processes

MDPI

Article

An Efficient Ant Colony Algorithm Based on Rank 2 Matrix Approximation Method for Aircraft Arrival/Departure Scheduling Problem

Bo Xu [1,*], Weimin Ma [2], Hua Ke [2], Wenjuan Yang [3] and Hao Zhang [1]

[1] Business School, University of Shanghai for Science and Technology, Shanghai 200093, China
[2] School of Economics and Management, Tongji University, Shanghai 200092, China
[3] School of Management Science & Engineering, Nanjing University of Finance & Economics, Nanjing 210023,China
* Correspondence: xubochn@163.com

Abstract: The Aircraft Arrival/Departure Problem (AADSP) is the core problem in current runway system, even has become the bottleneck to prevent the improvement of the airport efficiency. This paper studies the single runway AADSP. A Mixed Integer Programming (MIP) model is constructed and an algorithm named Ant Colony based on Rank 2 Matrix Approximation (RMA-AC) method is proposed. Numerical results validate that the new algorithm, as well as the new model, exhibits better performance than CPLEX and the traditional two-phase algorithm. The runway efficiency enhanced by RMA-AC, within 20 s computation, is about 2–5% even for the 800 aircraft sequences. It is a promising method to improve the efficiency of the future aircraft scheduling system.

Keywords: aircraft scheduling problem; rank 2 matrix approximation; mixed-integer programming

Citation: Xu, B.; Ma, W.; Ke, H.; Yang, W.; Zhang, H. An Efficient Ant Colony Algorithm Based on Rank 2 Matrix Approximation Method for Aircraft Arrival/Departure Scheduling Problem. *Processes* **2022**, *10*, 1825. https://doi.org/10.3390/pr10091825

Academic Editors: Danyu Bai, Xin Chen, Dehua Xu and Jedrzej Musial

Received: 29 August 2022
Accepted: 8 September 2022
Published: 10 September 2022

1. Introduction

With the rapid improvement of social economy and people's living standards, the air traffic flow has continued to grow in recent years (Boeing [1], Federal Aviation Administration [2]). Although the outbreak of COVID-19 has temporarily limited aviation demand, authoritative forecasts indicate that sharp increase of air traffic will appear after the epidemic (Gudmundsson [3]). How to improve the capacity of the air transportation system will continue to be a significant but challenging problem for governments, airlines and passengers.

Many researches and investigations have pointed out that insufficient throughput of runway is the bottleneck of the air transport system (Bennell et al. [4]). However, the construction of new runways is expensive. For example, the construction of the third runway in Wuhan Tianhe International Airport cost 3.08 billion yuan (Wuhan Network Information Office [5]). Therefore, many experts and scholars attempt to make the most use of the existed runway, that is, to sequence the aircraft on the runway to improve the efficiency of runway utilization. It is known as the Aircraft Arrival/Departure Scheduling Problem (AADSP), or Aircraft Scheduling Problem (ASP) for simple (Beasley et al. [6], Balakrishnan and Chandran [7], Ikli et al. [8]).

ASP aims to take off/land as many aircraft as possible within a given period of time. ASP concerns a given minimum separation time (MST) table. Table 1 illustrates the MST between the successive aircraft concerning three typical aircraft types with arrival or departure status, that is, small arrival, large arrival, heavy arrival, small departure, large departure and heavy departure. The MST table is asymmetrical. If a small arrival aircraft lands after a heavy arrival, the required separation between them is 196s; however, when a heavy arrival aircraft lands after a small arrival, the separation is only 60 s. So a proper scheduling strategy may greatly shorten the makespan. Figure 1 presents two scheduling

strategies for the same three aircraft. The strategy of sequences 2 saves more than 65% of time than that of sequence 1.

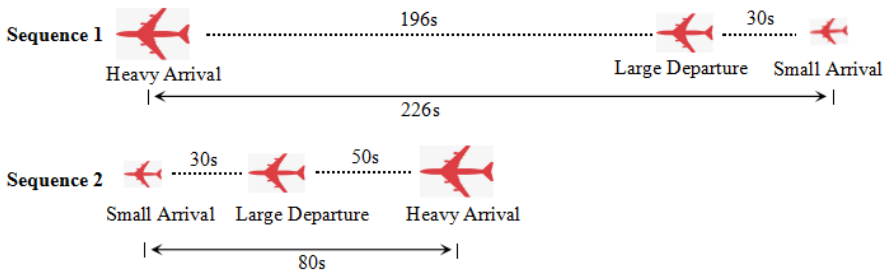

Figure 1. The runway occupation time reduces 65% for the same three aircraft after optimization.

The asymmetry nature leads ASP to be NP-hard with high computational complexity. There are at most $A_n^n = [n(n-1)(n-2)\cdots 1]$ possible sequencing strategies with n aircraft (Beasley et al. [6], Balakrishnan and Chandran [7]). The Branch and Bound (B&B), Ant Colony (AC) and Matrix Approximation (MA) algorithms are the most used solution methods.

Branch and Bound (B&B): Brinton [9] has introduced the first B&B approach for ASP in 1992, which is a foundation tool of the Center TRACON Automation System (CTAS) developed by NASA. Then Beasley et al. [6] proposes an widely cited Mixed Integer Programming (MIP) approach for ASP, and designs the corresponding B&B algorithm. Some additional constraints are added to reduce the zero-one space of the mixed integer formulation. Bennell et al. [4] present an extensive literature overview on B&B. Prakash et al. [10] proposes the B&B algorithm for single runway AADSP, where the data-splitting method is used to reduce the search space. It splits the original sequence into all possible pairs of sub-sequences, and then independently solves each sub-sequence, while ensuring global optimality. Then Prakash et al. [11] extend this method to the multi-runway case. The winter AADSP is studied by Pohl et al. [12]. The B&B algorithm is designed and three pruning tree rules, concerning time windows, separation identical aircraft and dominant snow removal patterns, are used.

Ant colony (AC): Ant colony is firstly adopted by Randall [13] for ASP in 2002. The objective function is to minimize the total penalty costs. Then Zhan et al. [14] combines the AC system with Receding Horizon Control (RHC) for real-time scheduling. By comparing with Hu and Chen [15], it validates that the AC is better than GA under the RHC strategy. Wu et al. [16] extends this method to the multi-runway case, and designs a two-stage RHC based AC algorithm. In the first stage, AC algorithm optimize the aircraft based on single runway problem; in the second stage, the aircraft are assigned to the multi-runway based on the first stage queue and the actual runway occupation. Bencheikh et al. [17] use a bi-level graph to model the multi-runway ASP and designs an AC algorithm. Ants start from a dummy initial node, select a runway, then insert an aircraft in this runway, based on the aircraft priority and the ant colony memory. Numerical results show that their AC finds optimal solution for 80% of the test instances. Bencheikh et al. [18] then study the dynamic ASP and propose a new algorithm named memetic which involves the local heuristic algorithm in AC. Ma et al. [19] uses the AC algorithm for the AADSP, a two-phase algorithm is proposed for the air traffic management. In the first phase, the takeoff aircraft are separated from the original sequence, and AC algorithm is used to schedule the landing sequence. In the second phase, the takeoff aircraft are inserted into the scheduled landing sequence while assure the landing first. A parameter M is set to restrict the time of each landing aircraft moving back from the scheduled time, which is not more than M. This method (named TPLP-M) is used as a comparison in the following numerical study.

