Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 236756, 8 pages
http://dx.doi.org/10.1155/2014/236756

Hindawi

Research Article

An Efficient Approximation Algorithm for Aircraft Arrival Sequencing and Scheduling Problem

Weimin Ma,[1] Bo Xu,[1] Ming Liu,[1] and Hui Huang[2]

[1]*School of Economics and Management, Tongji University, Shanghai 200092, China*
[2]*Foshan Shuyuan Science and Technology Company Limited, Foshan, Guangdong 528200, China*

Correspondence should be addressed to Bo Xu; 1986xubo@gmail.com

Received 23 June 2014; Accepted 20 August 2014; Published 31 December 2014

Academic Editor: Chunlin Chen

The aircraft arrival sequencing and scheduling (ASS) problem is a salient problem in airports' runway scheduling system, which proves to be nondeterministic polynomial (NP) hard. This paper formulates the ASS in the form of a constrained permutation problem and designs a new approximation algorithm to solve it. Then the numerical study is conducted, which validates that this new algorithm has much better performance than ant colony (AC) algorithm and CPLEX, especially when the aircraft types are not too many. In the end, some conclusions are summarized.

1. Introduction

With the rapid development of airline industry, serious congestions and frequent delays have been hitting most major airports in the world, especially in the United States and Europe [1]. How to enhance the air traffic capacity and reduce the delay becomes a severe problem [2, 3].

In 1998, runway has been identified as the primary bottleneck in air traffic [4]; that is, even small enhancements to runway throughput will significantly reduce the delay. However, building more runways is often considered not a realistic option because of practical constraints and huge investment costs. Therefore, many researches and technologists resort to a promising approach, which is to more optimally schedule the aircraft arrival sequence so that the runway can land as many aircraft as possible within a period of time. The optimization process is formulated in this paper as the aircraft arrival sequencing and scheduling (ASS) problem (see in Section 2).

However, ASS is inherently hard to solve [5]; it is nondeterministic polynomial (NP) hard [6, 7]. To cope with it, two methods are often adopted, which are mixed integer programming (MIP) and ant colony (AC) algorithm [5, 8–10].

ASS can be expressed by MIP formulations. In 1992, Brinton [11] has introduced, as far as we know, the first MIP formulations and designed an implicit enumeration (IE) algorithm to optimize it. In 1993, another MIP is presented by Abela et al. [12] for single-runway ASS. A branch and bound (B&B) algorithm is developed to solve it. Back in 1999, the MIP is presented not only in single but also in multiple runway [8]. Beasley et al. [5] give an improved B&B algorithm by employing linear programming (LP) based tree search. Then Bennell et al. [13] provide an extensive literature overview for ASS.

Ant colony (AC) algorithm [9, 10, 14] is another effective method for ASS. It is originally proposed by Dorigo in 1992 [15–17]. In 2002, Randall [10] presents its first application in ASS, which shows great advantages. Then AC is used to generate initial solutions and to incorporate local search heuristic for single- and multiple-runway ASS [9, 18]. Back in 2010, AC is developed to tackle the real-time ASS based on the receding horizon control by Zhan et al. [14]. Experimental results validate that AC is robust, effective, and efficient for ASS.

In this paper, rather than using the above two methods, we develop a new approximation algorithm for ASS. The core idea is to find the lower bound solution of the ASS problem.

Table 1: MST (in sec.) between operations on the same runway [22].

		Following		
		1	2	3
Leading	1	82	69	60
	2	131	69	60
	3	196	157	96

1 = small aircraft, 2 = large aircraft, and 3 = heavy aircraft.

Then the algorithm presents solutions infinitely approaching this bound. We compare the performance of this new algorithm with AC and MIP (by CPLEX). Computational results verify that this new algorithm returns much better solution and costs less time than AC and MIP, especially when there are several aircraft types.

This paper is organized as follows. In Section 2, some constraints for ASS are introduced. The approximation algorithm is proposed in Section 3. In Section 4, ant colony (AC) and MIP are designed for ASS. In Section 5, numerical study is conducted to compare the performance of this new algorithm with AC and MIP (by CPLEX), while some conclusion is summarized in Section 6.

2. Basic Concepts

2.1. Aircraft Sequencing and Scheduling (ASS) Problem. ASS aims to make the most use of runway, that is, to minimize the makespan of the landing sequence so that to land as many aircraft as possible within a period of time. The objective function is

$$\min W_\pi = x_{\pi_{\text{end}}}, \tag{1}$$

where $x_{\pi_{\text{end}}}$ is the landing time of the last aircraft in sequence π. For the ith aircraft in sequence π, its landing time is achieved by

$$x_{\pi_i} = \begin{cases} \max\left\{e_{\pi_i}, \text{current time}\right\}, & \text{if } i = 1; \\ \max\limits_{x_{\pi_{i-1}} \le l_{\pi_{i-1}}} \left\{e_{\pi_i}, x_{\pi_{i-1}} + \text{MST}_{\text{TP}(\pi_{i-1}),\text{TP}(\pi_i)}\right\}, & \text{if } 1 < i \le n, \end{cases} \tag{2}$$

where e_k and l_k insure the time-window constraint and MST insures the minimum separation time constraint. These two constraints, as well as some other constraints, are described below. If $x_{\pi_{i-1}} \le l_{\pi_{i-1}}$ is not satisfied, x_{π_i} equals $+\infty$.

