ijr} + y_{jir}, \forall i, j, i \neq j, r \quad (10)$$

$$y_{jir} + y_{ijr} \leq 1, \forall i, j, i \neq j, r \quad (11)$$

$$\sum_{r=1}^m x_{ir} = 1, \forall i \quad (12)$$

$$x_{ir} \in \{0,1\}, \forall i, r \quad (13)$$

$$y_{jir} \in \{0,1\}, \forall i, j, r \quad (14)$$

and

(6) - (8)

Constraints (10) and (11) guarantee that y_{jir} is equal to 1 if flight i is assigned after flight j on the corresponding runway r (not necessarily immediately). Otherwise, the y_{jir} takes a zero value. Each flight i is restricted to being assigned to only one runway r for the landing/take-off schedule by constraint (12). Constraints (13) and (14) confirm that the decision variables x_{ir} and y_{jir} are binary numbers.

In the iterative relaxation framework, each iteration determines the worst-case scenario of a given schedule in several extreme point scenarios by calculating the optimal makespan of each extreme point scenario to obtain the respective regret values of a given schedule as shown in **Proposition 1**. The extreme point scenario is defined in **Definition 1**. Under each extreme point scenario s, the scheduled operation time of each flight is equal to either its lower bound value $\underline{STO_i}$ for non-critical runways or its upper bound value $\overline{STO_i}$ for the critical runway in the min-max regret approach depending on the **Definition 1**.

Definition 1. Given a makespan minimisation in the ASSP, the regret of a solution $x \in X$ is maximised for the extreme point scenario s^k, which is defined as follows:

$$STO_i^{s^k} = \begin{cases} \overline{STO_i}, if\ x_{ik} = 1 \\ \underline{STO_i}, if\ x_{ik} = 0 \end{cases}, i = 1,2, \dots, n \quad (15)$$

or

$$STO_i^{s^k} = \overline{STO_i} x_{ik} + \underline{STO_i}(1 - x_{ik}), i = 1,2, \dots, n \quad (16)$$

Definition 2. A runway $\zeta \in m$ is declared to be critical in an aircraft schedule $X \in \omega$ under scenario $s \in \delta$ if the completion time of runway r is the longest in schedule X under scenario s.

$$C_{\zeta}^{s}(X) = \max_{r \in R}\{C_{r}^{s}(X)\} = F(X, s) \quad (17)$$

Proposition 1. For any schedule $X \in \omega$, let s^0 be a worst-case scenario under the flight landing and take-off schedule when runway $\zeta \in$ R is critical. There must be a scenario s^{ζ} that meets the following conditions. First, runway ζ is critical under scenario s^{ζ}. Second, scenario s^{ζ} is a worst-case scenario for flight schedule X.

Proof. See **Appendix A**.

3. Methodology

The makespan minimisation under the deterministic situation and the robust optimisation are the two optimisation processes as in the iterative relaxation procedure. With respect to the objective of minimising runway makespan, the objective function in optimisation under the worst-case scenario is C_r^s. The regret value of each runway can then be extracted by Equation (3), given that definitions 1 and 2 hold under the worst-case regret approach. As for robust optimisation, the objective value is the minimising of the maximum regret value by considering all the possible scenarios with Equation (5). The objective in robust optimisation is nonlinear and requires further modification of the objective function and constraints.

3.1. Implementation of an exact algorithm

The regret value $Regret(X, s)$ measures the difference between the makespan of a schedule and the optimal solution under scenario s. The optimal makespan F_s^* under scenario s can be directly calculated by mixed-integer programming. However, finding the minimal-maximal regret cannot be solely solved as it involves an infinite number of scenarios $|\Omega| = \infty$. Therefore, an iterative relaxation procedure is incorporated to obtain a robust solution in the min-max regret optimisation model (Inuiguchi and Sakawa, 1995; Mausser and Laguna, 1998; Mausser and Laguna, 1999). The relaxation procedure for the min-max regret optimisation is a standard approach for developing a robust schedule from an initial solution under scenario s^0 by adding regret cuts obtained from the worst-case scenarios in previous solutions iteratively. Since there is infinite number of possible realisation of scenarios, we can only consider the limited number of scenarios to confine the model by introducing regret cuts. Regret cut refers to the valid inequalities tightening the lower bound of the objective function of the robust optimisation during the search process. The lower bound of the objective function is tightened gradually during the iteration process, as more scenarios are taken into consideration. The initial solution is constructed by taking all the scheduled operation times of landing/take-offs as the lower bound values. The iterative process identifies the worst-case scenario of the latest solution, and further constructs a relaxed ASSP model by incorporating the newly identified worst-case scenario into the iterative relaxation procedure. Hence, the robust solution for the ASSP model can be determined using the iterative relaxation procedure.

Proposition 2. The maximum regret value is denoted by comparing with each runway as critical $\zeta \in$ R for the aircraft landing and take-off sequence X for a multiple runways system ($m \geq 2$), using the following equation.

$$Regret_{max}(X) = \max_{\zeta \in R}(C_{\zeta}^{s^{\zeta}} - F_{s^{\zeta}}^{*}) \quad (18)$$

Proof. See **Appendix A**.

The maximum regret value is unknown if the critical runway ζ is not defined in the robust ASSP. The critical runway ζ can be calculated by defining runway r^* as critical using the equation of $r^* = \arg\max_{\zeta \in R}(C_{\zeta}^{s^{\zeta}} - F_{s^{\zeta}}^{*})$. The assigned time of operation of the

flight T_i^s is associated with its scheduled time of operation STO_i^s or the scheduled operation time of the leading flight with separation requirement $T_j^s + S_{ji}$ by Equations (6) - (8) under scenario s, which further constitutes the formation of the completion time C_r^s. The regret value RV cannot be solely solved as it involves two optimisation operators in Equation (5), resulting in the nonlinear objective function. The robust aircraft landing and take-off sequence can be resolved by reformulating the model as follows:

$$min\ RV \tag{19}$$

$$s.t.$$

$$C_r^{s^k} - F_{s^k}^* \leq RV, \forall s^k \in \Omega, r = 1, 2, \dots, m \tag{20}$$

and

(6) - (8), (10) - (14)

$F_{s^r}^*$ is the optimal makespan under scenario s, while the C_r^s is the completion time of runway r under the worst-case scenario s^k. It is not appropriate to directly apply mixed-integer linear programming in solving the robust ASSP model. Therefore, an iterative process is required to consider each worst-case scenario by returning the constraints of the solution X to the robust ASSP model.

As in the aforementioned procedure of the min-max regret optimisation, the regret value cannot be obtained by a single optimisation method, and the number of worst-case scenarios is infinite. As we are seeking the optimal solution with minimal-maximal regret value across all the feasible solutions. The worst-case scenario of one solution can be used to confine the searching process of seeking a new incumbent solution. Specifically, once an incumbent solution found, its maximal regret value associated with its worst-case scenarios is recorded and presented in the form of cut for a subsequent iteration, implying that the new incumbent solution shall satisfy the prior identified worst-case scenario of the earlier found solutions. Proposition 2 develops and declares the worst-case scenarios with a limited set of scenarios $\Omega = (s^1, s^2, \dots, s^\omega)$. The near-optimal/optimal makespan of finite extreme point scenarios can be obtained by solving a MIP given a known scenario $s^k \in \Omega$. The constraint is a form of regret cut from the equation $C_r^{s^\zeta} - F_{s^\zeta}^* \leq RV,\ r = 1, 2, \dots, m$. Assume that the minimal-maximal regret for the robust ASSP model is $\widehat{RV}$. Given $\widetilde{\mathrm{RV}} \leq \widehat{RV}$, the objective value $\widetilde{RV}$ in the relaxed ASSP model is a non-decreasing value as the lower bound value $\widetilde{RV}$ increases to satisfy the regret cuts. Afterwards, the robust model can be solved using an iterative process by revising the objective function (9).

Given a schedule X, the maximal regret $Regret_{max}(X)$ can be obtained in accordance with its worst-case scenario $s \in \delta$ by proposition 1 and 2. First, the upper bound regret value $\widehat{RV}$ is set to be ∞ and the lower bound regret value $\widetilde{RV}$ is set to be 0. Then, the model obtains the optimal makespan under the lower-bound scenario as an initial solution $\hat{\mathrm{X}}$. The maximal regret from an initial solution $Regret_{max}(\hat{X})$ is defined as an upper bound regret value $\widehat{RV}$ for the robust ASSP model. If the current lower bound value is smaller than the upper bound value $\widetilde{RV} < \widehat{RV}$, the following process continues. The worst-case scenario $\hat{s}$ will be added in the set of scenarios δ in order to generate a new regret cut for the relaxed ASSP model. The objective value RV in the relaxed robust ASSP solution becomes the lower bound regret value $\widetilde{RV}$. The iteration is repeated until the lower bound regret value $\widetilde{RV}$ is equal to or larger than the upper bound regret value $\widehat{RV}$. Solution $\hat{X}$ will then become the robust optimal solution. The pseudo code of the iterative relaxation procedure is shown in **Table 2**.

Table 2

The pseudo code of the iterative relaxation procedure

The algorithm architecture of the iterative relaxation procedure

Set lower bound regret value $\widetilde{RV} = 0$ and upper bound regret value $\widehat{RV} = \infty$

Set the optimal makespan under lower-bound scenario as an initial solution $\hat{X}$

Define the upper bound regret value under extreme point scenario $\widehat{RV} = Regret_{max}(\hat{X})$

WHILE $\widetilde{RV} < \widehat{RV}$

Identify the worst case scenario $\hat{s}$ of the solution $\hat{X}$

IF $\widetilde{RV} \geq Regret_{max}(\hat{X})$

THEN $\widehat{RV} = Regret_{max}(\hat{X})$

Add regret cuts $C_r^{s^\zeta} - F_{s^\zeta}^* \leq RV,\ r = 1, 2, \ldots, m$ to the relaxed robust model

Solving the relaxed ASSP model and obtain the best-known solution $\hat{X}$ and set its $\widetilde{RV} = RV$

END

3.2. Proposed efficient artificial bee colony algorithm

The major drawback of using mixed integer programming (MIP) in the min-max regret approach is that the computation time is significantly lengthened in resolving large-sized instances, and high computational capacity is required. The computation using the exact algorithm in the min-max regret optimisation is costly as the complexity of the computation increases along with the number of worst-case scenarios (Feng et al., 2016). Also, the complexity of the computation increases dramatically with the size of the model due to the nature of the non-deterministic polynomial hard (NP-hard) problem (Bianco et al., 1997; Garey and Johnson, 1979). The relaxed ASSP model includes the makespan optimisation under each extreme point scenario and the robust optimisation in the iterative relaxation procedure. Meta-heuristics can be applied in these two optimisation problems. The ABC algorithm is a popular Swarm Intelligence-based algorithm which is able to balance the exploitation and exploration during the searching process. The conventional ABC algorithm adopts the division of labour strategy, self-organisation and collective behaviour of honey bees to explore and exploit the searching process efficiently. Although the basic ABC algorithm and the algorithm performance compared with other meta-heuristics have been well studied in regard to the general scheduling problem, customisation of the algorithm remains an important determinant to improve the algorithm's performance for robust modelling. In our exploratory experiments, biological evolution and variants of ABC algorithms have been applied. Our preliminary analysis showed that the results were not satisfied, in terms of the objective value and the computation time. The solution quality of the robust schedule is determined by the accuracy of the set of regret cuts added into the iterative relaxation model. The quality of regret cuts depends on the deviation of the best-known makespan under the worst-case scenario derived by the meta-heuristics from the real optimal makespan value. Specifically, the regret cut is presented in the form of $C_r^s - F_s^* \leq RV,\ r = 1, 2, \ldots, m$ in the mathematical model. For example, if F_s^1 obtained by meta-heuristics is larger than the true optimal, the value on the left-hand side of regret cuts becomes smaller than the one determined by the true optimal F_s^* in each round of the iterative relaxation procedure. In this connection, the quality of the regret cut tightening the lower bound of the objective function becomes weak. The best-known makespan under worst-case scenario F_s does not guarantee an optimal condition as meta-heuristics are approximation algorithms and limited computation time is required. Poor solution quality in the worst-case optimisation by meta-heuristics implies poor quality of the final min-max regret solution. In order to obtain close-to-optimal solution in reasonable computation time, a novel EABC algorithm is introduced in this section.