Matrix Approximation (MA): The MA method is proposed by Ma et al. [20] and Faye [21]. Ma et al. [20] use a rank 2 matrix to approximate the original Minimum Separation Time (MST) matrix. Then the ASP becomes sequence-independent and thus easier to solve. This

method sacrifices the solution quality, but accelerates the computational speed. Faye [21] use the rank 2 matrix approximation method to study the multi-runway scheduling problem where the B&B algorithm is involved in the method. Lidier and Stolletz [22] admit the advantages of this method, but points out that this method will produce certain errors when approximating the matrix. Xu [23] studies the mechanism of errors generating, and use the AC algorithm to repair the accuracy loss. The methods have two steps: first, the aircraft MST matrix is approximated by rank 2 matrix, which similar as Ma et al. [20] and Faye [21]; second, AC algorithm is used to supplement the loss in the matrix approximating process. The numerical study validates that this new algorithm is better than the algorithm in Ma et al. [20] when minimize the makespan.

Other methods: Some other methods for runway and aircraft scheduling see in Anagnostakis and Clarke [24], Liu et al. [25], Atkins and Brinton [26], Bohme [27], Lee [28], Willemain et al. [29], Pinedo et al. [30], Khassiba et al. [31], Huo et al. [32], Yu et al. [33], Ghosh et al. [34]. And some hybrid algorithms are in Eslami et al. [35], Khajehzadeh et al. [36], Khajehzadeh et al. [37].

Table 1. MST (in sec.) between successive aircraft on the same runway for three aircraft types with two status .

			Following					
			Arrival			Departure		
			A1	A2	A3	D1	D2	D3
Leading	Arrival	A1	99	133	196	40	40	40
		A2	74	107	131	35	35	35
		A3	74	80	98	30	30	30
	Departure	D1	50	53	65	60	90	120
		D2	50	53	65	60	60	90
		D3	50	53	65	60	60	60

Note: A1-heavy arrival, A2-large arrival, A3-small arrival, D1-heavy departure, D2-large departure, D3-small departure.

In this paper, we go on investigating the MA method, and combine it with the AC algorithm to solve AADSP. AC is chosen here because each ant in AC directly returns a sequencing strategy which is a possible choice to increase the precision of the MA method. We firstly generate a rank 2 matrix (denoted by MST′) to approximate the actual minimum separation time (MST), where the optimal solution concerning MST′ is easy to find when the aircraft time window constraint is ignored and the triangle inequality assumption is not violated. Then we use AC to optimize the AADSP while considering the differences between MST and MST′ and the additional constraints. The numerical study validates that this new method has better performance than CPLEX and the two-stage algorithm in Ma et al. [19]. It is a promising method to improve the efficiency of the aircraft scheduling system.

In the remainder of this paper, we use **RMA-AC** to denote "the AC algorithm based on RMA method" for abbreviation.

This paper is organized as follows. The AADSP is defined in Section 2. In Section 3, the MST constraint is analyzed, and the RMA method is proposed. In Section 4, RMA-AC algorithm is originally designed to solve AADSP. The numerical study is conducted in Section 5, while some conclusions are summarized in Section 6.

2. Problem Definition

2.1. Basic Concepts

(1) First Come First Served (FCFS)

The FCFS strategy is widely used in the airport scheduling system. In the landing case, FCFS is implemented according to the estimated landing time of each aircraft. The estimated landing time is achieved by the aircrafts' planned arrival route, cruise speed, and the standard procedure descent profile. In the take-off case, the captain proposes to the ATC the time when the aircraft can leave the gate and drive to the runway, so as to determine the

take-off FCFS sequence. The advantage of FCFS owes to its simplified operation process to the controller and fairness to each aircraft, while the disadvantage owes to the low runway utilization.

(2) Time Windows Constraint

The time window constraint restricts that the aircraft must take-off/land within its earliest and latest possible take-off/landing time, otherwise insecurity may happen. We use E_i and L_i to indicate the earliest and latest possible takeoff/landing time of the aircraft i, then the actual takeoff/landing time should be restricted within the range of $[E_i, L_i]$. E_i is determined by the factors including the aircraft speed, runway availability, weather factors, scheduling strategy, and so on; while L_i is determined by the factors including aircraft speed, runway availability, limited aircraft fuel, maximum allowable delay time, scheduling strategy, and so on.

(3) Minimum Separation Time (MST)

The aircraft in the air generates wake vortex. If two aircraft get too close, the generated wake will cause dangerous rolling moment to the following aircraft. So the minimum separation must kept between them. The MST is determined by the aircraft weight, type, speed, navigation facilities, and other factors. The aircraft weight is the most important factor relating to the aircraft wake vortex. Generally speaking, heavy aircraft generates stronger wake, and can also bear stronger wake disturbance; while small aircraft are just the opposite. Therefore, the case of a small aircraft following a heavy aircraft (compared with a heavy aircraft following a small) requires greater separation (see in Table 1, and see also in Hancerliogullari et al. [38]).

It is worthy to note that, in AADSP, the triangle inequality for aircraft minimum separation may not satisfy, that is, $d_{ij} < d_{is} + d_{sj}, i, j, s \in \Lambda$, where Λ is the aircraft sequence set, and d_{ij} is the MST of aircraft i followed by aircraft j. For example, when there are three aircraft for scheduling, which are 1 arrival heavy, 2 small departure, 3 small arrival, then $d_{13} = 196 > d_{12} + d_{23} = 75 + 60$.

2.2. Problem Description

Suppose there are n aircraft, and they are labeled according to their order in the FCFS sequence. $\Lambda = \{1, 2, \cdots, n\}$ denotes the aircraft set. π is the final scheduled sequence which indicates the order of all the aircraft. π_i is the ith aircraft in sequence π, and its value indicates the order of aircraft π_i in the original FCFS sequence. For example, $\pi_5 = 6$ means that the fifth aircraft in the scheduled sequence π is the sixth aircraft in the FCFS sequence. $d_{\pi_i \pi_{i+1}}$ denotes the MST between two successive aircraft (π_i and π_{i+1}) in sequence π. E_{π_i} and L_{π_i} are the earliest and latest take-off/landing time of aircraft π_i, respectively. Then the final takeoff/landing time x_{π_i} for aircraft π_i can be achieved by

$$x_{\pi_i} = \begin{cases} E_{\pi_i} & \text{if } i = 1; \\ \max_{x_{\pi_i} \le L_i} \{E_{\pi_i},\ x_{\pi_{i-1}} + d_{\pi_{i-1}\pi_i}\}, & \text{if } i = 2; \\ \max_{x_{\pi_i} \le L_i} \{E_{\pi_i},\ x_{\pi_{i-1}} + d_{\pi_{i-1}\pi_i},\ x_{\pi_{i-2}} + d_{\pi_{i-2}\pi_i}\} & \text{other.} \end{cases} \tag{1}$$

The objective function is the makespan (W), which is the takeoff/landing time of the last aircraft in the scheduled sequence π (denoted by π_{end}), that is,

$$(\text{MIP}^1) \quad \min W_\pi = x_{\pi_{\text{end}}}, \qquad \forall \pi_i \in \Lambda \tag{2}$$

2.3. Mixed Integer Programming (MIP) Model

Suppose the arrival/departure aircraft set is $\Lambda = \{1, 2, \cdots, n\}$. Each aircraft $i \in \Lambda$ has a predefined time window $[E_i, L_i]$. d_{ij} is the MST of aircraft i followed by j, and x_i is the executed take-off/landing time of aircraft i. The MIP model is described in Equations (3)–(8).