2.1.1. Minimum Separation Time (MST) Constraints. MST is a hard constraint to insure safety. When an aircraft flies in the air, it generates wake-vortex (WV). However, WV may result in the instability of the following aircraft (to shake or to lift) [19]. To avoid this, a MST is strictly kept between them.

Table 1 illustrates a typical MST table concerning three main types of aircraft. Generally, a smaller aircraft followed by a larger aircraft requires much shorter MST than the other way around. For example, a small aircraft has to wait for 196 s after the landing of a heavy aircraft. However, when a heavy aircraft lands after a small aircraft, the MST is only 60 s. One reason is that larger aircraft commonly generates and tolerates more turbulent air, while smaller aircraft generates and tolerates less.

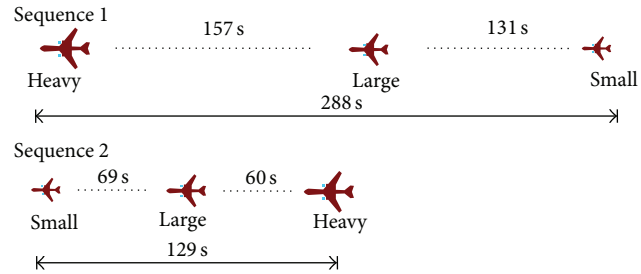

Figure 1: Two ordering methods result in different makespan for the same three aircraft.

The asymmetric nature of MST results in the feasibility and necessity of runway scheduling. Proper scheduling strategies can save a lot of landing time. For example in Figure 1, the makespan of sequence 1 equals 288 s; however, for sequence 2, the makespan is only 129 s, which saves more than 50% time.

2.1.2. Time-Window Constraints. Time-window is a hard constraint to insure that aircraft lands between its earliest and latest possible landing time, which is in the interval time set $[e_i, l_i]$. The earliest possible landing time (e_i) depends on the constraints such as maximum airspeed to speed up, runway availability, and possible manoeuvres, while the latest possible landing time (l_i) depends on the fuel limitation, maximum allowed delay, minimum airspeed, and so on [13]. Actually, it is not necessary that the time-window for an aircraft is only one that continues interval set [19]. Though we only discuss the single-interval case in this paper, our approach is also applicable to handle the situation that time-window constraints are disjoint intervals.

2.1.3. Precedence Constraints. Precedence constraints are pairwise requirements to insure whether one aircraft must land before another [19]. There are two reasons for these constraints. One is due to the airline company, which sometimes has strict restriction that one should land first for the reason of priority, banking operations, and so on. Another is due to the jet route, which does not allow two aircraft within the same jet route to overtake each other [20].

3. Approximation Algorithm for ASS

In this section, we design an approximation algorithm to solve ASS. The core idea is to find the lower bound solution of ASS. Then the approximation algorithm gives ASS solutions infinitely approaching this bound.

3.1. The Lower Bound Solution of ASS-MST. In the following we give the lower bound solution of ASS. This bound relates

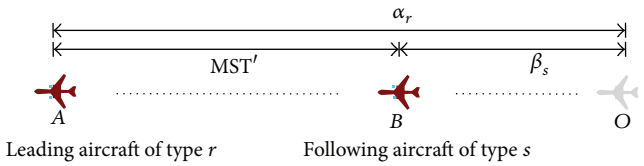

Figure 2: Decomposition of minimum separation time (MST′).

to a new generated MST (denoted by MST′), which is an approximation of the actual MST. How to determine MST′ is shown in LP[1] (formulas (3)–(6)).

The objective function (formula (3)) is to minimize the differences between MST and MST′ since ΔMST measures their differences (formula (4)).

Formula (5) calculates out MST′. We decompose each element in MST′ into two parts (see in Figure 2). One is the ability of the leading aircraft to generate the WV (denoted by α_r), and the other is the ability of the following aircraft to bear the disturbance (denoted by β_s).

In formula (6), ΔMST_{rs}, α_r, and β_s are restricted nonnegative. $\Delta\text{MST}_{rs} \geq 0$ insures that the elements in MST′ are not bigger than the corresponding elements in MST, which insures Theorem 1.