To improve the convergence rate of the ABC algorithm for the robust ASSP model, two enhanced strategies and two novel algorithmic components are introduced. As the computation in the min-max regret optimisation is costly using the iterative relaxation procedure, simplification of the decision variables in the solution representation and elimination of inefficient neighbourhood searching are introduced in the EABC algorithm to reduce the computational burden in the employed bee phase. In addition, we propose two novel components in the EABC algorithm to improve the solution quality. First, a constructive heuristic specifically designed for the mixed-mode runway scheduling under uncertainty. The constructive heuristic provides a fairly good initial solution

as an input and maintains a diversity of the population. The EABC algorithm will then continue to exploit better solution by an iterative process. Second, we adopt an objective-guided updating mechanism in scout bee phase. If a candidate solution c_i is the worst solution in regards to the objective value $fun(c_i)$ and reaches the maximum tolerance of unsuccessful update $trial(c_i) > limit$, during the iterative process of the EABC algorithm, the local optimal solution will be replaced by a memorised solution. Otherwise, it will be restructured by the local search operators to formulate a new solution. In this regard, the objective-guided updating mechanism accelerates the convergence of the algorithm. In the robust optimisation using min-max regret criterion, each iterative relaxation procedures will generate a worst-case optimal or a robust optimal solution. Before the start of the iterative process of the EABC algorithm, the solution from the previous iteration (either a worst-case optimal or a robust optimal solution) will be recorded as memorised solution. The memorised solution in the robust optimisation is obtained from the worst-case optimal/near-optimal solution in the previous iteration, while the memorised solution in the worst-case optimisation is produced from the robust optimal/near-optimal solution in the previous iteration. As a result, the proposed EABC algorithm yields better solutions than other well-known meta-heuristics. Following the description of the components of the EABC algorithm, the computational experiments and the robustness analysis of the algorithm are discussed. The notation of the EABC algorithm is presented in **Table 3**.

Table 3

Notation of the efficient artificial bee colony algorithm

Notations	Explanation
CS	The size of bee colony
SN	The number of candidate solutions
$MaxIter$	The maximum number of iterations
dim	The dimension of an independent solution
$c_{i,} i = 1,2, \dots, SN$	The position of each solution in bee colony
$fun(c_i)$	The objective value of solution c_i
$fit(c_i)$	The fitness value of solution c_i
$Prob(c_i)$	The probability of an individual solution c_i among the entire colony in term of fitness value
$\bar{c_i}$	The neighbour solution of an individual solution c_i
c_m	The memorised solution obtained in previous robust optimisation
$trial(c_i)$	The accumulated trial value of an individual solution c_i, which cannot be enhanced the quality of solution in terms of its objective value
$limit$	The maximum tolerance of $\text{trial}(c_i)$
p	Random number, $0 \leq p < 1$

The pseudo code of the proposed EABC algorithm is shown in **Table 4**. The EABC algorithm initially produces a set of candidate solutions $c_i, i = 1,2, \dots, SN$. Each candidate solution will be replaced by a proposed constructive heuristic, as stated in **Section 3.2.2**, to formulate a "seed" for exploitation and exploration. The constructive heuristic provides a high-quality solution and leads to a fast convergence to close-to-optimal solutions by the EABC algorithm. Then, the EABC algorithm performs searching by the employed bee phase (as described in **Section 3.2.3**), onlooker bee phase (as described in **Section 3.2.4**) and scout bee phase (as described in **Section 3.2.5**) to improve the candidate solutions iteratively until the stopping criterion is met. After the completion of the iterative process, the EABC algorithm reports the best-known solution during the searching process.

Table 4

The pseudo code of the efficient artificial bee colony algorithm

The algorithm architecture of the efficient artificial bee colony algorithm

Initialization

Develop the First-come-first-serve sequence according to the scheduled operation time

Generate an initial solution using constructive heuristic

Compute the objective value $fun(c_i)$ of each solution c_i

Initial the parameter $loop$, and set $MaxIter$

DO

Employed Bee Phase

Apply neighbourhood search on each candidate solution c_i, $\forall$i to construct a neighbourhood solution $\overline{c_i}$

Calculate the objective value $fun(c_i)$ and $fun(\overline{c_i})$ of candidate solution c_i and neighbourhood solution $\overline{c_i}$

Apply greedy selection to construct the new solution

IF the objective value $fun(\overline{c_i})$ is better than $fun(c_i)$

THEN

Replace candidate solution c_i by neighbourhood solution $\overline{c_i}$

ELSE

trial$(c_i) = trial(c_i) + 1$

Onlooker Bee Phase

Calculate the fitness value $fit(c_i)$ of candidate solution c_i using the equation (28)

Generate a uniform random number p and select one candidate solution under roulette wheel selection scheme

Apply neighbourhood search to the corresponding candidate solution

Scout Bee Phase

IF the trial of a candidate solution $limit(c_i)$ is over the maximum tolerance of unsuccessful updates $limit$

THEN

IF the candidate solution c_i is the worst solution among the population

THEN

Replace candidate solution c_i by the memorised solution c_m

ELSE

Perform non-objective guided neighbourhood search to restructure the candidate solution c_i

Stopping criterion

$currentIter = currentIter + 1$

WHILE $currentIter < MaxIter$

Return the best-known solution

3.2.1. Solution representation

The binary decision variables in the MIP model are inefficient when the model is resolved by meta-heuristics, particularly in population-based meta-heuristics, since extra memory and computation time are required for binary searching in such cases. Therefore, the transformation is an essential step to reduce the computational effort in the meta-heuristics formulation. In the robust ASSP model, aircraft assignment and sequencing are conducted by the decision variables x_{ir} and y_{jir}. The assignment and sequencing are determined with regard to the position on the array under the meta-heuristics approach. Assuming that there are three aircraft in the landing system, aircraft 0 is assigned to land immediately before aircraft 1 on runway 0, while aircraft 2 is assigned to land on runway 1. Aircraft 0 and 2 are in the first position of the landing sequence on the corresponding runways. Therefore, the

decision variables x_{00}, x_{10} and x_{21} are equal to 1 in the aircraft assignment problem. As for the aircraft-sequencing problem, the decision variable y_{010} is equal to 1. In order to reduce unnecessary solution space during searching, the solution is represented as a single array with the dimension dim, where $dim = n \times m$. The value of -1 in the solution denotes an empty position in the ASSP solution. The solution representation is shown in **Fig. 2**.

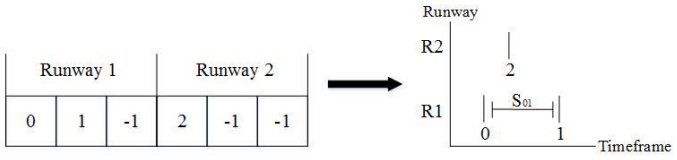

Fig. 2. Solution representation under the meta-heuristics approach

The design of the min-max regret approach aims to obtain a robust ASSP schedule to minimise the maximal regret value by considering a set of worst-case scenarios. The two main stages require optimisation techniques in the solution. They are robust scheduling and makespan optimisation under the worst-case scenario. The process workflow using the ABC algorithm in deriving the worst-case optimal makespan and the robust makespan are the same, except for the computation of objective value and fitness value. The objective function in the optimal makespan under the worst-case scenario is $F(X, s^{\zeta})$, while the objective function in the robust ASSP considering all the worst-case scenario follows Equations (19) and (20).

3.2.2. Initialisation using constructive heuristic

Randomised initialisation in the SI algorithms causes a lengthy convergence time, as a randomised procedure does not guarantee that the searching process started from a promising solution region. SI algorithms perform exploitation and exploration procedures through trial-and-error interaction to obtain close-to-optimal value. The algorithm requires extra effort in searching from an unpromising solution region to a satisfactory solution, which leads to extensive exploitation. To a certain extent, the algorithm undertakes considerable unsuccessful searching and results in frequent termination searching in the exploration phase. With the purpose of minimising the adverse effect in the SI algorithm, a constructive heuristic is a successive approximation scheme to generate a satisfactory initial solution with high quality. A simple constructive heuristic approach is introduced in this section.

The design of the constructive heuristic approach takes a similar perspective of an objective function and the FCFS scheme. The maximum number of flights is equal to n. Flights $i = (1,2, \dots, n)$ are sorted in ascending order as an FCFS scheme using an insertion operator according to its scheduled operation time $STO_i = (STO_1, STO_2, \dots, STO_{n-1}, STO_n)$. The constructive heuristic takes n steps $(1 \leq i \leq n)$ to complete the initial solution. After creating the sequential order of the scheduled operation, each flight is assigned a runway by the greedy method. The first flight is assigned a random runway as a seed. For the remaining flights on the sorted list, the algorithm measures the assigned time of operation T_{ir} if flight i is assigned after the last assigned flight on runway $r = (1 \leq r \leq m)$ in a consecutive sequence, including the separation time requirement. If the assigned time of operation T_{ir} is equal to the scheduled time of operation STO_{ir}, then flight i is successively scheduled on runway r without any delays. Otherwise, the calculation will be repeated and $r = r + 1$ until flight i is assigned or $r = m$. The iterative process will be complete when r is equal to m, which refers to the situation in which flight i requires extra time to perform the operation compared to its preferred time of operation due to limited resources. Two approaches can be applied to resolve this solution – the least completion time method or by random assignment. The runway assignment is selected by measuring the completion time of the runway for m scenarios when flight i is assigned to the latest position of each runway using the least completion time method. Random assignment refers to a situation in which flight i is randomly assigned to a runway. According to our preliminary study, the random assignment approach ensures the diversity of the initial solution, and is able to achieve better optimality in the iterative

relaxation procedure. In our model, the random assignment approach is considered in the constructive heuristic, and the operation performs the next iteration until all flights are assigned.