The objective function (3) is to minimize the makespan of the final aircraft sequence, which is equivalent to Equation (2). Here, x_i is the takeoff/landing time of aircraft i,

and constraint (7) restrict $W \geq x_i$, $i = 1, 2, \cdots, n$. Thus W is the take-off/landing time of the last aircraft.

Constraint (4) ensures each aircraft i takoff/landing within its time window, which is between the earliest takeoff/landing time E_i and the latest takeoff/landing time L_i.

In constraint (5), δ_{ij} is a 0-1 variable. $\delta_{ij} = 1$ means that aircraft i should take-off/landing before aircraft j, while $\delta_{ij} = 0$ means that aircraft i should after j. δ_{ij} and δ_{ji} should not equal to 0 (or 1) at the same time. $\delta_{ij} + \delta_{ji} = 1$ ensures this requirement. U is the domain of all possible (i, j) in δ_{ij}, that is $U = \{(i, j) : i, j \in \Lambda, i \neq j\}$.

Constraint (6) ensures the MST constraint, where d_{ij} is the MST of aircraft j following i. There are two cases

(i) Aircraft i before j ($\delta_{ji} = 0$). Then $x_j \geq x_i + d_{ij}$ ensures the separation d_{ij} between aircraft i and j.

(ii) Aircraft i after j ($\delta_{ji} = 1$). Then $x_j \geq x_i + d_{ij} - M$, which is ineffective when M is large enough.

For the special case of triangle inequality not satisfying, constraint (6) ensures all aircraft in the runway satisfying MST constraint, and thus involves the special case.

Constraint (8) is the definition and restriction of all variables.

$$(\mathrm{MIP}^2) \quad \min W \tag{3}$$

$$\text{s.t.} \quad E_i \leq x_i \leq L_i; \quad \forall i \in \Lambda \tag{4}$$

$$\delta_{ij} + \delta_{ji} = 1; \quad \forall (i, j) \in U \tag{5}$$

$$x_j - x_i \geq d_{ij} - \delta_{ji} \cdot M; \quad \forall (i, j) \in U \tag{6}$$

$$W \geq x_i; \quad \forall i \in \Lambda \tag{7}$$

$$x_i \geq 0, W \geq 0, \delta_{ij} = 0, 1; \quad \forall (i, j) \in U \tag{8}$$

3. Rank 2 Matrix Approximation (RMA) Method

The RMA method is originally proposed by Ma et al. [20] and Faye [21]. Firstly, it generates a new rank 2 matrix (denoted by MST′), and the ASP concerning this new MST′ is linearly solvable. Secondly, MST′ limitlessly approximates the actual MST until they get close enough. Finally, the solution of ASP concerning the new MST′ is exact the (hypo-)optimal solution of that concerning the actual MST, that is, the traditional difficult ASP (concerning MST) is transferred to an easier ASP (concerning MST′). Following are the MST′ generating process and some useful properties.

3.1. Construction of Rank 2 Matrix Approximation Method

An important factor leading to the high complexity of ASP is the asymmetric nature of the MST matrix. The idea of the RMA method is to approximate the actual MST matrix by a rank 2 matrix, thus the ASP is simplified (Ma et al. [20], Faye [21], Xu [23])

The MST matrix is used to avoid the wake-vortex effect and ensure the safety of the following aircraft (Xu [23]). As we known, heavy aircraft can generate and bear stronger vortex, while small aircraft just the opposite. Then we can divide the MST matrix into two parts, where α_r denotes the ability of the leading aircraft of type r generating vortex and β_s denotes the ability of the following aircraft of type s bearing vortex. Thus the new MST matrix is achieved by $D'_{rs} = \alpha_r - \beta_s$ (see in Figure 2). Here, D' is the new rank 2 matrix, which is achieved by (LP^3). Θ is the aircraft type set, and suppose there are p type of aircraft.

The objective function (9) is to minimize the total differences between the original MST matrix D and the lately generated rank 2 matrix D'. ΔD measures the differences between D and D' (see in Equation (10)).

Constraint (11) is the lately generated rank 2 matrix D', where each element in it can be divided into two parts. The first part (denoted by α_r) models the ability of the leading aircraft of type r to generate vortex, and the second part (denoted by β_s) models the ability of the following aircraft of type s to bear vortex.

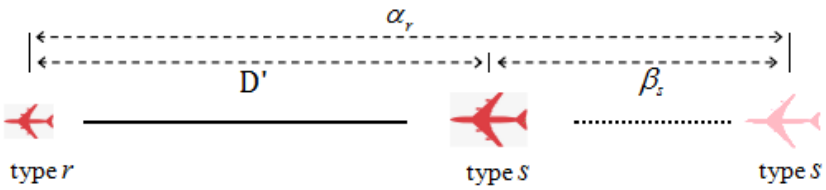

Figure 2. The division of MST matrix.

Constraint (12) restricts the main variables (ΔD_{rs}, α_r and β_s) non-negative. $\Delta D_{rs} \geq 0$ makes each element in D' not bigger than the corresponding element in D, which ensures the following theorem 1 and the condition (3) in theorem 3.

$$(\mathrm{LP}^3) \quad \min \Sigma_{r=1}^{p} \Sigma_{s=1}^{p} \Delta D_{rs} \tag{9}$$

$$\text{s.t.} \quad D_{rs} - D'_{rs} = \Delta D_{rs}; \quad \forall r, s \in \Theta \tag{10}$$

$$\alpha_r - \beta_s = D'_{rs}; \quad \forall r, s \in \Theta \tag{11}$$

$$\Delta D_{rs} \geq 0, \ \alpha_r \geq 0, \ \beta_s \geq 0; \quad \forall r, s \in \Theta \tag{12}$$

We can also use ΔMST to denote ΔD. Then there comes three main types of ASP, which are ASP-MST, ASP-MST′, and ASP-ΔMST. ASP-MST means that in the ASP the separation between the successive aircraft is determined by MST; ASP-MST′ means that determined by MST′; and ASP-ΔMST means that determined by ΔMST.

Following, D_{rs}, D'_{rs} and ΔD_{rs} denotes, respectively, the element in the matrices of MST, MST′ and ΔMST when an aircraft of type r is followed by another of type s.

3.2. Important Property of ASP-MST′

The rank 2 matrix MST′ generated above has two main properties to make optimization easier. Firstly, ASP-MST′ is the lower-bound of ASP-MST. Secondly, ASP-MST′ is sequence-independent and easier to solve (while ASP-MST is sequence-dependent). When the RAM method optimizes the original ASP-MST, it first finds the lower-bound via ASP-MST′, and then generates a solution approximating this lower bound to form a good final solution.