It is worthy to note that we do not restrict $\text{MST}'_{rs} \geq 0$, because it does not affect the result in Theorems 1, 2, and 4 from the perspective of mathematical calculation. In addition, it allows more elements in ΔMST_{rs} to be 0:

$$\left(\text{LP}^1\right): \ \min \sum_{r=1}^{m}\sum_{s=1}^{n}\Delta\text{MST}_{rs}, \tag{3}$$

subject to

$$\text{MST}_{rs} - \text{MST}'_{rs} = \Delta\text{MST}_{rs}; \quad \forall r, s \in \Lambda, \tag{4}$$

$$\alpha_r - \beta_s = \text{MST}'_{rs}; \quad \forall r, s \in \Lambda, \tag{5}$$

$$\Delta\text{MST}_{rs} \geq 0, \quad \alpha_r \geq 0, \quad \beta_s \geq 0; \quad \forall (r, s) \in \Lambda. \tag{6}$$

Theorem 1. *The makespan of each sequence π in ASS-MST′ is the lower bound of π in ASS-MST.*

Proof. Referring to formulas (4) and (6), we have $\text{MST}_{rs} - \text{MST}'_{rs} \geq 0$. So for any sequence π, $\text{MST}'_{\text{TP}(\pi_{i-1}),\text{TP}(\pi_i)} \leq \text{MST}_{\text{TP}(\pi_{i-1}),\text{TP}(\pi_i)}$. It is easy to confirm that $W'_\pi \leq W_\pi$ (see formulas (1) and (2)). □

Actually, this lower bound is very close to the optimal solution when the time-window constraint is not in consideration since its optimality only depends on the first and last aircraft in the sequence (Theorem 2).

Theorem 2. *The optimal solution of minimizing makespan for ASS-MST′ only depends on the first and the last aircraft in the final sequence, if the time-window constraint is not taken into consideration and α_r and β_r are given constant, $\forall r \in \Lambda$.*

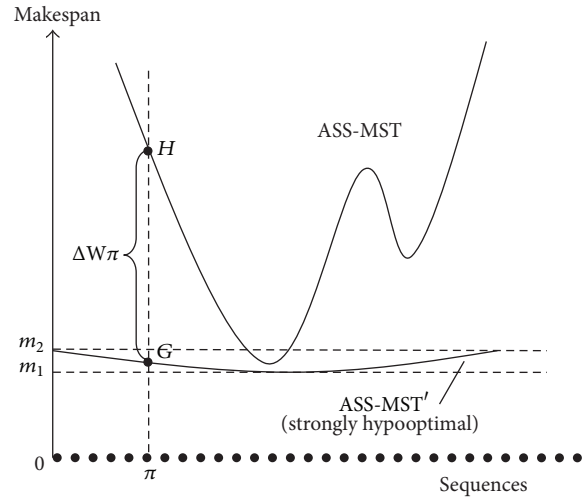

Figure 3: Makespan of sequences in ASS-MST and ASS-MST′.

Proof. For the landing sequence (π) with n aircraft, the MST′ between aircraft π_i and π_{i+1} is $(\alpha_{\text{TP}(\pi_i)} - \beta_{\text{TP}(\pi_{i+1})})$. So we have

$$W'_\pi = \sum_{i=1}^{n-1}\left(\alpha_{\text{TP}(\pi_i)} - \beta_{\text{TP}(\pi_{i+1})}\right), \tag{7}$$

which is equivalent to

$$W'_\pi = \sum_{i=1}^{n}\left(\alpha_{\text{TP}(\pi_i)} - \beta_{\text{TP}(\pi_i)}\right) + \left(\beta_{\text{TP}(\pi_1)} - \alpha_{\text{TP}(\pi_n)}\right). \tag{8}$$

In formula (8), $\sum_{i=1}^{n}(\alpha_{\text{TP}(\pi_i)} - \beta_{\text{TP}(\pi_i)})$ is a constant number since the type of each aircraft in π is known. So the makespan (W'_π) is determined by $(\beta_{\text{TP}(\pi_1)} - \alpha_{\text{TP}(\pi_n)})$, which only concerns the first and last aircraft in π. □

Definition 3. A sequence is called *strongly hypooptimal* if and only if the change of the first and the last aircraft leads to its optimality.

So any sequence in ASS-MST′ is at least strongly hypooptimal (SHO). Figure 3 lists the relationship between ASS-MST curve and ASS-MST′ curve when the time-window constraint is not in consideration. $(m_1 - m_2)$ is very small. We wish to minimize ΔW_π which measures how a solution is close to SHO solution (the lower bound).

3.2. Approximation Algorithm Approaches the Lower Bound. The approximation algorithm gives ASS solutions infinitely approaching the lower bound; that is, it attempts to find a sequence π with minimized ΔW_π (see in Figure 3) since ΔW_π measures how a solution is close to a SHO solution (the lower bound) when the time-window constraint is not in consideration (ΔW_π is also a useful measure when considering time-window). The numerical result in Section 5 validates the statement and the efficiency of algorithm.