3.2.3. Employed bee phase

The EABC algorithm is a population-based algorithm and involves searching a set of candidate solutions $c_i = 1,2,\dots,SN$. The employed bee phase aims to generate a neighbourhood solution $\overline{c_i}$ from a known solution c_i by exploitation operators so as to obtain a search of the robust ASSP model for all the solutions c_i. Neighbourhood searching includes local search operators and crossover operators. Local search operators are simple operators to randomly select element(s) in a known solution and revise the structure of a solution. Insert, swap and reverse operators are commonly applied in the employed bee phase in job-shop scheduling and vehicle routing problems (Pan et al., 2013; Schiavinotto and Stützle, 2007; Szeto et al., 2011). The insert operator randomly removes a flight at the i^{th} position and reinserts it into the j^{th} position. The swap operator randomly exchanges two elements at the i^{th} position and j^{th} position. The reverse operator randomly selects a subsequence of flights and reverses the order of the corresponding subsequence. Crossover operators are combining two different candidate solutions to construct a neighbourhood solution (Vallada and Ruiz, 2011). The random element selection falls into the range of $1 \leq i^{th}, j^{th} \leq dim$.

The reverse operator and crossover operator are common local search operators in hybrid meta-heuristics. Crossover is a genetic operator and usually hybridised with other metaheuristics (Gandomi and Alavi, 2012; Pant et al., 2007; Zhang et al., 2016; Zhang et al., 2014). However, these two operators are suggested to be removed if the model requires timely decision-making. In the robust modelling, the iterative relaxation procedure takes several rounds of the optimisation process. The number of iterations in the iterative relaxation procedure is subjected to the number of the worst-case scenarios. To enhance the efficiency of the optimisation process, inefficient operators should be removed. In our prior study, the reverse operator was found to be less effective in improving the solution as the order of a solution represents the sequential time of operation. The neighbourhood solution obtained by reverse operators usually has a higher makespan value. Also, the computation to obtain the neighbourhood solution using a crossover operator requires the measurement of the fitness value on the population and formulates the probabilistic distribution in the wheel roulette selection (Zhang et al., 2014). The computation time of crossover is significant in each iteration. Comparatively, swap and insert operators are simple and effective in obtaining satisfactory neighbourhood solutions.

Four operators are considered in the employed bee phase shown as follows:

- Select a flight and use the insert operator on the same runway for schedule x
- Select a flight and use the insert operator on different runways for schedule x
- Select two flights and use the swap operator on the same runway for schedule x
- Select two flights and use the swap operator on different runways for schedule x

After generating a neighbourhood solution, the objective-guided greedy method is considered in comparing the solution quality between candidate solution c_i and neighbourhood solution $\overline{c_i}$. The quality of both solutions is evaluated by the objective function $\text{fun}(c_i)$. If the solution quality of the neighbourhood solution $\text{fun}(\overline{c_i})$ is better than the original one $\text{fun}(c_i)$, the candidate solution will be updated by the neighbourhood solution. Any unsuccessful update using each operator will be counted for each candidate solution c_i by the parameter $trial(c_i)$.

3.2.4. Onlooker bee phase

The Onlooker bee phase enhances the exploitation process by utilising the fitness probability distribution to improve the overall solution quality. The onlooker bee utilises the information, shared by the employed bee, to further exploit the selected food source.

The selection criterion of high-quality candidate among the candidate solutions depends on the winning probability value p_i. The fitness approximation $fit(c_i)$ is a measurement to identify the quality of each solution according to a cumulative probability distribution of a population set, which deviates from the objective value $\mathrm{fun}(c_i)$ using Equation (21) (Pan et al., 2013). The objective value $fun(c_i)$ in the worst-case optimisation is determined by the makespan value $F(X,s)$ of a schedule under an extreme scenario s, while the objective value $fun(c_i)$ in the robust optimisation is computed by the regret value RV. The proposed Equation (21) is able to handle zero and positive values, as the regret value can be a non-negative value in Equation (3). The larger fitness value $fit(c_i)$ implies a better solution quality across the population. The selective probability of each solution $prob(c_i)$ is derived by Equation (22).

$$fit(c_i) = \frac{1}{1+fun(c_i)}, \forall i \tag{21}$$

$$prob(c_i) = \frac{fit(c_i)}{\sum_{i=1}^{SN} fit(c_i)}, \forall i \tag{22}$$

Due to the simplicity, the selection process in Equation (22) is used in the ABC algorithm (Tasgetiren et al., 2011). Each onlooker bee generates a uniform random number p ranging from [0,1) under the roulette wheel selection scheme, and carries out a neighbourhood search on the selected candidate solution, as stated in Section 3.2.3. The greedy approach is also applied to measure the solution quality between c_i and $\overline{c_i}$. Satisfactory solutions are selected several times in the neighbourhood search procedure, and speeds up the time to convergence in the onlooker bee phase.

3.2.5. Scout bee phase

Candidate solutions will be trapped in local optimum under the intensive neighbourhood searching process, which refers to the degraded searching scenario. The algorithm is not able to obtain a superior neighbourhood solution from the preceding solution in such case. To escape from the local optimum trap in the scout bee phase, the trapped solution is abandoned, and a certain level of solution diversity remains in the population. The selection criteria of the trapped solution are based on the negative feedback mechanism using the parameter $trial$, as mentioned in **Section 3.2.3**. If the number of unsuccessful updates of a solution $trial(c_i)$ exceeds the maximum tolerance of an unsuccessful update, the solution will then be updated by a new solution to maintain diversity among the candidate solutions.

The proposed updating mechanism utilises the information from the prior optimised solution by an iterative relaxation procedure. The design of the updating mechanism insists on enhancing the convergence rate by information exchange (Su et al., 2017). To reduce the computational effort for optimisation in the iterative relaxation procedure, a robust solution in each round of iterations is memorised as a source for a promising search start point, except for the initial solution in the iterative relaxation procedure. Since the optimal solution for the worst-case scenario and the robust optimal solution in the next iteration shares the same sequential elements and structure, the preceding solution can be regarded as a promising and new intermediate solution in the worst-case scenario optimisation and the robust optimisation. The updating mechanism is explained as follows. If one solution is considered as a trapped candidate solution, the algorithm will further evaluate the solution quality regarding the objective value $fit(c_i)$ among the population. If the trapped candidate solution is the worst-observed objective value compared with other candidates, the solution will be replaced by the memorised solution in the previous iteration. Otherwise, non-objective guided insert and swap operators will revise the solution sequence of the trapped candidate solution to restructure the trapped solution. The proposed scout bee phase helps to catalyse the process of intensification and diversification to obtain a preferable or optimal solution until the stopping criteria are satisfied.

4. Result of experiment

4.1. Description of test instances

Since there are no benchmark instance sets of the robust ASSP model in the literature, randomly generated instance sets were adopted in this research. Test instances were generated for the evaluation of the proposed algorithm following the below criteria. The uniform distributions of the scheduled time of operation STO_{ir}^{s} fell into an interval $[\underline{STO_{ir}}, \overline{STO_{ir}}]$. The lower bound value of scheduled time of operation $\underline{STO_{ir}}$ was randomly assigned from an interval of $[0, {}^{60m}/_{n}]$ to represent an arrival/departure rate of flights for a runway. The upper bound value of scheduled time of operation $\overline{STO_{ir}}$ is constructed by the lower bound value of scheduled time of operation STO_{ir} with a time length of runway dependent arrival or departure delay for each flight ε_{ir}, where $\varepsilon_{ir} \in (0, \beta]$. The description of the test instance is shown in **Table 5** with $n = (6,12,18,24,30)$ for two runways – mixed-mode parallel operation, and $n = (10,18,28,36,46)$ for three runways – mixed-mode parallel operation. Flight classes SSF, MSF and LSF represent small-size, medium size and large size flights correspondingly. The distribution of the flight classes for both sets of the test instances are (50% LSF and 50% MSF) and (50% LSF, 33.3% MSF and 16.7% SSF). (60, 120) are used for beta value β, resulting in a total of 40 test instances. β is the maximum deviation of possible operation time ε_{ir} in the computational experiment.

Table 5

Instance description of the robust ASSP model

ASSP model	%LSF=50%, % MSF=50%		%LSF=50%,%MSF=33.3%,%SSF=16.7%	
	Instance ID	n	Instance ID	n
Two runways - mixed mode parallel operation	1001	6	2001	6
	1002	12	2002	12
	1003	18	2003	18
	1004	24	2004	24
	1005	30	2005	30
Three runways - mixed mode parallel operation	3001	10	4001	10
	3002	18	4002	18
	3003	28	4003	28
	3004	36	4004	36
	3005	46	4005	46

SSF=Small size flight; MSF= Medium size flight; LSF=Large size flight

The configuration of the computational environment was an Intel Core i7 3.60 GHz CPU and 16 GB random-access memory under the Windows 7 Enterprise 64-bit operating system. The performance of the EABC algorithm was evaluated by randomly generated ASSP instances and compared to other meta-heuristics and modified ABC algorithms. These included the Branch-and-Bound (B-B) algorithm, Genetic Local Search (GLS) (Liu, 2011), Original Artificial Bee Colony (ABC) algorithm (Karaboga and Basturk, 2008), Modified Artificial Bee Colony (MABC) Algorithm and Hybrid Artificial Bee Colony (HABC) algorithm (Zhang et al., 2014).

Table 6

Algorithm design for comparison

Algorithm design	GLS	ABC	MABC	HABC	EABC
Initialization	Randomized	Randomized	Constructive heuristic	Constructive heuristic	Constructive heuristic
Exploitation	Local search Crossover	Local search	Local search	Local search Crossover	Local search

Exploration	NULL	Scout Bee	Scout bee	Scout bee	Efficient scout bee

The exact method was also applied for baseline results by *IBM ILOG CPLEX Optimization Studio 12.6.3* so as to evaluate the performance of the proposed algorithm. All the algorithms are written in C# language with visual studio 2015. The parameter setting was conducted in the preliminary study for parameter tuning (the detailed parameter analysis is described in **Appendix B**). The final parameters of the proposed EABC algorithm for the worst-case optimisation and the robust optimisation are set as follows:

- $CS = 40, SN = {}^{CS}/_{2}$
- $limit = 2 \times SN \times m \times n$
- $MaxIter = 1000 \times \mathrm{m} \times n$

4.2. Effectiveness of meta-heuristics

Each algorithm was given a maximum computation time of 3,600 seconds to resolve the test instance. The time limit was chosen in accordance with the characteristic of the instance. The number of flights in the large-sized instances is the approximate number of air traffic scenarios during the peak hours in the Hong Kong International Airport. The runway schedule was designed for an hour of traffic. Therefore, the computation time of the schedule must be less than 3,600 seconds to maintain a continuity of runway scheduling in the time horizon. In order to evaluate the algorithm performance, optimality gap $Regret\ Gap$ %, average robust cost ARC, relative percentage increase $RPI(ARC)$, least significant difference LSD and the number of best value $\#best$ are reported.