In this subsection, the time window constraint is assumed ignored and the triangle inequality for MST is assumed not violated, for simple. The case of violating the two assumptions is discussed in Section 4 when designing the AC algorithm for real scheduling. Theorems 1 and 2 explain the two properties of ASP-MST′ (see their proofs in Ma et al. [20]). Theorem 3 supports the approximating process (see its proof in Xu [23]). Here, W_π, W'_π and ΔW_π are the makespan of sequence π in ASP-MST, ASP-MST′ and ASP-ΔMST, respectively. Θ is the aircraft type set. When considering Table 1, we have $\Theta = \{1(A1), 2(A2), 3(A3), 4(D1), 5(D2), 6(D3)\}$.

Theorem 1. *The makespan of each sequence π in ASP-MST′ is the lower bound of π in ASP-MST, that is, $W'_\pi \leq W_\pi$.*

Theorem 2. *The optimal solution of minimizing makespan for ASP-MST′ only depends on the first and the last aircraft in the final sequence, if the time window constraint is assumed ignored, the triangle inequality for MST is assumed not violated, and α_i and β_i are given constant, $\forall i \in \Theta$.*

Theorem 2 simplifies ASP-MST′ when the time window constraint is ignored and triangle inequality is not violated. Theorem 3 explains the relationship of ASP-MST and ASP-MST′, and presents sufficient condition of the optimal solution of ASP-MST.

Theorem 3. *When the time window constraint is ignored and triangle inequality is not violated, the sequence Π is optimal (with minimized makespan) in ASP-MST model if the following three requirements are satisfied.*

(1) Sequence Π is the optimal sequence in ASP-MST′ model;

(2) *Sequence* Π *has 0 makespan in ASP-*Δ*MST model, i.e.,* $\Delta W_{\Pi} = 0$;
(3) *All elements in* Δ*MST are nonnegative, i.e., all elements in MST* $'$ *are not bigger than the corresponding elements in MST.*

Figure 3a illustrates the relationship between ASP-MST and ASP-MST′. Requirement (3) shows the ASP-MST′ curve always below ASP-MST curve. Requirements (2) means that ASP-MST curve is tangent to ASP-MST′ curve at the point representing sequence Π. Requirement (1) shows that the ASP-MST′ curve reaches the bottom at the point representing sequence Π. Thus ASP-MST is minimized at the point of Π.

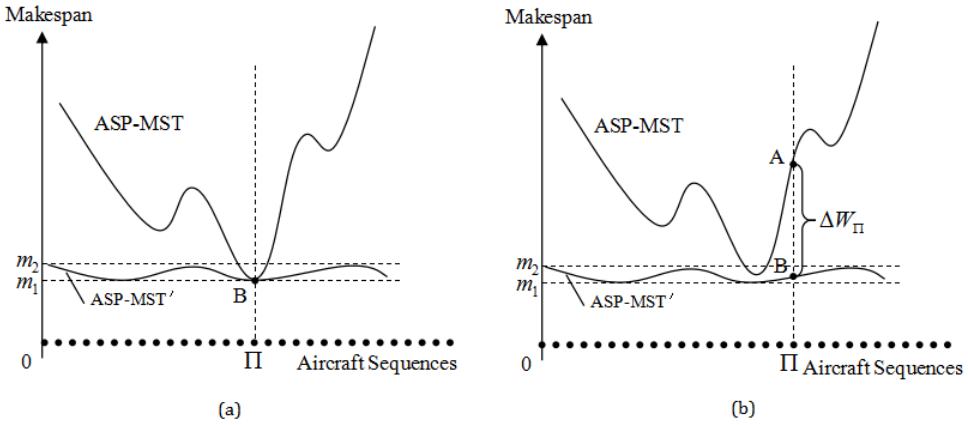

Figure 3. Makespan of sequences in ASP-MST and ASP-MST′. (**a**) the optimal solution appear in point *B*; (**b**) the optimization process is to minimize ΔW_{Π}.

If requirement (1) is not satisfied, but requirements (2) and (3) are satisfied, by a sequence Π, then Π is strongly hypo-optimal. Here, "strongly hypo-optimal" means that the sequence's optimality is only determined by its first and last aircraft. A sequence in ASP-MST′ is at least strongly hypo-optimal when the time window constraint is ignored and triangle inequality is not violated (Theorem 1).

Based on the above theorems, the optimization of ASP based on the RMA method contains three steps. First, for a given aircraft sequence, we use LP^3 to construct a rank 2 matrix D' satisfying the requirement (3) of theorem 3. Second, optimize ASP-ΔMST according to the requirement (2) in theorem 3 to minimize the makespan of ASP-ΔMST. Finally, the optimal or nearly optimal solution is obtained (that is, find the aircraft sequence Π so that ΔW_{Π} in Figure 3b is minimized). Following we use AC algorithm to optimize ASP-ΔMST and consider the case with time window constraint and triangle inequality violation.

4. Ant Colony Algorithm Based on RMA Method

As shown above, RMA method generates rank 2 matrices of MST′. Then we can optimize ASP-MST′ which is sequence-independent and easier to solve, but a significant loss of precision may happen between MST and MST′. To overcome it, the AC algorithm is used to supplement the precision via considering $\Delta\text{MST}(=\text{MST}-\text{MST}')$. The time window constraint and triangle inequality violation are also considered. Numerical results in Section 4 validate the efficiency of this new algorithm, named RMA-AC.

4.1. RMA-AC Algorithm Construction

Following introduces the RMA-AC algorithm for AADSP. AC is a famous meta-heuristic algorithm via modeling the ants behavior searching for food. It is originally developed for the Traveling Salesman Problem (TSP), which aims to find the shortest route for a traveler visiting each node once and only once in a graph, starting and finishing at the same node (Dorigo [39]).

The AADSP is similar as an open TSP (the tour does not return to the origin), where the runway is regarded as a traveler and aircraft as nodes (Randall [13], Bennell et al. [4]).

The time window constraint and triangle inequality violation are considered. The objective is to minimize makespan in Equation (2). The ants are expected to visit out all nodes as soon as possible, which is equivalent to the runway landing as many aircraft as possible. Following is the techniques setting for RMA-AC.

(1) Aircraft Sequences Classification

The aircraft with the same type are not expected to overtake each other, so they are put together to construct subsequences (F^r) according to their orders in the FCFS sequence (F). Each time only the first ant-unvisited aircraft in F^r is chosen to take-off/land so that the overtaking of aircraft with the same type is avoided. Since there are p aircraft types, we get p subsequences, which are $F^1, F^2, \cdots, F^p$. All aircraft in F^r have the same aircraft type of r, $r \in \Theta$. F^r_i denotes the ith aircraft in F^r.

For example, a FCFS sequence is of types $1, 5, 2, 1, 2, 2, 1, 5$, $\Theta = \{1, 2, 3, 4, 5, 6\}$. Then there are 6 subsequences. F^1 is constructed by the 1st, 4th and 7th aircraft in FCFS according to their order; F^2 is by the 3rd, 5th and 6th aircraft; F^3 and F^4 are empty sets, F^5 is constructed by the 2nd and 8th aircraft, and F^6 is also an empty set.

(2) Ants Initialization. The initial pheromone in each arc is set as $\tau_0(i, j) = 1, \forall i \neq j$. Here i and j are the nodes (aircraft), and arcs are imagined between nodes. $L^s = \{L^s_1, L^s_2, \cdots, L^s_p\}$ is an aircraft set for the ant s's next possible visiting. At the starting, $L^s_r = F^r_1, \forall r \in \Theta$, and each ant s randomly visits an aircraft in $\{F^1_1, F^2_1, \cdots, F^p_1\}$. After ant s visiting F^r_1, we set $L^s_r := F^r_2$; after visiting F^r_2, we then set $L^s_r := F^r_3$; and so on until all aircraft in F^r are visited.