3.2.1. Contain Time-Window with Aircraft Not Too Many. In ASS, each aircraft is associated with an estimated landing time

(ELT), which is used to estimate the time when the aircraft lands. The ELT relates to the earliest and the latest landing time of the time-window. Generally, the earliest landing time (e_i) is one minute less than the ELT, because more than one-minute forward move is often not economical [20]. The latest landing time (l_i) can be 60 minutes after the ELT if aircraft have enough fuel and without any emergency accident.

Based on the above discussion, the earliest landing time e_i is the main constraint in time-window, since the latest landing time is often inactive if the number of aircraft is not too many. So we firstly develop an algorithm for ASS without too many aircraft (often less than 30). And then we extend it to solve the case with many aircraft (often more than 30).

The core idea is to infinitely approach the lower bound, that is, minimizing additional makespan when comparing ASS-MST and ASS-MST′. The additional makespan consists of ΔMST and the earliest landing time (e_i). We give the approximation of the additional makespan.

For the FSFC sequence F, all aircraft with the same type are put together according to their order in F to construct subsequences. There are p subsequences, which are $F^1, F^2, \ldots, F^p$, where $\mathrm{TP}(F_i^r) = r$, $r \in \Lambda$, and F_i^r is the ith aircraft in F^r.

For example, a FCFS sequence is of types $1, 2, 3, 2, 1, 2, 3, 3$, $\Lambda = \{1, 2, 3\}$. Then there are 3 subsequences. F^1 is constructed by the 1st and 5th aircraft in FCFS according to their order, F^2 is by the 2nd, 4th, and 6th aircraft, and F^3 is by the 3rd, 7th, and 8th aircraft. The complete algorithm consists of the following 3 steps to determine the final landing sequence Π.

Step 1. Determine the first aircraft of Π. Find the smallest Δw^r in formula (9) and transfer the aircraft F_1^r to the first position in Π; delete F_1^r from subsequence F^r. Then set $i := 2$:

$$\Delta w^r = \begin{cases} e_{F_1^r}, & r = r^* \\ e_{F_1^r} + \Delta\mathrm{MST}_{\mathrm{TP}(r),\mathrm{TP}(r^*)}, & \text{otherwise,} \end{cases} \tag{9}$$

where $r^* = \min\{s : s \in \Lambda, F^s \neq \varnothing\}$.

Step 2. Determine the ith aircraft of Π. Find the aircraft F_1^r with the smallest Δw^r in formula (10). Transfer it to the ith position in Π and delete it from subsequence F^r. Then set $i := i + 1$:

$$\Delta w^r = \begin{cases} \Delta\mathrm{MST}_{\mathrm{TP}(\Pi_{i-1}),r} + g(i,r), & r = r^* \\ \Delta\mathrm{MST}_{\mathrm{TP}(\Pi_{i-1}),r} + g(i,r) + \Delta\mathrm{MST}_{r,r^*}, & \text{otherwise,} \end{cases} \tag{10}$$

where $g(i,r) = \max\{0, e_{F_1^r} - (x_{\Pi_{i-1}} + \mathrm{MST}_{\mathrm{TP}(\Pi_{i-1}),r})\}$ and $r^* = \min\{s : s \in \Lambda, F^s \neq \varnothing\}$.

Step 3. Terminal criteria: if all aircraft in subsequences (F^r, $r \in \Lambda$) are transferred into Π, return sequence Π; otherwise, go to Step 2.

Theorem 4. *The complexity of the approximation algorithm to generate the final sequence with n aircraft and p aircraft types is* $O(pn)$.

Proof. Step 2 costs $O(p)$ to find the smallest Δw^r. Since there are n aircraft, Step 2 repeats for n times. So the complexity is $O(pn)$. □

3.2.2. Contain Time-Window with Many Aircraft. If the number of aircraft is too large (often more than 30), the "inactive" assumption of the latest landing time is often broken, since some aircraft may be assigned to land 1 hour later. To avoid this, we firstly reschedule the first 30 aircraft (from the 1st to the 30th) in FCFS by the above algorithm and then reschedule the next 30 aircraft (from the 31st to the 60th) and then the next 30 aircraft (from the 61st to the 90th) and go on until all aircraft have been considered. In the end, we connect all the rescheduled sequences one by one to construct the ultimate sequence. Here the length of 30 aircraft is based on the numerical result in Section 5. Of course we can set other lengths for subrescheduling but without such a good solution. These whole processes do not exceed the complexity of Theorem 4.

3.2.3. Contain Precedence. To consider the precedence, we may freeze the subsequence. For example, aircraft i in subsequence F^s should land before aircraft j in F^r; we can freeze subsequence F^r when j is in the first position of F^r. The only sufficient condition to unfreeze F^r is that aircraft i has been transferred into the final sequence Π.