The robust optimisation in the iterative relaxation procedure measures the regret deviation between $\widehat{RV}$ and $\widetilde{RV}$ as a termination criterion. The optimisation process is not guaranteed to produce a robust solution with the situation of $\widehat{RV} = \widetilde{RV}$ given a limited computation time. Therefore, the optimality gap is reported for the performance of the exact method by Equation (23) (Pereira and Averbakh, 2011).

$$Regret\ Gap\ \% = \frac{\widehat{RV} - \widetilde{RV}}{\widehat{RV}} \tag{23}$$

The robustness cost $RC(X)$ of schedule X is defined as its maximum regret value among all possible scenarios in the set S by Equation (24). 0 value in $Regard\ Gap\ \%$ implies a robust optimal solution and the worst-case scenarios are obtained (Lu et al., 2014). Otherwise, the near-optimal robust solution is obtained. The maximum regret value $Regret_{max}(X_{FCFS})$ using FCFS sequencing policy with mixed-mode runways (FCFS-MIX) was also included in the experiment for comparison. The heuristic of FCFS-MIX sequencing policy followed the rules proposed by Farhadi et al. (2014). The construction of a FCFS-MIX schedule considers the lower bound value of scheduled time of operation STO_{ir} as an input source.

$$RC(X) = Regret_{max}(X) = \max_{s \in \delta} Regret(X, s) = \max_{s \in \delta}(F(X, s) - F_s^*) \tag{24}$$

Repeatability testing was adopted to measure the effectiveness of the proposed algorithm under the same condition as the selected instances, such as delay time, and scheduled operation time of each flight. The RC obtained by the exact method provides a reference point for the evaluation of the meta-heuristics algorithms, comparing the deviation of the maximum regret value with the average regret value. The meta-heuristics approaches generate a robust near optimal solution with less computation effort. In each iteration, the F_s^* obtained is not guaranteed to be an optimal condition under worst-case scenario s. Thus, the maximum regret value $Regret_{max}(X)$ constructed by meta-heuristics is an approximate value. The performance of the proposed algorithm was measured by the $RPI(ARC)$, as shown in Equation (25), given that the RC^* is the maximum regret value obtained by the exact method with

limited computational power. The average Robustness cost ARC is the average performance of the maximum regret value obtained by the meta-heuristics approach in ten repeated experiments.

$$RPI(ARC) = \frac{ARC - RC^*}{RC^*} \times 100 \quad (25)$$

The LSD intervals for the mean estimation of the "noise" in the computational results is needed to provide valid mean comparisons. Equation (26) measures the LSD intervals at a 95% confidence level for estimating the statistically significant difference of the RPI among the test instances (Pan et al., 2013). Since the computation time for optimisation is limited to 3,600 seconds, the RC^* is not guaranteed to be an optimal condition. If ARC is better than the RC^*, $RPI(ARC)$ will be a negative value to indicate better algorithm performance. In this regard, the lower bound value of LSD may also be negative.

$$LSD \in [RPI(ARC) - 1.96\frac{\sigma_{RPI(RC)}}{\sqrt{n}}, RPI(ARC) + 1.96\frac{\sigma_{RPI(RC)}}{\sqrt{n}}] \quad (26)$$

With the purpose of measuring the robustness of the proposed algorithm, the $\#best$ reports the best number of $RC(X)$ values out of the solutions obtained by the exact method, given 10 repeated experiments.

4.3. Computational results with meta-heuristics

With the aim of evaluating the effectiveness of the proposed algorithm and other meta-heuristics, the computational results by the exact method is a reference or baseline for comparison of the deviation of $RC(X)$ and computation time. The computational results by the exact method using *CPLEX* are presented in **Table 7**. *CPLEX* failed to obtain the robustness cost for the instances with less than or equal to 18 flights, given a 3,600-second computational limit. Except the InstanceID-1005-β-60 and InstanceID-2004-β-60, the $Regret\ Gap\ \%$ is at least 20% from the regret lower bound.

Table 7

Computational performance by the exact method

Instance ID	β	CPU (s)	$\widehat{RV}$	Regret Gap %	Instance ID	β	CPU (s)	$\widehat{RV}$	Regret Gap %
1001	60	3.38	30	*Opt*	3001	60	108.08	0	*Opt*
	120	3.24	55	*Opt*		120	31.23	14	*Opt*
1002	60	14.54	9	*Opt*	3002	60	>3600	9	*Opt*
	120	1166.73	18	*Opt*		120	>3600	9	*Opt*
1003	60	>3600	2	100%	3003	60	>3600	36	*Opt*
	120	>3600	123	96.75%		120	>3600	66	71.21%
1004	60	>3600	108	67.59%	3004	60	>3600	79	88.61%
	120	>3600	102	65.69%		120	>3600	68	91.18%
1005	60	>3600	12	*Opt*	3005	60	>3600	61	90.16%
	120	>3600	183	62.84%		120	>3600	98	74.49%
2001	60	1.31	0	*Opt*	4001	60	37.89	23	*Opt*
	120	4.04	13	*Opt*		120	158.03	59	*Opt*
2002	60	1312.40	19	*Opt*	4002	60	>3600	25	96.00%
	120	2636.59	28	*Opt*		120	>3600	62	91.94%

2003	60	>3600	14	64.29%	4003	60	>3600	31	41.94%
	120	>3600	65	92.31%		120	>3600	88	100.00%
2004	60	>3600	67	47.76%	4004	60	>3600	16	*Opt*
	120	>3600	87	62.07%		120	>3600	77	66.23%
2005	60	>3600	48	50.00%	4005	60	>3600	94	21.28%
	120	>3600	130	40.77%		120	>3600	220	39.09%

Opt: Optimal condition

We then investigated the effectiveness of the meta-heuristics in contrast with the performance of the exact method. **Table 8** shows the computational performance of the meta-heuristics for the instances of two and three runways' mixed-mode parallel operations respectively. A total of ten replications for each instance were conducted to calculate the average performance of the algorithm. The average robustness cost ARC and the deviation between the average value and optimal $RPI(ARC)$ are reported. The closer value of $RPI(ARC)$ indicates less deviation from the optimal value. If RC^* is in an optimal condition, the value of $RPI(ARC)$ must be larger than or equal to 0.00%. Negative values in $RPI(ARC)$ indicate that meta-heuristics obtain a better solution than the exact method. Concurrently, RC^* was not in the optimal condition. The bold values in $RPI(ARC)$ are the best values or have the smallest gap from the optimal compared to other meta-heuristics. It can be seen that the EABC algorithm achieved the best performance in $RPI(ARC)$ generally. It can be observed that the smallest $RPI(ARC)$ compared with other meta-heuristic approaches was counted as 24 out of the 40 instances. The average $RPI(ARC)$ of EBAC algorithm is equal to 3.40%, which is smaller than others. The result presents that the $RPI(ARC)$ of the proposed EABC algorithm achieved the best value over the other meta-heuristics approaches.

Table 8

Comparison of FCFS-MIX heuristic and meta-heuristics for the instances of two / three runways' mixed-mode parallel operation

β	Instance ID	Average RC gap $RPI(ARC)$ for two runways system (in percentage)						Instance ID	Average RC gap $RPI(ARC)$ for three runways system (in percentage)					
		FCFS	GLS	ABC	MABC	HABC	EABC		FCFS	GLS	ABC	MABC	HABC	EABC
60	1001	236.67	10.50	7.39	**0.00**	**0.00**	1.42	3001	15500.00	29.24	7.04	6.71	**4.80**	7.04
120		76.36	37.77	7.70	3.00	**1.59**	5.73		1064.29	28.11	8.18	6.76	5.84	**5.54**
60	1002	2755.56	13.05	**1.59**	4.44	4.03	3.98	3002	2088.89	14.37	7.81	10.91	7.48	**6.00**
120		1355.56	14.01	8.98	1.73	1.01	**0.88**		2133.33	25.85	12.08	9.81	**6.32**	8.68
60	1003	25300.00	9.79	11.83	4.84	3.74	**1.03**	3003	1144.44	14.86	8.20	4.09	3.09	**1.65**
120		323.58	18.86	10.83	7.60	7.82	**5.16**		562.12	16.82	7.71	3.88	2.37	**2.36**
60	1004	538.89	13.83	**4.19**	4.80	4.98	**4.19**	3004	677.22	10.28	**-1.35**	3.18	2.49	**-1.35**
120		609.80	12.12	7.96	7.14	7.82	**6.18**		777.94	16.62	0.78	1.04	0.14	**-0.70**
60	1005	6675.00	5.26	0.38	0.38	1.52	**0.00**	3005	845.90	3.97	1.60	1.78	0.89	**0.54**
120		340.98	11.27	3.13	4.35	3.92	**2.55**		552.04	4.01	6.53	**-0.38**	**-0.38**	0.62
60	2001	2600.00	12.05	0.52	**0.00**	**0.00**	**0.00**	4001	486.96	22.10	8.12	8.02	8.67	**7.73**
120		223.08	23.26	4.82	5.42	**2.17**	5.39		164.41	34.98	13.88	13.54	13.46	**11.83**
60	2002	547.37	3.97	8.32	**1.04**	3.37	3.37	4002	1252.00	10.64	**4.11**	**4.11**	7.52	**4.11**
120		357.14	16.75	6.53	**6.03**	6.82	7.48		493.55	16.01	9.40	4.15	**3.68**	5.25
60	2003	3421.43	4.25	**0.48**	4.21	1.67	**0.48**	4003	1058.06	9.71	**1.93**	2.67	2.86	4.33

120		676.92	11.45	21.60	4.89	5.45	**4.16**		368.18	18.96	11.49	6.13	**3.83**	5.11
60	2004	568.66	3.84	**1.13**	6.18	3.14	**1.13**	4004	3987.50	5.18	**0.09**	1.58	0.76	0.26
120		414.94	0.49	**-0.84**	1.42	5.14	0.64		707.79	12.49	4.39	1.84	0.33	**-1.29**
60	2005	1356.25	15.48	10.94	4.05	1.97	**1.28**	4005	824.47	12.05	8.72	0.85	**0.01**	0.09
120		464.62	6.55	14.20	7.56	5.56	**4.43**		275.00	6.09	7.73	0.09	**-0.20**	0.25
Average		2442.14	12.23	6.59	3.95	3.59	**2.98**		1748.20	15.62	6.42	4.54	3.70	**3.40**

Bold value: The minimum RPI(ARC) across the same instance

To demonstrate the statistical difference between the meta-heuristics regarding solution quality $RPI(ARC)$, the LSD intervals for different algorithms are presented in **Table 9**. **Fig. 3** shows the LSD intervals at a 95% confidence level with the LSD factors and algorithm type. The results show that the LSD intervals of the GLS and ABC algorithms have a wide range, which implies that the algorithms fail to obtain satisfactory and steady performance across the selected instances. It is clear from **Fig. 3** that the HABC and EABC are significantly better than other meta-heuristics approaches. HABC and EABC algorithms perform similarly using the measurement of LSD intervals. Besides the solution quality of the proposed algorithm, the computational burden is also a major indicator in determining the efficiency of the algorithm. The average computation time of the HABC algorithm is around triple the results for the EABC algorithm. Considering the computational effort as well as the algorithm performance, the performance of the EABC algorithm surpassed the other types of ABC approaches. Detailed information on the computation time is given in **Appendix A**. It can be concluded that the EBC algorithm improved the average $RPI(ARC)$ and computation time.