(3) State Transition Rule. After ant s visiting aircraft i, it randomly selects the next allowable aircraft j to visit according to $\mathrm{Pb}(i, j)$.

$$\mathrm{Pb}(i,j) = \begin{cases} \frac{[\tau(i,j)]^{\alpha}[\eta(i,j)]^{\beta}}{\sum_{u \in L^s}[\tau(i,u)]^{\alpha}[\eta(i,u)]^{\beta}}, & \text{if } j \in L^s \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

where L^s is the allowable aircraft which follows aircraft i. $\tau(i, j)$ is the existed pheromone in arc (i, j), and $\eta(i, j)$ is the heuristic information factor. The parameters α and β determine the relative importance of τ versus η. The heuristic information $\eta(i, j)$ is set as

$$\eta(i,j) = \begin{cases} \frac{1}{\Delta d_{ij} + \max(0, E_j - x_i) + 1}, & i = \pi_1 \\ \frac{1}{\max\{\Delta d_{ij}, \Delta d_{i-1,j} - \Delta d_{i-1,i}\} + \max(0, E_j - x_i) + 1}, & i \geq \pi_2 \end{cases}, \tag{14}$$

because there are three main factors affecting the final makespan, which are the nonnegative number in ΔMST (denoted by Δd_{ij}), the time window constraint (denoted by $\max(0, E_j - x_i)$), and the triangle inequality violation (denoted by $\max\{\Delta d_{ij}, \Delta d_{i-1,j} - \Delta d_{i-1,i}\}$). The denominator is set as $+1$ to avoid the infeasible case of 0 denominator. In the actual computing, a simple setting may be more effective, which is

$$\text{OR} \quad \eta(i,j) = \frac{1}{\Delta d_{ij} + \max(0, E_j - x_i) + 1}. \tag{15}$$

It weakens the effect of triangle inequality violation in the state transition process. The final makespan computing should be added when ants complete their tours.

(4) Pheromone Updating Rules. The pheromone updating is carried out on each arc (i, j) as Equation (16) after all ants completing their tours.

$$\tau^{t+1}_{ij} = (1 - \gamma)\tau^t_{ij} + \Delta\tau_{ij} \tag{16}$$

where τ_{ij}^t is the original pheromone in arc (i,j), and τ_{ij}^{t+1} is the updated pheromone. $(1-\gamma)\tau_{ij}^t$ models the pheromone evaporation, where the evaporation ratio is $(1-\gamma)$. $\Delta\tau_{ij}$ denotes the lately released pheromone. $\Delta\tau_{ij} = \sum_s \Delta\tau_{ij}^s$, and $\Delta\tau_{ij}^s$ is defined, for an ant s, as

$$\Delta\tau_{ij}^s = \begin{cases} \frac{Q}{x_{\pi_{\text{end}}}^s}, & \text{if arc } (i,j) \text{ is visited by ant } s \\ 0, & \text{otherwise} \end{cases} \tag{17}$$

where $x_{\pi_{\text{end}}}^s$ is the tour makespan achieved by Equation (1) for ant s, and Q is a given constant.

4.2. Complete AC Algorithm

The complete RMA-AC algorithm is explained as follows, see also in Figure 4.

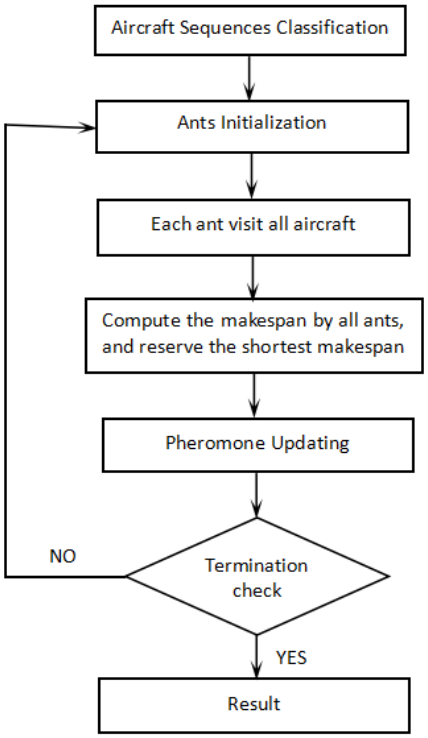

Figure 4. The complete RMA-AC algorithm.

Step 1. Initialization. There are m ants traveling between aircraft. The aircraft sequence is divided into p subsequences according to Section 4.1(1). $L^s = \{L_1^s, L_2^s, \cdots, L_p^s\}$ is the possible aircraft set for ant s's next visiting. At the starting, each ant s randomly selects an aircraft in L^s to visit. If $L_i^s = F_1^i$ is chosen, we set $L_i^s = F_2^i$.

Step 2. Each ant s randomly visits the next aircraft in L^s by Equations (13) and (15). After s visiting $L_i^s = F_j^i$, set $L_i^s := F_{j+1}^i$. If all aircraft in F^i have been visited ($j+1 > |F^i|$), delete L_i^s from L^s.

Step 3. If all aircraft in L^s have been visited by ant s ($L^s = \emptyset$), go to Step 4; otherwise ($L^s \neq \emptyset$), go to Step 2.

Step 4. Calculate out ant s's tour makespan $x_{\pi_{\text{end}}}^s$ by Equation (1), then find the shortest tour $\min_s x_{\pi_{\text{end}}}^s$. Compare it with the historical shortest tour, and reserve the shorter one.

Step 5. Update the pheromone τ_{ij} by Equations (16) and (17).

Step 6. Termination check. If the termination criteria of the limit time is met, then break the loop and return the final result. Otherwise, go to Step 1.

5. Numerical Study

In this section, two types of data are used to evaluate the algorithm. One is the fixed date given by the literature [19], and the other is the random data generated according to literature [38]. Here, 1, 2 and 3 represent *heavy*, *large* and *small* type of landing aircraft, respectively; 4, 5 and 6 represent *heavy*, *large* and *small* type of takeoff aircraft, respectively. The parameters of RMA-AC algorithm, as well as TPLP-M algorithm in [19], are set as $m = 150$, $\alpha = 1$, $\beta = 4$, $\gamma = 0.1$, $Q = 100$. The completion time of FCFS is calculated out by Equation (1), where π is replaced by FCFS. Matlab 7.11.0 and CPLEX Optimization Studio 12.5 are used for programming. The computation is carried out in the personal computer with Intel(R) Core(TM) i3-2310M CPU @ 2.10 GHz 2.10 GHz, RAM 4.00GB and Windows 7 OS.

5.1. Test of Fixed Aircraft Sequence

The data in the literature [19] is used for computation, which contains 40 aircraft in total for take-off and landing. Table 2 illustrates the final results by CPLEX, TPLP-M and RMA-AC algorithms. It is clear that all of the three algorithms find the optimal solution within 20 s. The reason of setting 20 s is because the air-traffic controller should make quick decision when scheduling the aircraft. A decision cost more than 20 s time of computing loses practical value.