4. Ant Colony and MIP for ASS

4.1. Ant Colony (AC) for ASS. Ant colony (AC) algorithm is an intensively studied method [13]. Many papers adopt it for single-runway scheduling [9, 10, 14]. Here AC is used to find a solution for ASS, which is a valid comparison with the approximation algorithm. The following are some important settings of AC in the computation.

4.1.1. State Transition Check. All aircraft are labeled according to their positions in the FCFS. After finishing visiting an aircraft i, all ants choose the next allowable aircraft j to visit. To make sure most routes returned by ants satisfy the time-window constraints, we restrict the allowable aircraft to the integer set $\{\max\{1, i-k\}, \ldots, \min\{n, i+k\}\}$ when an ant does its ith visit. This principle is also called k-CPS [19].

Under this principle, an aircraft i should be visited before an ant does its $(i+k+1)$th visit. So if aircraft i has not been visited when an ant does its $(i+k)$th visit, it is forced to be visited by this ant. In the numerical study here, setting the integer number $k = 5$ is good to satisfy time-window.

4.1.2. State Transition Rule. After finishing visiting an aircraft i, all ants choose the next allowable aircraft j according to the probability of Pb(i, j):

$$\mathrm{Pb}(i,j) = \begin{cases} \dfrac{[\tau(i,j)]^a [\eta(i,j)]^b}{\sum_{u\in \mathrm{allowed}_i} [\tau(i,u)]^a [\eta(i,u)]^b}, & \text{if } j \in \mathrm{allowed}_i \\ 0, & \text{otherwise.} \end{cases} \quad (11)$$

4.1.3. Pheromone Updating Rule. The pheromone updating operation is carried out on each arc in the completed tour as in formulas (12) and (13) for each ant:

$$\tau_{ij}(t+1) = (1-\gamma)\tau_{ij}(t) + \Delta\tau_{ij}, \quad (12)$$

where $(1-\gamma)\tau_{ij}(t)$ models the evaporation of the pheromone and $\Delta\tau_{ij}$ is the lately released pheromone by ants. For an ant u, $\Delta\tau_{ij}^u$ is defined as

$$\Delta\tau_{ij}^u = \begin{cases} \dfrac{Q}{W^u}, & \text{if arc } (i,j) \text{ is visited by ant } u \\ 0, & \text{otherwise,} \end{cases} \quad (13)$$

where Q is a given constant and W^u is makespan of tour route by ant u.

4.2. Mixed Integer Programming (MIP) for ASS. In 2000, Beasley et al. [5] proposed a MIP for the single-runway scheduling problem. Here we extend it to ASS as a solution comparison of the approximation algorithm. The following are some important settings.

The objective function (formula (14)) is to minimize the makespan. $W \geq x_i$ (formula (18)) insures W is the landing time of the last aircraft.

In formula (15), δ_{ij} is a 0-1 variable. $\delta_{ij} = 1$ (aircraft i lands before j) and $\delta_{ij} = 0$ (i after j). δ_{ij} and δ_{ji} must not equal 0 (or 1) at the same time. $\delta_{ij} + \delta_{ji} = 1$ ensures this requirement.

Formula (16) is the separation constraint. There are two cases.

(a) $\delta_{ji} = 0$. Then $x_j \geq x_i + \mathrm{MST}_{\mathrm{TP}(i),\mathrm{TP}(j)}$, which ensures a time separation.

(b) $\delta_{ji} = 1$. Then $x_j \geq x_i + \mathrm{MST}_{\mathrm{TP}(i),\mathrm{TP}(j)} - M$, which is effectively inactive if M is large enough.

Formula (17) insures the time-window constraint:

$$(\mathrm{MIP}^1): \min \quad W \quad (14)$$

$$\text{s.t.} \quad \delta_{ij} + \delta_{ji} = 1; \quad \forall (i,j) \in U \quad (15)$$

$$x_j - x_i \geq \mathrm{MST}_{\mathrm{TP}(i),\mathrm{TP}(j)} - \delta_{ji} \cdot M; \quad \forall (i,j) \in U \quad (16)$$

$$e_i \leq x_i \leq l_i; \quad i = 1, 2, \ldots, n \quad (17)$$

$$W \geq x_i; \quad \forall i = 1, 2, \ldots, n \quad (18)$$

$$x_i \geq 0, \quad W \geq 0, \quad \delta_{ij} = 0, 1; \quad \forall (i,j) \in U. \quad (19)$$

Table 2: MST (in sec.) between operations on the same runway [6].

		Following			
		1	2	3	4
Leading	1	90	80	70	72
	2	110	80	70	72
	3	130	100	70	72
	3	228	200	181	96

1 = Mc Donnel Douglas DC9, 2 = Boeing 727, 3 = Boeing 707, and 4 = Boeing 747.