Table 9

Comparison of the least significant difference

LSD interval at a 95% confidence level	GLS	ABC	MABC	HABC	EABC
Maximum max $RPI(ARC)$	37.77%	21.60%	13.54%	13.46%	11.83%
Minimum min $RPI(ARC)$	0.49%	-1.35%	-0.38%	-0.38%	-1.35%
Mean $\overline{RPI(ARC)}$	13.92%	6.50%	4.25%	3.64%	3.19%
Standard deviation $\sigma_{RPI(RC)}$	8.57%	4.89%	3.17%	3.04%	3.06%
Upper bound of Least Significant Different $\overline{\text{LSD}}$	30.71%	16.08%	10.46%	9.59%	9.19%
Lower bound of Least Significant Different $\underline{\text{LSD}}$	-2.87%	-3.07%	-1.97%	-2.31%	-2.82%
Average CPU (sec)	153.32	25.92	103.00	188.39	56.61

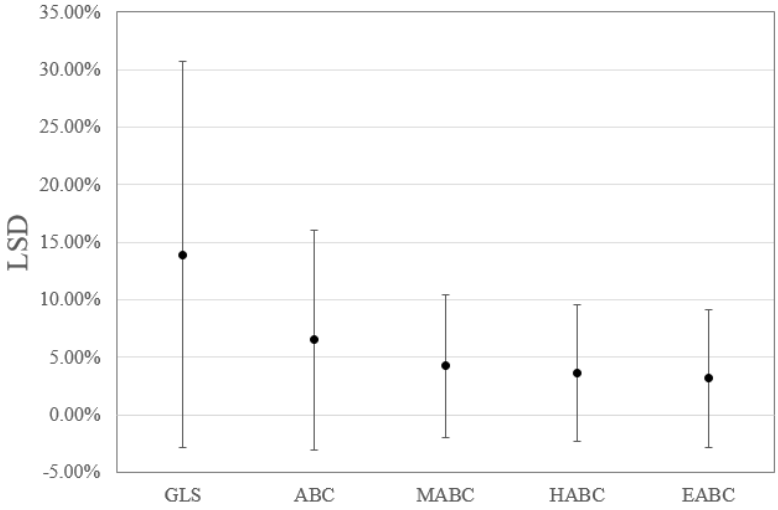

Fig. 3. Means and 95% LSD intervals for different meta-heuristics approaches

The number of best solutions out of the optimal performance are also important factors in the replication testing. Each algorithm was repeated ten times for each instance. The $\#best$ performance is shown in **Table 10**. The $\#\ best$ parameter counts the number reaching optimal or better than the RC^* under limited computational power, if the $RC(X)$ outperformed the result by the exact method. The average values of $\#\ best$ for the GLS, ABC, MABC, HABC and EABC algorithms are 0.70, 0.75, 1.65, 1.70 and 2.55 respectively. The results indicate that the EABC algorithm is a fairly good approximation algorithm for the robust ASSP model.

Table 10

The number of best solutions out of the optimal performance

Instance ID	β	$\#best$					Instance ID	β	$\#best$				
		GLS	ABC	MABC	HABC	EABC			GLS	ABC	MABC	HABC	EABC
1001	60	0	0	**10**	**10**	9	3001	60	0	0	**1**	**1**	**1**
	120	0	0	6	**9**	3		120	0	0	0	0	0
1002	60	0	0	0	0	0	3002	60	0	0	0	0	0
	120	0	0	2	2	**5**		120	0	0	0	0	0
1003	60	**1**	0	0	0	0	3003	60	0	0	0	0	**3**
	120	0	0	0	0	0		120	0	0	1	0	0
1004	60	0	0	0	0	0	3004	60	0	10	0	0	**10**
	120	0	0	0	0	0		120	0	0	1	**6**	**6**
1005	60	0	0	0	0	**5**	3005	60	0	0	0	1	**5**
	120	0	0	0	0	0		120	1	0	**8**	6	4
2001	60	2	9	**10**	**10**	**10**	4001	60	**1**	0	**1**	0	**1**
	120	1	1	1	**4**	0		120	0	0	0	0	**1**
2002	60	2	0	**7**	0	0	4002	60	0	0	0	0	0
	120	**1**	0	0	0	0		120	**1**	0	0	0	**1**
2003	60	0	0	0	0	0	4003	60	0	0	0	0	0
	120	**8**	0	0	0	2		120	0	0	0	0	0
2004	60	0	0	0	0	0	4004	60	0	0	3	4	**7**
	120	9	**10**	4	0	6		120	0	0	1	3	**9**
2005	60	0	0	0	0	0	4005	60	0	0	3	4	**9**
	120	0	0	0	0	0		120	1	0	7	**8**	5
Average		1.20	1.00	**2.00**	1.75	**2.00**	Average		0.20	0.50	1.30	1.65	**3.10**

Bold value: The maximum number of #best across the same instance

In the computational experiment, we evaluated the proposed algorithm using the measurement of $RPI(ARC)$, LSD intervals at a 95% confidence level, average CPU and $\#best$. The EABC algorithm was able to obtain a robust solution with 3.40% deviation from optimal with a minute of computation time on average. Our results suggested that the proposed algorithm is overall beneficial for practical usage and provides close-to-optimal results in a one-hour air traffic situation.

4.4. Discussion on the computational results

The conclusions are restricted to the setting of the test instances. In accordance with the algorithm design, the enhanced strategies

and novel algorithmic components contribute to the algorithm efficiency. The constructive heuristic contributes a 2.25% improvement on average RC gap by comparing the mean $\overline{RPI(ARC)}$ of ABC and MABC algorithms. Regarding the hybridisation of GA and MABC algorithm, there is a 0.61% improvement of the mean $\overline{RPI(ARC)}$ with extra-computation time significantly. We notice that the computation time in GLS and HABC algorithms is costly when crossover operator is applied. In order to simplify the computational burden and remain the same searching capability, an objective-guided updating mechanism is introduced in the scout bee phase and crossover operator is eliminated. Although there is no statistical different between the HABC and the EABC regarding the solution quality, the computation time of the EABC algorithm is 232% improvement than the HABC algorithm.

In general, the proposed EABC algorithm outperforms other meta-heuristic algorithms to achieve better solution quality and satisfactory computation time. Particularly, the EABC algorithm yields close-to-optimal solutions with average one minute computation time comparing to the benchmarking solutions obtained by an exact method with one hour computational limit. As for the test instances with more than 18 flights $(n \geq 18)$, exact algorithm is not able to obtain global optimal results within one hour. The computational results suggest that exact algorithm would be more preferable when the number of flights is below 12 in two/three runways system. Otherwise, the proposed EABC algorithm is recommended for practical usage. The average values of RC gap obtained by the EABC algorithm for two and three runways system are 2.98% and 3.40%. Among 24 out of the 40 test instances, the EABC algorithm computes the best average RC gap over ten runtimes compared to the results obtained by FCFS policy, GLS and variants of ABC algorithms. Therefore, we can conclude that the performance of EABC algorithm surpasses the meta-heuristics and commercial MILP solver.

5. Concluding remarks

This paper investigates the potential of using the min-max regret approach for the mixed-mode parallel runway operation with arrival and departure delays using the swarm intelligence-based algorithm. Compared to solely landing or take-off runways, mixed-mode parallel operation enhances the runway capacity given that the arrival and departure rates per hour in an airport are not equal and both operations can be performed for all the runways. The capacity of the runways can be enhanced by allowing landing and take-off on the same runway schedule, but the conservatism in handling airborne and airport traffic in such operations should be increased, as any accident due to the improper runway usage causes dramatic loss and disruption to the airport management. Arrival and departure delays in the aircraft sequencing and scheduling problem are common phenomena in air traffic control and operation. Current research still focuses on the reassignment method or ground delay programs to alleviate and partially absorb the effect of disrupted scheduling and passenger unease. However, the effect of aggregate delays should not be underestimated with the rising air traffic demand. In practice, the arrival/departure time of flights is uncertain and cannot be estimated in advance by a satisfactory probability distribution. The delay costs caused by rising air traffic demands includes administration costs for ASSP reassignment, the ripple effect on subsequent flight scheduling, the financial cost of delayed management and passenger dissatisfaction. The approaching and leaving time of an aircraft may deviate from the scheduled operation time due to bad weather conditions, congested terminal airspace and aviation system delays. We believe that enhancing the robustness of a schedule can offset the uncontrollable factors and schedule disruption. The Min-max regret approach is a risk-neutral decision, whereby flight scheduling can be performed under uncertain environments. Robust ASSP scheduling with the min-max regret criteria is introduced herein to obtain a robust schedule that considers the worst-case scenarios.

Regarding the solution procedure of the robust optimisation using the min-max regret approach, the proposed efficient artificial bee colony algorithm can be a benefit to ATC to obtain the close-to-optimal schedules within a reasonable computation time for practical usage. It is difficult to obtain a solution from large-sized instances by an exact algorithm, and therefore an efficient artificial bee colony algorithm is proposed to solve the robust aircraft sequencing and scheduling problem with arrival and departure delays for

daily operation. The computational results demonstrated the effectiveness of the proposed algorithm by comparison with other meta-heuristic approaches on generated instances. The proposed algorithm outperformed other meta-heuristic approaches regarding objective function and computation time.

Several interesting aspects can be considered for future work. First, the uncertainty environment and the level of resilience of the robust model can be extended. For instance, flight cancellation and emergency landing can be investigated to enhance the robustness of the model under the extreme weather. Second, the definition of the robust criteria or worst-case scenario can cover other stakeholders' interests. For example, fairness in developing a robust ASSP schedule from the viewpoint of airlines can be considered. Third, investigation of other meta-heuristics in robust optimisation can be studied.