Table 2. RMA-AC, TPLP-M and CPLEX to minimize makespan for the given data within 20 s.

Data			FCFS	CPLEX		TPLP-M		RMA-AC	
No.	Aircraft Type	Earliest Time (s)	Executed Time (s)	No.	Executed Time (s)	No.	Executed Time (s)	No.	Executed Time (s)
1	2	71	71	1	71	2	128	3	154
2	2	128	178	2	178	3	202	1	287
3	1	154	252	3	252	1	335	6	361
4	5	348	348	4	348	4	370	4	401
5	4	353	408	5	408	6	420	2	494
6	1	355	458	6	458	5	460	5	529
7	1	474	557	7	557	7	519	7	579
8	1	622	656	8	656	8	633	9	628
9	4	628	696	10	696	9	673	8	678
10	6	640	816	9	756	11	829	10	748
11	3	651	881	11	821	10	859	11	874
12	4	734	911	12	851	12	919	12	904
13	4	807	971	13	911	15	1024	13	964
14	4	1011	1031	14	1011	13	1059	15	1017
15	2	1013	1084	15	1064	16	1131	14	1052
16	2	1113	1191	16	1171	14	1166	16	1124
17	5	1160	1226	17	1206	20	1262	18	1161
18	4	1161	1286	20	1302	17	1292	21	1224
19	4	1175	1346	18	1332	21	1342	19	1264
20	3	1221	1411	19	1392	18	1382	20	1420
21	1	1224	1485	22	1445	23	1447	17	1450
22	2	1404	1618	21	1519	19	1487	22	1503
23	1	1430	1692	23	1618	22	1580	23	1577
24	4	1520	1732	24	1658	24	1615	24	1617
25	5	1526	1822	27	1717	26	1665	26	1676
26	1	1599	1872	25	1757	25	1705	25	1716
27	1	1699	1971	28	1816	27	1764	27	1775
28	1	1744	2070	30	1856	30	1814	30	1815
29	2	1807	2203	26	1915	28	1864	29	1908
30	6	1814	2238	31	1955	31	1904	31	1943
31	6	1874	2298	29	2048	29	1997	28	1993
32	5	1959	2358	32	2083	32	2032	32	2033
33	4	2082	2418	33	2143	33	2092	33	2093
34	3	2195	2483	34	2208	34	2224	34	2195
35	6	2241	2538	35	2263	35	2254	35	2241
36	1	2250	2588	37	2328	37	2322	37	2306
37	3	2283	2784	38	2360	38	2360	38	2360
38	5	2360	2814	36	2410	36	2410	36	2410
39	5	2379	2874	39	2450	39	2450	39	2450
40	4	2392	2934	40	2510	40	2510	40	2510
makespan(s)			2934		2510 *		2510 *		2510 *
CPU Time(s)					1.8		20		20

Note: 1-Heavy Arrival, 2-Large Arrival, 3-Small Arrival, 4-Heavy Departure, 5-Large Departure, 6-Small Departure, * represents the best solution.

In order to prove the effectiveness and stability of TPLP-M and RMA-AC algorithms, the data in Table 2 is tested for 20 times. The average values, worst solution and the

best solution are exhibited (see in Table 3). When the CPU computing time is 10 s, 20 s and 60 s, all the solutions obtained by RMA-AC algorithm have better performance than TPLP-M algorithm.

Table 3. Solution quality of RMA-AC, TPLP-M and CPLEX with the time limit of 20 s.

Algorithm	Worst Solution	Best Solution	Average Value	Average CPU Time	Runway Enhance
CPLEX	2510	2510	2510	2	14.5%
TPLP-M	2618	2510	2559	10	12.8%
RMA-AC	2533	2510	2525	10	13.9%
TPLP-M	2627	2510	2545	20	13.3%
RMA-AC	2531	2510	2518	20	14.2%
TPLP-M	2611	2510	2568	60	12.5%
RMA-AC	2526	2510	2517	60	14.2%

In order to test the convergence of the RMA-AC algorithm, Figure 5 records the average sequence makespan in each generations. It is clear that the results returned by the RMA-AC gradually converges with the increase of the ants generations.

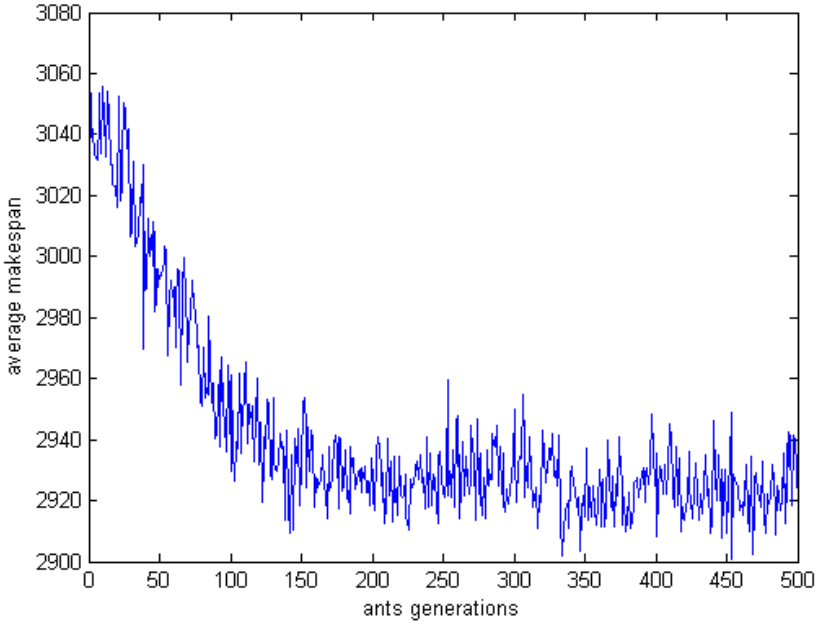

Figure 5. Relationship between ants generations and the average makespan in each generation (total computation time of 20 s).

5.2. Test of Random Aircraft Sequences

The random data is generated according to literature [38]. In the sequence, six types of aircraft are randomly generated, and the arrival and departure aircraft both accounts for 50%. The heavy aircraft accounts for 50%, the large aircraft accounts for 30%, and the small aircraft accounts for 20%. The earliest arrival/departure time E_i follows the discrete uniform distribution, which is a random number in the interval of $[0, 65n]$ (n denotes the number of aircraft). The latest arrival/departure time is the earliest arrival/departure time plus 60 min. The CPU computing time is restricted 20 s. The main reason is that the aircraft scheduling decision is a real-time decision-making problem, which need quickly calculated results. Decision with more than 20 s time of computing often loses practical value since the aircraft has changed its position for a long distance. Table 4 and Figure 6 show the results of 20 groups with the number of aircraft from 40 to 800. The data in each group were tested for 5 times and the average value is illustrated.

By comparing the results of three methods of CPLEX, TPLP-M and RMA-AC, we can see that the three methods are obviously better than the FCFS algorithm which is widely used in the current runway system. When the number of aircraft is less than or equal to 120, CPLEX gives the best results, and the runway efficiency is improved by about 4–11%. When the number of aircraft is greater than 120, RMA-AC gives the best solution, and the runway efficiency is improved by about 2–5%. By comparing the results of TPLP-M and RMA-AC, it is obvious that the latter has better performance than the former one.