We use CPLEX Optimization Studio 12.5 to solve MIP[1], and the traditional Branch and Bound (B&B) search function in CPLEX is used. From the point of view of the LP relaxation, M (in formula (16)) is wished to be as small as possible. Because large M leads $(\mathrm{MST}_{\mathrm{TP}(i),\mathrm{TP}(j)} - \delta_{ji} \cdot M)$ to be much smaller than $\mathrm{MST}_{\mathrm{TP}(i),\mathrm{TP}(j)}$ even if δ_{ji} is very small, the small M is achieved by replacing it with $(l_i + \mathrm{MST}_{\mathrm{TP}(i),\mathrm{TP}(j)} - e_j)$ [5].

5. Numerical Study of Three Methods

Here is the numerical result of the approximation algorithm for ASS. The AC and CPLEX are used as a valid comparison of this new algorithm.

5.1. Randomly Generate the MST. In ASS, the concerning MST is important. Table 1 gives a typical MST table by a classical aircraft types classification. However, there are some other MST tables by other classifying principles. For example, Table 2 illustrates another MST table with more than 3 aircraft types. We believe that our algorithm applies almost all possible MST matrixes. So in the numerical study we randomly generate MST matrixes which have two common properties.

For MST matrixes with p aircraft types, their general properties are summarized as follows.

(1) Larger aircraft landing before the same aircraft needs more (at least equal) separation; that is, $\mathrm{MST}_{r,s} \geq \mathrm{MST}_{r-1,s}$, $r = 2, 3, \ldots, p$; $s = 1, 2, \ldots, p$.

(2) Larger aircraft landing after the same aircraft needs less (at least equal) separation; that is, $\mathrm{MST}_{r,s} \geq \mathrm{MST}_{r,s+1}$, $r = 1, 2, \ldots, p$; $s = 1, 2, \ldots, (p-1)$.

In numerical study, MST can be generated by

$$\mathrm{MST}_{rs} = \begin{cases} k_{rs}\mathrm{MST}_{r,s+1}, & \text{if } r = 1; \\ & s = 1, 2, \ldots, p-1 \\ k_{rs}\mathrm{MST}_{r-1,s}, & \text{if } r = 2, 3, \cdots, p; \\ & s = p \\ k_{rs} \cdot \max\{\mathrm{MST}_{r,s+1}, \mathrm{MST}_{r-1,s}\}, & \text{otherwise,} \end{cases} \quad (20)$$

where MST_{1p} is a given number and $k_{rs}(\geq 1)$ is a random number. In the following numerical study, if $p \leq 4$, k_{rs} is in the interval of $[1, 2]$ satisfying uniform distribution; if $p > 4$,

Table 3: WVM, AC, and MIP to minimize makespan in ASS when $p = 3, 9$.

Aircraft types	Number of aircraft	AC		CPLEX		Approx. algo.	
		Time (sec.)	Runway enhancement	Time (sec.)	Runway enhancement	Time (sec.)	Runway enhancement
$p = 3$	20	4.9	1.34%	0.2	4.07%	0.003	**4.60%**
	30	7.7	3.04%	0.2	4.77%	0.002	**6.21%**
	40	10.5	1.63%	0.4	2.58%	0.003	**4.39%**
	60	16.4	2.93%	0.7	2.70%	0.004	**5.09%**
	80	22.0	1.87%	0.9	1.51%	0.006	**4.18%**
	100	27.8	2.94%	1.2	1.80%	0.007	**3.39%**
	120	33.6	3.68%	1.9	1.59%	0.008	**3.81%**
	140	39.1	2.47%	2.2	1.40%	0.010	**3.13%**
	160	45.2	2.38%	2.1	0.87%	0.011	**4.55%**
	180	51.1	2.05%	2.9	1.16%	0.012	**3.69%**
	200	57.5	2.38%	3.7	1.23%	0.014	**2.67%**
$p = 9$	20	4.9	1.69%	0.2	**3.82%**	0.004	2.81%
	30	7.7	1.32%	0.3	2.35%	0.003	**2.64%**
	40	10.5	2.06%	0.4	1.98%	0.004	**4.15%**
	60	16.2	1.07%	0.6	0.98%	0.006	**2.41%**
	80	22.0	1.17%	1.0	0.74%	0.007	**2.63%**
	100	27.8	**1.28%**	1.3	0.82%	0.009	**1.28%**
	120	33.7	**1.21%**	1.9	0.48%	0.011	1.08%
	140	39.4	**1.36%**	2.6	0.84%	0.012	1.04%
	160	45.5	**1.79%**	2.9	0.55%	0.015	1.00%
	180	51.9	**1.55%**	3.1	0.90%	0.020	0.80%
	200	57.8	0.99%	3.5	0.68%	0.018	**1.09%**

Bold characters mark the maximum runway enhancement by these methods.

k_{rs} is in $[1, 1 + 1/(p-3)]$ satisfying uniform distribution. This setting insures MST_{p1} is not too much bigger than MST_{1p}. It is in accord with the actual MST; for example, Tables 1 and 2 are basically satisfied.