Appendix A. Mathematical proof

Proof of Proposition 1. To obtain a regret value under the worst-case scenario by the equation of $Regret(X, s^{\zeta}) = F(X, s^{\zeta}) - F_s^*$, a feasible solution under the worst-case scenario must satisfy the above two conditions. Denoting that all the aircraft are scheduled on runway r in a landing/take-off sequence X as $P(X, r), r \in R$, the worst-case scenario s^{ζ} can be derived from a scenario s^0 by modifying the scheduled operation time of each flight STO_i^0 under scenario s^o. The regret value is measured as follows:

$$F_{s^{\zeta}}^* - F_{s^0}^* \leq F(X_{s^0}^*, s^{\zeta}) - F(X_{s^0}^*, s^0) \leq \Delta = F(X, s^{\zeta}) - F(X, s^0) \quad (27)$$

The scheduled operation time for the flight $STO_i^{s^0}$ scheduled on critical runway ζ are set to be the upper bound operation time $\overline{STO_i}$ for all $i \in P(X, \zeta), \zeta \in R$, while the scheduled operation time for the flight $STO_i^{s^0}$ scheduled on non-critical runway r is equal to the corresponding lower bound operation times $\underline{STO_i}, i \in P(X, r), r \in R$. The deviation of the objective value or makespan Δ between the worst-case scenario $F(X, s^{\zeta})$ and optimal solution under scenario $F_s^*, s \in \delta$ becomes maximal by the transformation of scheduled operation time of all the flights on the critical runway $P(X, \zeta)$. Hence, the regret value is maximised by manipulating the scheduled operation time STO_i^s in the solution of the worst-case scenario $X_{s^{\zeta}}^*$ compared with the optimal solution under scenario $X_{s^0}^*$ using $F(X, s^{\zeta}) - F(X, s^0)$. □

Proof of Proposition 2. The maximum regret value must be a positive real number in Equation (20). Assume that there exists a runway $r \in R$, $Regret_{max}(X) \leq \max_{\zeta \in R}(C_{\zeta}^{s^{\zeta}} - F_{s^{\zeta}}^*)$. There exists a runway $u \in R$ by contraction such that $Regret_{max}(X) < \max_{u \in R}(C_u^{s^u} - F_{s^u}^*)$. Since $F(X, s^u) \geq C_u^{s^u}$, $Regret_{max}(X) < F(X, s^u) - F_{s^u}^*$, the maximum regret value will be less than 0, which is not feasible in accordance with the definition of $Regret_{max}(X)$. Therefore, Equation (20) holds. □

Appendix B. Parameter analysis of the EABC algorithm

The ABC algorithm, proposed by (Karaboga, 2005), is a swarm intelligence based on the behaviour of natural honeybee swarms. Karaboga and Basturk (2007) compared the ABC algorithm with other meta-heuristics for numerical function optimisation. The discrete ABC algorithm with modifications was applied in jobs-shop scheduling, machine scheduling and aircraft landing problems (Lin and Ying, 2014; Ng and Lee, 2016a; Pan et al., 2013; Tasgetiren et al., 2011; Zhang et al., 2013). The parameters in the ABC algorithm include CS, $limit$ and $MaxIter$. The parameter tuning process is similar to the process in Akay and Karaboga (2009).

The experiments of parameter analysis were conducted by optimising a schedule $F(X_{initial}, s)$, and the scheduled operation time for all the flights STO_{ir} is equal to the corresponding lower bound operation times $\underline{STO_{ir}}$. The computations by the EABC algorithm were repeated in ten runtimes to obtain the average $F(X_{initial}, s)$. The solution gap is calculated by Equation (25). Eight large-sized instances were selected in the parameter evaluation for simplicity.

The CS parameter controls the size of the population in the ABC algorithm. The CS incrementally increased by 5 units until there were no significant changes to the solution gap. **Fig. 4**. presents the computational result by fixing the $limit$ as 2 and $MaxIter$ as $1000 \times m \times n$. After the CS reached 40, the improvement of the solution was less significant along with the increase of CS by calculating the solution gaps for each instance. Therefore, the tuned CS parameter was equal to 40 in our experiments.

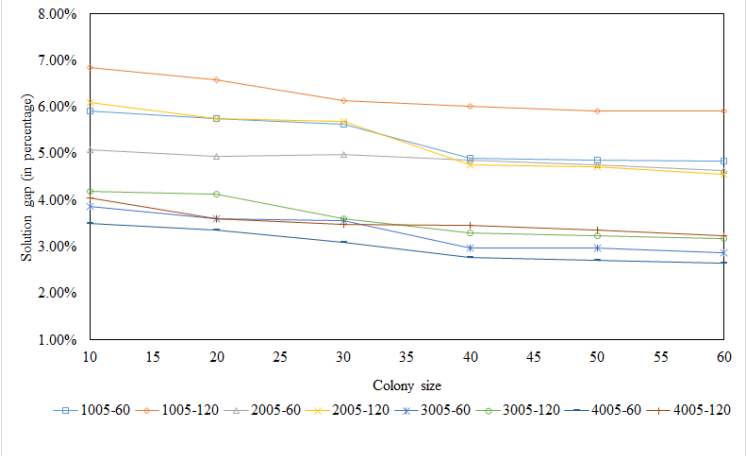

Fig. 4. Effect of colony size parameter

The $limit$ parameter restricted the maximum number of unsuccessful updates by neighbourhood searching. A low value in the $limit$ parameter causes an insufficient tolerance of neighbourhood searching, while a large value in the $limit$ parameter affects the ability of escaping local optima. Therefore, balancing the exploitation and exploration using the $limit$ parameter is able to facilitate the convergence of the searching. The changing of the $limit$ parameter is suggested to be around $SN \times Dim$, where Dim is equal to $m \times n$ (Akay and Karaboga, 2009). In our experiment, we considered $limit$ as $\beta \times m \times n$, where β is the coefficient of $limit$ parameter and $\beta = 0.5, 1, 2, 3, 5, 10$. **Fig. 5.** indicates that when the $limit$ parameter is equal to 2, and the selected instances obtained the low solution gap from the optimal. Therefore, the tuned coefficient of $limit$ parameter β was equal to 2 in our experiments.

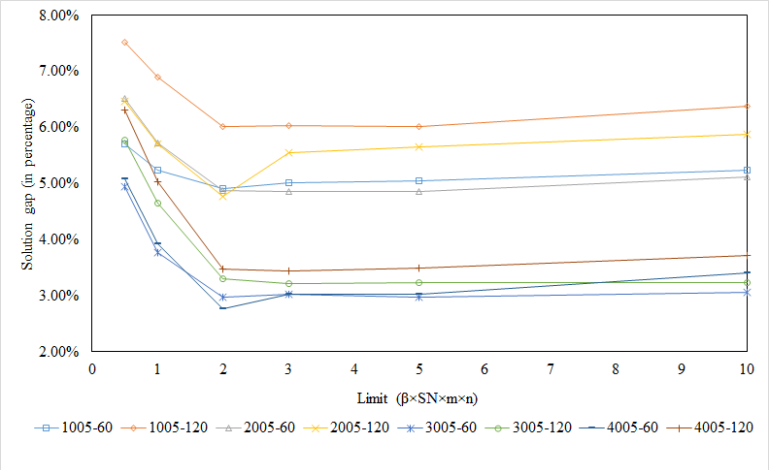

Fig. 5. Effect of limit parameter

The maximum iteration $maxIter$ parameter affects the computation time in the optimisation. Designing an appropriate $maxIter$ parameter provides an efficient convergence process in practical usage. The cycle time is equal to $\alpha \times m \times n$, where α is the coefficient of $maxIter$ parameter and $\alpha = 10, 100, 250, 500, 750, 1000, 1250, 1500, 1750, 2000$. As shown in **Fig.6**., the result shows that the solution gap has no significant changes after $maxIter$ parameter = 1,000. Therefore, the tuned coefficient of $maxIter$ parameter α was set to be 1,000.

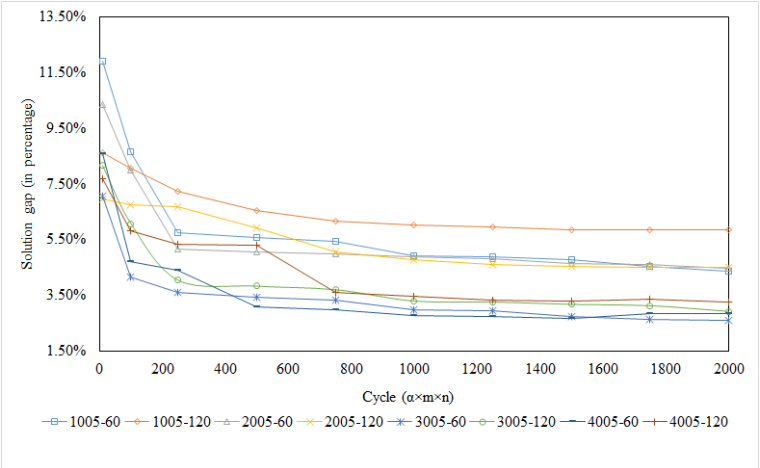

Fig.6. The effect of cycle parameter

In conclusion, the fine-tuned parameters ($CS = 40, limit = 2 \times m \times n, MaxIter = 1000 \times m \times n$) were obtained by the above parameter analysis.

Appendix C. Detailed iteration and average computation time

Table 11

Average iteration and for the instances of two runways' mixed-mode parallel operation

Instance ID	β	Itr					Average CPU (sec)				
		GLS	ABC	MABC	HABC	EABC	GLS	ABC	MABC	HABC	EABC
1001	60	1.00	2.30	2.20	2.20	2.10	6.04	4.26	4.06	8.40	5.09
	120	1.90	2.80	2.30	2.10	2.40	9.37	5.40	4.22	6.60	3.77
1002	60	0.70	0.00	0.30	1.30	0.40	16.76	2.66	3.75	13.45	4.63
	120	1.80	0.70	3.20	3.10	3.50	30.68	5.23	16.30	27.83	24.91
1003	60	1.20	0.10	0.40	0.40	0.80	41.74	6.42	11.36	12.39	10.63
	120	2.10	0.00	3.20	3.30	2.40	70.75	6.93	48.86	53.93	24.41
1004	60	1.20	0.00	0.20	0.20	0.00	67.45	11.49	10.82	16.31	8.93
	120	2.40	0.00	1.20	2.40	2.20	120.88	11.29	26.07	70.73	39.59
1005	60	0.40	0.00	0.00	0.00	0.30	56.88	17.17	11.99	17.55	17.83
	120	2.30	0.00	0.40	0.00	0.10	170.45	15.64	20.42	137.54	15.65
2001	60	1.30	2.50	2.30	2.60	2.20	7.02	4.79	4.35	9.38	5.17
	120	3.00	2.80	2.90	2.90	2.10	16.97	5.46	5.66	9.20	4.49
2002	60	0.20	0.50	0.00	0.00	0.00	11.28	4.73	2.87	4.68	3.42
	120	2.40	1.90	2.40	3.10	1.20	38.77	10.24	16.62	29.05	11.59
2003	60	0.10	0.00	0.10	0.00	0.30	16.98	6.30	8.34	6.90	8.52
	120	1.30	0.00	2.60	2.60	2.00	40.41	10.53	44.42	44.56	27.48
2004	60	0.20	0.00	0.10	0.10	0.00	30.67	9.15	12.93	14.05	9.15
	120	0.40	0.00	0.40	1.30	0.50	28.61	11.25	16.59	42.97	17.38
2005	60	0.60	0.10	0.10	0.00	0.00	70.45	17.79	15.56	17.60	12.81
	120	3.00	0.10	1.60	3.10	2.30	222.35	17.97	40.24	150.62	56.14
Average		1.38	0.69	1.30	1.54	1.24	53.73	9.24	16.27	34.69	15.58