Table 4. A comparison of the solution by the three methods for the random data.

	FCFS	CPLEX		TPLP-M		RMA-AC	
Aircraft Number	**Makespan (s)**	**Runway Enhance**	**Time (s)**	**Runway Enhance**	**Time (s)**	**Runway Enhance**	**Time (s)**
40	3188	10.87% *	2	10.36%	20	10.64%	20
80	5984	5.73% *	7	4.81%	20	5.43%	20
120	8832	4.51% *	10	3.65%	20	4.45%	20
160	11,846	4.87%	20	4.65%	20	5.11%	20
200	14,664	4.48%	20	3.78%	20	4.71%	20
240	17,498	2.61%	20	2.70%	20	3.58%	20
280	20,838	2.64%	20	4.84%	20	5.32%	20
320	23,380	3.16%	20	3.58%	20	4.29%	20
360	26,363	1.88%	20	3.28%	20	3.92%	20
400	29,111	1.44%	20	2.14%	20	3.06%	20
440	31,995	1.52%	20	2.37%	20	3.17%	20
480	34,658	0.81%	20	1.93%	20	2.69%	20
520	37,440	0.49%	20	1.67%	20	2.54%	20
560	40,257	0.48%	20	1.88%	20	2.64%	20
600	43,873	0.32%	20	3.03%	20	3.65%	20
640	46,595	0.19%	20	2.88%	20	3.72%	20
680	49,434	0.25%	20	2.39%	20	3.25%	20
720	52,378	0.26%	20	2.46%	20	3.18%	20
760	54,940	0.50%	20	2.02%	20	2.89%	20
800	58,038	0.39%	20	2.47%	20	3.40%	20

Note: * labels the optimal solution, underline labels the best solution among three methods.

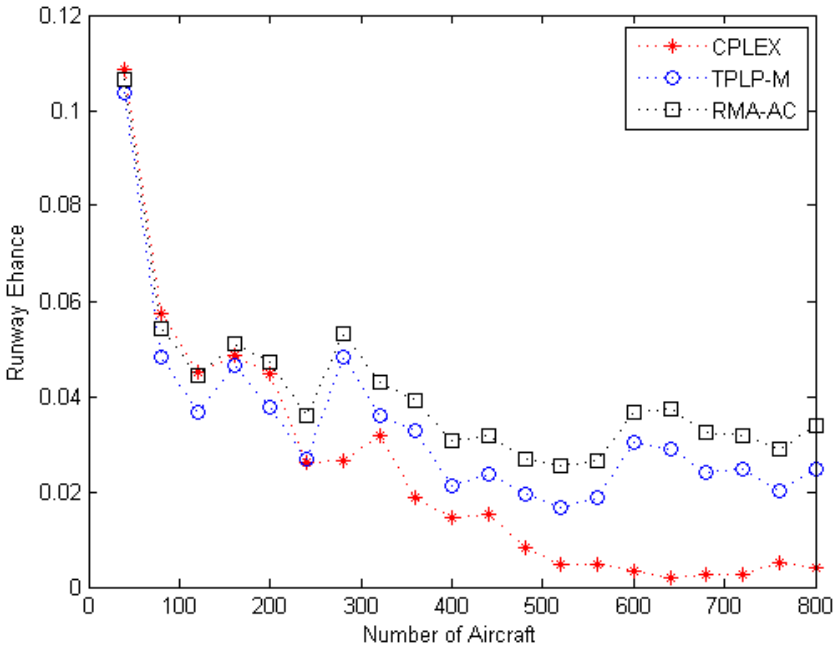

Figure 6. A comparison of the solution by the three methods for the random data.

6. Conclusions and Future Work

The Aircraft Scheduling Problem (ASP) is the key problem determining airport scheduling efficiency. This paper studies the single runway Aircraft Arrival/Departure

Scheduling Problem (AADSP). A Mixed Integer Programming (MIP) model is constructed, and three methods are used, which are the classical Two-Phase with Landing Priority (TPLP-M) algorithm, Ant Colony based on Rank 2 Matrix Approximation (RMA-AC) and the CPLEX optimizer. Numerical experiments show that the three methods can improve the efficiency of runway scheduling. When the number of aircraft is not more than 120, the CPLEX solution is the best; When the number of aircraft is more than 120, the RMA-AC solution is the best. These methods have a positive effect on optimizing the current airport scheduling system, improving the efficiency of runway scheduling and reducing aircraft delay. Numerical result also validates that RMA-AC has better performance than TPLP-M.

There are some research directions for the future study. Firstly, a more efficient algorithm can be designed to improve the efficiency of solution, such as combining the RMA methods with other heuristic algorithm for optimization to get higher efficiency. Secondly, considering the terminal area structure and runway structure, a practical model suitable for a specific airport can be established, such as multi-runway scheduling model. Thirdly, AADSP can be carried out under specific scenarios, such as in busy hours, under emergencies, in the epidemic prevention and control environment. In addition, extending the RMA method to solve other combinatorial optimization problems is also challenging and interesting.

Author Contributions: Conceptualization, B.X.; methodology, B.X.; software, B.X.; formal analysis, H.K.; resources, W.M.; writing original draft preparation, B.X., W.Y. and H.Z. All authors have read and agreed to the published version of the manuscript.

Funding: National Natural Science Foundation of China, grant number 71071113, 71371141; the National Social Science Fund of China, grant number 20BGL115

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available in the article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Boeing. Current Market Outlook. 2022. Available online: http://www.boeing.com/commercial/cmo/ (accessed on 20 May 2022).
2. Federal Aviation Administration. 2022. Available online: https://aspm.faa.gov/ (accessed on 20 May 2022).
3. Gudmundsson, S.V.; Cattaneo, M.; Redondi, R. Forecasting temporal world recovery in air transport markets in the presence of large economic shocks: The case of COVID-19. *J. Air Transp. Manag.* **2021**. *91*, 1–8. [CrossRef]
4. Bennell, J.A.; Mesgarpour, M.; Potts, C.N. Airport runway scheduling. *Q. J. Oper. Res.* **2011**, *9*, 115–138. [CrossRef]
5. Wuhan Network Information Office. Hubei Passenger and Cargo "Double Hub" Construction Booming and the Third Runway of Tianhe Airport Get the "Birth Permit". 2022. Available online: https://baijiahao.baidu.com/s?id=1729380164607604828&wfr=spider&for=pc (accessed on 20 May 2022). (In Chinese)
6. Beasley, J.E.; Krishnamoorthy, M.; Sharaiha, Y.M.; Abramson, D. Scheduling aircraft landings—The static case. *Transp. Sci.* **2000**, *34*, 180–197. [CrossRef]
7. Balakrishnan, H.; Chandran, B.G. Algorithms for scheduling runway operations under constrained position shifting. *Oper. Res.* **2010**, *58*, 1650–1665. [CrossRef]
8. Ikli, S.; Mancel, C.; Mongeau, M.; Olive, X.; Rachelson, E. The aircraft runway scheduling problem: A survey. *Comput. Oper. Res.* **2021**, *132*, 1–20.
9. Brinton, C.R. An implicit enumeration algorithm for arrival aircraft scheduling. In Proceedings of the IEEE/AIAA 11th Digital Avionics Systems Conference, Seattle, WA, USA, 5–8 October 1992.
10. Prakash, R.; Piplani, R.; Desai, J. An optimal data-splitting algorithm for aircraft scheduling on a single runway to maximize throughput. *Transp. Res. Part C* **2018**, *95*, 570–581. [CrossRef]
11. Prakash, R.; Piplani, R.; Desai, J. An optimal data-splitting algorithm for aircraft sequencing on two runways. *Transp. Res. Part C* **2021**, *132*, 1–15. [CrossRef]
12. Pohl, M.; Kolisch, R.; Schiffer, M. Runway scheduling during winter operations. *Omega* **2021**, *102*, 1–16. [CrossRef]
13. Randall, M.C. Scheduling aircraft landings using ant colony optimization. In Proceedings of the I-ASTED International Conference Artificial Intelligence and Soft Computing, Ban, QC, Canada, 17–19 July 2002.
14. Zhan, Z.H.; Zhang, J.; Li, Y.; Liu, O.; Kwok, S.K.; Ip, W.H.; Kaynak, O. An efficient ant colony system based on receding horizon control for the aircraft arrival sequencing and scheduling problem. *IEEE Trans. Intell. Transp. Syst.* **2010**, *11*, 399–412. [CrossRef]