It is worthy to note that even if a MST does not strictly satisfy the above settings, it can also be well coped with by the approximation algorithm.

5.2. Numerical Result. We test the cases of $p = 3$ and $p = 9$. In each case, ten groups (aircraft number $n = 20, 40, 60, \ldots, 200$) of aircraft sequences are randomly generated. Each group contains 20 random sequences; the number of all types of aircraft is the same in each sequence. The MST is randomly generated for each sequence (see detailed setting and its parameters explanation in formula (20)).

The estimated landing time (ELT) of aircraft is simulated by a Poisson arrival process, which has been validated by Willemain et al. [21]. The expectation of interval time between neighbor aircraft in ELT is set to be $(4/5)E(\mathrm{MST})$, where $E(\mathrm{MST})$ is the mean value of all elements in MST. It models how crowded an airport is. The earliest landing time is set to be one minute less than the ELT, because more than a minute forward move is often not economical [20]. The latest possible landing time is set 60 minutes after the earliest landing time.

For approximation algorithm, long aircraft sequences (with more than 30 aircraft) are scheduled each time for 30 aircraft (see the explanation in Section 3.2.2). For AC algorithm, the termination criteria are the maximal generations of 100. The parameters in formulas (11)–(13) are set as $a = 1$, $b = 5$, $\gamma = 0.1$, and $Q = 100$. The allowable aircraft are restricted in set $\{\max\{1, i-5\}, \ldots, \min\{n, i+5\}\}$ when an ant does its ith visit. For MIP, the traditional B&B search in CPLEX is used and the maximal tree nodes of B&B search are set to be 4000. The numerical result is shown in Table 3 with average runtime and average runway enhancement of the tested 20 sequences in each group. For each sequence, the "runway enhancement" is achieved by $(W_{\mathrm{FCFS}} - W_{\Pi})/W_{\mathrm{FCFS}} \times 100\%$. It is comparison of the makespan between the final sequence Π and the FSFC sequence since FCFS is a widely used method in current runway scheduling system.

When there are fewer aircraft types (3 types), the approximation algorithm enhances the runway throughput more than AC and CPLEX and costs less time. It validates the performance of the algorithm. Actually classifying the aircraft into 3 types is often enough. If there are too many aircraft types, we can classify the aircraft into several big categories and reconstruct the MST table concerning the big categories other than the aircraft types—each big category contains aircraft types with very close properties. So the ASS in this condition can also be well coped with by the approximation algorithm.

When there are 9 aircraft types, the approximation algorithm generally has higher runway enhancement when

the number of aircraft is not too many (<100 aircraft). For too many aircraft (100–200 aircraft), AC enhances the runway throughput more. However, the approximation algorithm costs less time than the other two methods.

By comparing AC and MIP, we find that MIP (by CPLEX) has better performance when both the number of aircraft types and aircraft are small. For a large number of aircraft (or many aircraft types), it is better to use AC for scheduling other than CPLEX. It is also worthy to note that the approximation algorithm has better performance than CPLEX in almost all the enumerated cases in Table 3.

6. Conclusions

To cope with aircraft sequencing and scheduling (ASS) problem, we design an approximation algorithm, which is infinitely approaching the lower bound of the optimal solution. Numerical result validates that this algorithm, with less runtime, can get a smaller makespan than AC and CPLEX, especially when the number of aircraft types is not too many. This is a big increase of the runway.

Nomenclature and Notation

Nomenclature

AC: Ant colony
ASS: Aircraft sequencing and scheduling
ASS-MST: ASS problem concerning MST; that is, in ASS the separation between aircraft is determined by MST matrix
ASS-MST$'$: In ASS the separation is determined by MST$'$ matrix
FCFS: First come first served
LP: Linear programming
MIP: Mixed integer programming
MST: Minimum separation time (matrix). MST_{ij} denotes the MST of an aircraft of type i followed by another aircraft of type j. $\mathrm{MST}_{rs} \leqslant \mathrm{MST}_{rt} + \mathrm{MST}_{ts}, \forall r, s, t \in \Lambda$
MST$'$: A newly generated MST matrix. MST'_{rs} denotes the MST$'$ of an aircraft of type r followed by another of type s
ΔMST: Differences between MST and MST$'$. $\Delta\mathrm{MST} = \mathrm{MST} - \mathrm{MST}'$. $\Delta\mathrm{MST}_{rs}$ denotes the ΔMST of an aircraft of type r followed by another of type s. All elements in ΔMST are nonnegative
SHO: Strongly hypooptimal
WV: Wake-vortex.