Itr: Iteration

Table 12

Average iteration and computation time for the instances of three runways' mixed-mode parallel operation

Instance ID	β	Itr					Average CPU (sec)				
		GLS	ABC	MABC	HABC	EABC	GLS	ABC	MABC	HABC	EABC
3001	60	1.10	3.00	2.80	3.60	2.60	32.89	32.32	18.00	40.17	14.56
	120	1.50	4.20	3.80	3.50	3.00	32.37	36.78	23.29	39.92	16.46
3002	60	1.30	0.50	1.10	3.10	1.80	86.30	17.01	23.47	101.51	27.55
	120	2.20	4.20	4.30	5.00	4.40	127.90	83.84	73.92	160.54	63.80
3003	60	1.20	0.10	0.50	3.40	0.30	162.10	18.37	29.99	227.71	21.48
	120	2.80	3.50	4.20	4.50	4.40	304.53	154.89	136.56	324.84	124.71
3004	60	0.60	0.00	0.50	1.40	0.00	154.21	27.42	36.93	171.84	22.06

	120	2.00	0.00	4.70	3.60	3.90	361.79	27.64	253.86	417.25	229.42
3005	60	0.60	0.00	4.40	3.90	1.30	249.55	40.90	537.98	768.33	165.63
	120	2.70	0.10	6.40	5.10	4.60	741.68	48.61	806.27	989.57	309.24
4001	60	1.60	3.50	2.50	3.50	2.70	40.74	33.88	15.67	40.56	14.34
	120	2.00	4.90	5.20	4.30	4.30	45.39	38.22	31.86	49.12	25.48
4002	60	0.30	0.00	0.00	0.00	0.00	158.52	9.64	7.81	14.02	7.01
	120	1.20	0.30	2.20	3.20	1.70	73.49	14.21	41.16	107.83	29.15
4003	60	1.40	0.00	0.30	2.30	0.90	193.71	18.06	24.32	183.22	37.47
	120	2.30	1.50	5.80	6.40	6.00	251.88	86.45	194.03	460.31	212.68
4004	60	1.20	0.00	2.40	2.40	2.00	230.59	27.37	225.74	297.14	88.55
	120	2.90	0.00	4.40	4.60	4.60	632.91	27.44	255.14	553.03	258.44
4005	60	1.90	0.10	3.90	3.40	0.10	548.32	50.90	420.01	681.81	36.70
	120	2.20	0.20	6.10	6.10	4.60	629.59	58.08	638.54	1213.12	248.25
Average		1.65	1.31	3.28	3.67	2.66	252.92	42.60	189.73	342.09	97.65

Itr: Iteration

References

Aissi, H., Bazgan, C., Vanderpooten, D., 2009. Min–max and min–max regret versions of combinatorial optimization problems: A survey. *European Journal of Operational Research* 197(2), 427-438.

Akay, B., Karaboga, D., 2009. Parameter Tuning for the Artificial Bee Colony Algorithm, In: Nguyen, N.T., Kowalczyk, R., Chen, S.-M. (Eds.), *Computational Collective Intelligence. Semantic Web, Social Networks and Multiagent Systems: First International Conference, ICCCI 2009, Wrocław, Poland, October 5-7, 2009. Proceedings*. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 608-619.

Artiouchine, K., Baptiste, P., Dürr, C., 2008. Runway sequencing with holding patterns. *European Journal of Operational Research* 189(3), 1254-1266.

Atkin, J.A.D., Burke, E.K., Greenwood, J.S., Reeson, D., 2008. On-line decision support for take-off runway scheduling with uncertain taxi times at London Heathrow airport. *Journal of Scheduling* 11(5), 323.

Balakrishnan, H., Chandran, B.G., 2010. Algorithms for Scheduling Runway Operations Under Constrained Position Shifting. *Operations Research* 58(6), 1650-1665.

Ball, M., Barnhart, C., Dresner, M., Hansen, M., Neels, K., Odoni, A., Peterson, E., Sherry, L., Trani, A., Zou, B., 2010. Total delay impact study : a comprehensive assessment of the costs and impacts of flight delay in the United States. NEXTOR Report Prepared for the Federal Aviation Administration.

Basso, L.J., 2008. Airport deregulation: Effects on pricing and capacity. *International Journal of Industrial Organization* 26(4), 1015-1031.

Beasley, J.E., Krishnamoorthy, M., Sharaiha, Y.M., Abramson, D., 2000. Scheduling Aircraft Landings—The Static Case. *Transportation Science* 34(2), 180-197.

Beasley, J.E., Krishnamoorthy, M., Sharaiha, Y.M., Abramson, D., 2004. Displacement problem and dynamically scheduling aircraft landings. *Journal of the Operational Research Society* 55(1), 54-64.

Beasley, J.E., Sonander, J., Havelock, P., 2001. Scheduling Aircraft Landings at London Heathrow Using a Population Heuristic. *The Journal of the Operational Research Society* 52(5), 483-493.

Beatty, R., Hsu, R., Berry, L., Rome, J., 1999. Preliminary Evaluation of Flight Delay Propagation through an Airline Schedule. *Air Traffic Control Quarterly* 7(4), 259-270.

Bell, D.E., 1982. Regret in Decision Making under Uncertainty. *Operations Research* 30(5), 961-981.

Bencheikh, G., Boukachour, J., Alaoui, A.E.H., Khoukhi, F., 2009. Hybrid method for aircraft landing scheduling based on a job shop formulation. *International Journal of Computer Science and Network Security* 9(8), 78-88.

Bennell, J.A., Mesgarpour, M., Potts, C.N., 2011. Airport runway scheduling. *4OR* 9(2), 115.

Bennell, J.A., Mesgarpour, M., Potts, C.N., 2017. Dynamic scheduling of aircraft landings. *European Journal of Operational Research* 258(1), 315-327.

Bianchi, L., Dorigo, M., Gambardella, L.M., Gutjahr, W.J., 2009. A survey on metaheuristics for stochastic combinatorial optimization. *Natural Computing* 8(2), 239-287.

Bianco, L., Dell'Olmo, P., Giordani, S., 1997. Scheduling Models and Algorithms for TMA Traffic Management, In: Bianco, L., Dell'Olmo, P., Odoni, A.R. (Eds.), *Modelling and Simulation in Air Traffic Management*. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 139-167.

Campanelli, B., Fleurquin, P., Arranz, A., Etxebarria, I., Ciruelos, C., Eguíluz, V.M., Ramasco, J.J., 2016. Comparing the modeling of delay propagation in the US and European air traffic networks. *Journal of Air Transport Management* 56, 12-18.

Caprì, S., Ignaccolo, M., 2004. Genetic algorithms for solving the aircraft-sequencing problem: the introduction of departures into the dynamic model. *Journal of Air Transport Management* 10(5), 345-351.

Churchill, A., Lovell, D., Ball, M., 2010. Flight Delay Propagation Impact on Strategic Air Traffic Flow Management.

Transportation Research Record: Journal of the Transportation Research Board 2177, 105-113.

D'Ariano, A., Pacciarelli, D., Pistelli, M., Pranzo, M., 2015. Real-time scheduling of aircraft arrivals and departures in a terminal maneuvering area. *Networks* 65(3), 212-227.

D'Ariano, A., Pistelli, M., Pacciarelli, D., 2012. Aircraft retiming and rerouting in vicinity of airports. *IET Intelligent Transport Systems* 6(4), 433-443.

Dastgerdi, K., Mehrshad, N., Farshad, M., 2015. A New Intelligent Approach to Aircrafts Take-off/Landing Planning at Congested Single Runway Airports. *Journal of Soft Computing and Decision Support Systems* 2(2), 17-25.

Dear, R.G., Sherif, Y.S., 1989. The dynamic scheduling of aircraft in high density terminal areas. *Microelectronics Reliability* 29(5), 743-749.

Eun, Y., Hwang, I., Bang, H., 2010. Optimal Arrival Flight Sequencing and Scheduling Using Discrete Airborne Delays. *IEEE Transactions on Intelligent Transportation Systems* 11(2), 359-373.

Farhadi, F., Ghoniem, A., Al-Salem, M., 2014. Runway capacity management – An empirical study with application to Doha International Airport. *Transportation Research Part E: Logistics and Transportation Review* 68, 53-63.

Feng, X., Zheng, F., Xu, Y., 2016. Robust scheduling of a two-stage hybrid flow shop with uncertain interval processing times. *International Journal of Production Research* 54(12), 3706-3717.

Francis, G., Humphreys, I., Ison, S., 2004. Airports' perspectives on the growth of low-cost airlines and the remodeling of the airport–airline relationship. *Tourism Management* 25(4), 507-514.

Gandomi, A.H., Alavi, A.H., 2012. Krill herd: A new bio-inspired optimization algorithm. *Communications in Nonlinear Science and Numerical Simulation* 17(12), 4831-4845.

Garey, M.R., Johnson, D.S., 1979. *Computers and Intractability: A Guide to the Theory of NP-completeness*. W. H. Freeman.

Gelhausen, M.C., Berster, P., Wilken, D., 2013. Do airport capacity constraints have a serious impact on the future development of air traffic? *Journal of Air Transport Management* 28, 3-13.

Ghoniem, A., Sherali, H.D., Baik, H., 2014. Enhanced Models for a Mixed Arrival-Departure Aircraft Sequencing Problem. *INFORMS Journal on Computing* 26(3), 514-530.

Glover, F., 1986. Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research* 13(5), 533-549.

Hancerliogullari, G., Rabadi, G., Al-Salem, A.H., Kharbeche, M., 2013. Greedy algorithms and metaheuristics for a multiple runway combined arrival-departure aircraft sequencing problem. *Journal of Air Transport Management* 32, 39-48.

Hansen, J.V., 2004. Genetic search methods in air traffic control. *Computers & Operations Research* 31(3), 445-459.

Hansen, M., Zou, B., 2013. Airport Operational Performance and Its Impact on Airline Cost, *Modelling and Managing Airport Performance*. John Wiley & Sons, pp. 119-143.

Harikiopoulo, D., Neogi, N., 2011. Polynomial-Time Feasibility Condition for Multiclass Aircraft Sequencing on a Single-Runway Airport. *IEEE Transactions on Intelligent Transportation Systems* 12(1), 2-14.

Hu, H., Ng, K.K.H., Qin, Y., 2016. Robust Parallel Machine Scheduling Problem with Uncertainties and Sequence-Dependent Setup Time. *Scientific Programming* 2016, 13.