15. Hu, X.B.; Chen, W.H. Receding horizon control for aircraft arrival sequencing and scheduling. *IEEE Trans. Intell. Transp. Syst.* **2005**, *6*, 189–197. [CrossRef]
16. Wu, L.J.; Zhan, Z.H.; Hu, X.M.; Guo, P.; Zhang, Y.; Zhang, J. Multi-runway Aircraft Arrival Scheduling: A Receding Horizon Control Based Ant Colony System Approach. In Proceedings of 2019 IEEE Congress on Evolutionary Computation, Wellington, New Zealand, 10–13 June 2019; pp. 538–545.
17. Bencheikh, G.; Boukachour, J.; Alaoui, A.E.H. Improved Ant Colony Algorithm to Solve the Aircraft Landing Problem. *Int. J. Comput. Theory Eng.* **2011**, *3*, 224–233. [CrossRef]
18. Bencheikh, G.; Boukachour, J.; Alaoui, A.E.H. A memetic algorithm to solve the dynamic multiple runway aircraft landing problem. *J. King Saud Univ. Comput. Inf. Sci.* **2016**, *28*, 98–109. [CrossRef]
19. Ma, W.; Yang, W.; Xu, B. Optimization algorithm for sequencing of single-runway mixed arrival-departure aircrafts. *J. Syst. Eng.* **2018**, *33*,125–135. (In Chinese)
20. Ma, W.; Xu, B.; Liu, M.; Huang, H. An efficient approximation algorithm for aircraft arrival sequencing and scheduling problem. *Math. Probl. Eng.* **2014**, *2014*, 236756. [CrossRef]
21. Faye, A. Solving the Aircraft Landing Problem with time discretization approach. *Eur. J. Oper. Res.* **2015**, *242*, 1028–1038. [CrossRef]
22. Lieder, A.; Stolletz, R. Scheduling aircraft take-offs and landings on interdependent and heterogeneous runways. *Transp. Res. Part E Logist. Transp. Rev.* **2016**, *88*, 167–188. [CrossRef]
23. Xu, B. An Efficient Ant Colony Algorithm Based on Wake-Vortex Modeling Method for Aircraft Scheduling Problem. *J. Comput. Appl. Math.* **2017**, *317*, 157–170. [CrossRef]
24. Anagnostakis, I.; Clarke, J.P. Runway operations planning: A two-stage heuristic algorithm. In Proceedings of the AIAA Aircraft, Technology, Integration and Operations Forum, Los Angeles, CA, USA, 1–3 October 2002.
25. Liu, M.; Liang, B.; Zhu, M.; Chu, C. Stochastic Runway Scheduling Problem With Partial Distribution Information of Random Parameters. *IEEE Access* **2020**, *8*, 68460–68473. [CrossRef]
26. Atkins, S.; Brinton, C. Concept description and development plan for the surface management system. *J. Air Traffic Control* **2002**, *44*, 1–8.
27. Bohme, D. Tactical departure management with the Eurocontrol/DLR DMAN. In Proceedings of the 6th USA/Europe ATM R&D Seminar, Baltimore, MD, USA, 27–30 June 2005.
28. Lee, H. Tradeoff Evaluation of Scheduling Algorithms for Terminal-Area Air Traffic Control. Master's Thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2008.
29. Willemain, T.R.; Fan, H.; Ma, H. *Statistical Analysis of Intervals between Projected Airport Arrivals*; DSES Technical Report 38-04-510; Rensselaer Polytechnic Institute: Troy, NY, USA, 2004
30. Pinedo, M.; Zacharias, C.; Zhu, N. Scheduling in the service industries: An overview. *J. Syst. Sci. Syst. Eng.* **2015**, *24*, 1–48. [CrossRef]
31. Khassiba, A.; Bastin, F.; Cafieri, S.; Gendron, B.; Mongeau, M. Two-stage stochastic mixed-integer programming with chance constraints for extended aircraft arrival management. *Transp. Sci.* **2020**, *54*, 897–919. [CrossRef]
32. Huo, Y.; Delahaye, D.; Sbihi, M. A probabilistic model based optimization for aircraft scheduling in terminal area under uncertainty. *Transp. Res. Part C* **2021**, *132*, 103374. [CrossRef]
33. Vincent, F.Y.; Qiu, M.; Pan, H.; Chung, T.P.; Gupta, J.N. An Improved Immunoglobulin-Based Artificial Immune System for the Aircraft Scheduling Problem With Alternate Aircrafts. *IEEE Access*, **2021**, *9*, 16532–16545.
34. Ghosh, S.; Laguna, S.; Lim, S.H.; Wynter, L.; Poonawala, H. A Deep Ensemble Method for Multi-Agent Reinforcement Learning: A Case Study on Air Traffic Control. In Proceedings of the Thirty-First International Conference on Automated Planning and Scheduling, Guangzhou, China, 7–12 June 2021; 31, pp. 468–476.
35. Eslami, M.; Shareef, H.; Mohamed, A.; Khajehzadeh, M. Damping Controller Design for Power System Oscillations Using Hybrid GA-SQP. *Int. Rev. Electr. Eng.* **2011**, *6*, 888–896.
36. Khajehzadeh, M.; Taha, M.R.; Eslami, M. Multi-objective optimisation of retaining walls using hybrid adaptive gravitational search algorithm. *Civ. Eng. Environ. Syst.* **2014**, *31*, 229–242. [CrossRef]
37. Eslami, M.; Neshat, M.; Khalid, S.A. A Novel Hybrid Sine Cosine Algorithm and Pattern Search for Optimal Coordination of Power System Damping Controllers. *Sustainability*, **2022**, *14*, 541. [CrossRef]
38. Hancerliogullari, G.; Rabadi, G.; Al-Salem, A.H. Greedy algorithms and metaheuristics for a multiple runway combined arrival-departure aircraft sequencing problem. *J. Air Transp. Manag.* **2013**, *32*, 39–48. [CrossRef]
39. Dorigo, M. Optimization, Learning and Natural Algorithms. Ph.D. Thesis, Dip. Elettronica, Politecnico di Milano, Italy, 1992.