Inuiguchi, M., Sakawa, M., 1995. Minimax regret solution to linear programming problems with an interval objective function. *European Journal of Operational Research* 86(3), 526-536.

Jacquillat, A., Odoni, A.R., 2015a. Endogenous control of service rates in stochastic and dynamic queuing models of airport congestion. *Transportation Research Part E: Logistics and Transportation Review* 73, 133-151.

Jacquillat, A., Odoni, A.R., 2015b. An Integrated Scheduling and Operations Approach to Airport Congestion Mitigation. *Operations Research* 63(6), 1390-1410.

Jacquillat, A., Odoni, A.R., Webster, M.D., 2017. Dynamic Control of Runway Configurations and of Arrival and Departure Service

Rates at JFK Airport Under Stochastic Queue Conditions. *Transportation Science* 51(1), 155-176.

Jeanne, R.L., 1986. THE EVOLUTION OF THE ORGANIZATION OF WORK IN SOCIAL INSECTS. *Monitore Zoologico Italiano - Italian Journal of Zoology* 20(2), 119-133.

Jiang, Y., Xu, Z., Xu, X., Liao, Z., Luo, Y., 2014. A Schedule Optimization Model on Multirunway Based on Ant Colony Algorithm. *Mathematical Problems in Engineering* 2014, 11.

Kafle, N., Zou, B., 2016. Modeling flight delay propagation: A new analytical-econometric approach. *Transportation Research Part B: Methodological* 93, 520-542.

Karaboga, D., 2005. An idea based on honey bee swarm for numerical optimization. Technical Report-TR06, Erciyes University, Erciyes University, Kayseri, Turkey, p. 200.

Karaboga, D., Akay, B., 2009. A survey: algorithms simulating bee swarm intelligence. *Artif. Intell. Rev.* 31(1-4), 61-85.

Karaboga, D., Basturk, B., 2007. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. *Journal of Global Optimization* 39(3), 459-471.

Karaboga, D., Basturk, B., 2008. On the performance of artificial bee colony (ABC) algorithm. *Applied Soft Computing* 8(1), 687-697.

Karaboga, D., Gorkemli, B., Ozturk, C., Karaboga, N., 2014. A comprehensive survey: artificial bee colony (ABC) algorithm and applications. *Artificial Intelligence Review* 42(1), 21-57.

Kouvelis, P., Yu, G., 1997. Robust Scheduling Problems, *Robust Discrete Optimization and Its Applications*. Springer US, Boston, MA, pp. 241-289.

Kube, C.R., Bonabeau, E., 2000. Cooperative transport by ants and robots. *Robotics and Autonomous Systems* 30(1), 85-101.

Lieder, A., Stolletz, R., 2016. Scheduling aircraft take-offs and landings on interdependent and heterogeneous runways. *Transportation Research Part E: Logistics and Transportation Review* 88, 167-188.

Lin, S.-W., Ying, K.-C., 2014. ABC-based manufacturing scheduling for unrelated parallel machines with machine-dependent and job sequence-dependent setup times. *Computers & Operations Research* 51, 172-181.

Liu, Y.-H., 2011. A genetic local search algorithm with a threshold accepting mechanism for solving the runway dependent aircraft landing problem. *Optimization Letters* 5(2), 229-245.

Lu, C.-C., Ying, K.-C., Lin, S.-W., 2014. Robust single machine scheduling for minimizing total flow time in the presence of uncertain processing times. *Computers & Industrial Engineering* 74, 102-110.

Ma, W., Xu, B., Liu, M., Huang, H., 2014. An Efficient Approximation Algorithm for Aircraft Arrival Sequencing and Scheduling Problem. *Mathematical Problems in Engineering* 2014, 8.

Mausser, H.E., Laguna, M., 1998. A New Mixed Integer Formulation for the Maximum Regret Problem. *International Transactions in Operational Research* 5(5), 389-403.

Mausser, H.E., Laguna, M., 1999. A heuristic to minimax absolute regret for linear programs with interval objective function coefficients. *European Journal of Operational Research* 117(1), 157-174.

Ng, K.K.H., Lee, C.K.M., 2016a. Makespan minimization in aircraft landing problem under congested traffic situation using modified artificial bee colony algorithm, *2016 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*. IEEE, Bali, Indonesia, pp. 750-754.

Ng, K.K.H., Lee, C.K.M., 2016b. A modified Variable Neighborhood Search for aircraft Landing Problem, *2016 IEEE International Conference on Management of Innovation and Technology (ICMIT)*. IEEE, Bangkok, Thailand, pp. 127-132.

Ng, K.K.H., Lee, C.K.M., 2017. Aircraft Scheduling Considering Discrete Airborne Delay and Holding Pattern in the Near Terminal Area, *Intelligent Computing Theories and Application: 13th International Conference, ICIC 2017, Liverpool, UK*, pp. 567-576.

Pan, Q.K., Wang, L., Mao, K., Zhao, J.H., Zhang, M., 2013. An Effective Artificial Bee Colony Algorithm for a Real-World Hybrid Flowshop Problem in Steelmaking Process. *IEEE Transactions on Automation Science and Engineering* 10(2), 307-322.

Pant, M., Thangaraj, R., Abraham, A., 2007. A New PSO Algorithm with Crossover Operator for Global Optimization Problems, In: Corchado, E., Corchado, J.M., Abraham, A. (Eds.), *Innovations in Hybrid Intelligent Systems*. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 215-222.

Pereira, J., Averbakh, I., 2011. Exact and heuristic algorithms for the interval data robust assignment problem. *Computers & Operations Research* 38(8), 1153-1163.

Pinol, H., Beasley, J.E., 2006. Scatter Search and Bionomic Algorithms for the aircraft landing problem. *European Journal of Operational Research* 171(2), 439-462.

Pyrgiotis, N., Malone, K.M., Odoni, A., 2013. Modelling delay propagation within an airport network. *Transportation Research Part C: Emerging Technologies* 27, 60-75.

Rodríguez-Díaz, A., Adenso-Díaz, B., González-Torre, P.L., 2017. Minimizing deviation from scheduled times in a single mixed-operation runway. *Computers & Operations Research* 78, 193-202.

Sabar, N.R., Kendall, G., 2015. An iterated local search with multiple perturbation operators and time varying perturbation strength for the aircraft landing problem. *Omega* 56, 88-98.

Salehipour, A., Modarres, M., Moslemi Naeni, L., 2013. An efficient hybrid meta-heuristic for aircraft landing problem. *Computers & Operations Research* 40(1), 207-213.

Samà, M., D'Ariano, A., Corman, F., Pacciarelli, D., 2017. Metaheuristics for efficient aircraft scheduling and re-routing at busy terminal control areas. *Transportation Research Part C: Emerging Technologies* 80, 485-511.

Samà, M., D'Ariano, A., D'Ariano, P., Pacciarelli, D., 2014. Optimal aircraft scheduling and routing at a terminal control area during disturbances. *Transportation Research Part C: Emerging Technologies* 47, 61-85.

Samà, M., D'Ariano, A., D'Ariano, P., Pacciarelli, D., 2015. Air traffic optimization models for aircraft delay and travel time minimization in terminal control areas. *Public Transport* 7(3), 321-337.

Samà, M., D'Ariano, A., Pacciarelli, D., 2013. Rolling Horizon Approach for Aircraft Scheduling in the Terminal Control Area of Busy Airports. *Procedia - Social and Behavioral Sciences* 80, 531-552.

Schiavinotto, T., Stützle, T., 2007. A review of metrics on permutations for search landscape analysis. *Computers & Operations Research* 34(10), 3143-3153.

Sinclair, K., Cordeau, J.-F., Laporte, G., 2014. Improvements to a large neighborhood search heuristic for an integrated aircraft and passenger recovery problem. *European Journal of Operational Research* 233(1), 234-245.

Soomer, M., Koole, G., 2008. Fairness in the aircraft landing problem. Vrije Universiteit Amsterdam, Amsterdam, The Netherlands.

Su, W., Chen, H., Liu, F., Lin, N., Jing, S., Liang, X., Liu, W., 2017. A novel comprehensive learning artificial bee colony optimizer for dynamic optimization biological problems. *Saudi Journal of Biological Sciences* 24(3), 695-702.

Szeto, W.Y., Wu, Y., Ho, S.C., 2011. An artificial bee colony algorithm for the capacitated vehicle routing problem. *European Journal of Operational Research* 215(1), 126-135.

Tasgetiren, M.F., Pan, Q.-K., Suganthan, P.N., Chen, A.H.L., 2011. A discrete artificial bee colony algorithm for the total flowtime minimization in permutation flow shops. *Information Sciences* 181(16), 3459-3475.

Trelea, I.C., 2003. The particle swarm optimization algorithm: convergence analysis and parameter selection. *Information Processing Letters* 85(6), 317-325.

Vadlamani, S., Hosseini, S., 2014. A novel heuristic approach for solving aircraft landing problem with single runway. *Journal of Air Transport Management* 40, 144-148.

Vallada, E., Ruiz, R., 2011. A genetic algorithm for the unrelated parallel machine scheduling problem with sequence dependent setup times. *European Journal of Operational Research* 211(3), 612-622.

Xie, J., Zhou, Y., Zheng, H., 2013. A Hybrid Metaheuristic for Multiple Runways Aircraft Landing Problem Based on Bat Algorithm. *Journal of Applied Mathematics* 2013, 8.

Xu, X., Cui, W., Lin, J., Qian, Y., 2013. Robust makespan minimisation in identical parallel machine scheduling problem with interval data. *International Journal of Production Research* 51(12), 3532-3548.

Zhan, Z.H., Zhang, J., Li, Y., Liu, O., Kwok, S.K., Ip, W.H., Kaynak, O., 2010. An Efficient Ant Colony System Based on Receding Horizon Control for the Aircraft Arrival Sequencing and Scheduling Problem. *IEEE Transactions on Intelligent Transportation Systems* 11(2), 399-412.

Zhang, L.L., Lee, C., Zhang, S., 2016. An integrated model for strategic supply chain design: Formulation and ABC-based solution approach. *Expert Systems with Applications* 52, 39-49.

Zhang, R., Song, S., Wu, C., 2013. A hybrid artificial bee colony algorithm for the job shop scheduling problem. *International Journal of Production Economics* 141(1), 167-178.

Zhang, S., Lee, C.K.M., Chan, H.K., Choy, K.L., Wu, Z., 2015. Swarm intelligence applied in green logistics: A literature review. *Engineering Applications of Artificial Intelligence* 37, 154-169.

Zhang, S., Lee, C.K.M., Choy, K.L., Ho, W., Ip, W.H., 2014. Design and development of a hybrid artificial bee colony algorithm for the environmental vehicle routing problem. *Transportation Research Part D: Transport and Environment* 31, 85-99.

Zou, B., Hansen, M., 2012. Impact of operational performance on air carrier cost structure: Evidence from US airlines. *Transportation Research Part E: Logistics and Transportation Review* 48(5), 1032-1048.