Notation

$\mathrm{allowed}_i$: A set of allowed aircraft for ants to visit followed by aircraft i
a, b: Parameters determining the relative importance of τ versus η
(i, j): Arc from aircraft i to aircraft j
n: The number of the total aircraft in the FCFS sequence
p: The number of aircraft types in MST
$\mathrm{Pb}(i, j)$: The probability of ants to choose arc(i, j)
$\mathrm{TP}(i)$: The aircraft type of the ith aircraft in FCFS
U: An integer set $U = \{(i, j) : i = 1, 2, \ldots, n; j = 1, 2, \ldots, n; i \neq j\}$
W_π: The makespan of sequence π in ASS-MST, that is, the landing time of the last aircraft in sequence π in ASS-MST
W'_π: The makespan of sequence π in ASS-MST$'$
ΔW_π: Additional makespan from W'_π to W_π; that is, $\Delta W_\pi = W_\pi - W'_\pi$
Δw^r_π: Additional makespan of adding aircraft of type r to a sequence
x_i: The landing time of aircraft i
α_r: The ability of the leading aircraft of type r to generate WV
β_s: The ability of the following aircraft of type s to bear disturbance
Λ: The aircraft type set
M: A large positive constant
F: The FSFC sequence. F^r is a subsequence where all aircraft of the same type r are put together according to their order in F
π: An aircraft landing sequence. π_i is the ith aircraft in sequence π, and π_{end} is the last aircraft in π
Π: The final aircraft landing sequence. Π_i is the ith aircraft in Π
δ_{ij}: Aircraft i lands before aircraft j ($\delta_{ij} = 1$) and i after j ($\delta_{ij} = 0$)
$\tau(i, j)$: Pheromone in arc(i, j) in AC. $\tau_0(i, j)$ is initial pheromone
$\eta(i, j)$: Heuristic information in arc(i, j), where $\eta(i, j) = 1/\mathrm{MST}_{\mathrm{TP}(i)\mathrm{TP}(j)}$
γ: Parameter modeling the pheromone evaporation ratio.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The authors are deeply indebted to the editors and referees for their invaluable suggestions and comments which will greatly improve the presentation of this paper. The work was partly supported by the National Natural Science Foundation of China (71071113, 71101106), a Foundation for the Author of National Excellent Doctoral Dissertation of PR China (200782), Shanghai Philosophical and Social Science Program (2010BZH003), and the Fundamental Research Funds for the Central Universities.

References

[1] Federal Aviation Administration, 2009, https://aspm.faa.gov/opsnet/sys/Main.asp?force=atads.

[2] D. Böhme, "Improved airport surface traffic management by planning," in *Advanced Technologies for Air Traffic Flow Management*, H. Winter and H.-G. Nüßer, Eds., vol. 198 of *Lecture Notes in Control and Information Sciences*, pp. 191–224, Springer, Berlin, Germany, 1994.

[3] K. D. Arkind, "Requirements for a novel terminal area capacity enhancement concept in 2022," in *American Institute of Aeronautics and Astronautics, Guidance, Navigation and Control Conference Exhibit*, American Institute of Aeronautics and Astronautics, Reston, Va, USA, 2004.

[4] H. R. Idris, B. Delcaire, I. Anagnostakis et al., "Identification of flow constraint and control points in depa rture operations at airport systems," in *Proceedings of the American Institute of Aeronautics and Ast ronautics, Guidance, Navigation and Control Conference*, American Institute of Aero nautics and Astronautics, Boston, Mass, USA, 1998, AIAA-1998-42 91.

[5] J. E. Beasley, M. Krishnamoorthy, Y. M. Sharaiha, and D. Abramson, "Scheduling aircraft landings—the static case," *Transportation Science*, vol. 34, no. 2, pp. 180–197, 2000.

[6] L. Bianco, P. Dell'Olmo, and S. Giordani, "Scheduling models and algorithms for TMA traffic management," in *Modelling and Simulation in Air Traffic Management*, L. Bianco, P. Dell'Olmo, and A. R. Odoni, Eds., pp. 139–167, Springer, New York, NY, USA, 1997.

[7] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman, New York, NY, USA, 1979.

[8] A. T. Ernst, M. Krishnamoorthy, and R. H. Storer, "Heuristic and exact algorithms for scheduling aircraft landings," *Networks*, vol. 34, no. 3, pp. 229–241, 1999.

[9] G. Bencheikh, J. Boukachour, A. E. H. Alaoui, and F. E. Khoukhi, "Hybrid method for aircraft landing scheduling based on a job shop formulation," *International Journal of Computer Science and Network Security*, vol. 9, pp. 78–88, 2009.

[10] M. C. Randall, "Scheduling aircraft landings using ant colony optimi sation," in *Proceedings of the IASTED International Conference on Artificial Intelligence and Soft Computing*, Banff, Canada, 2002.

[11] C. R. Brinton, "An implicit enumeration algorithm for arrival air craft scheduling," in *Proceedings of the IEEE/AIAA 11th Digital Avionics Systems Conference*, Seattle, Wash, USA, 1992.

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

[13] J. A. Bennell, M. Mesgarpour, and C. N. Potts, "Airport runway scheduling," *4OR*, vol. 9, no. 2, pp. 115–138, 2011.

[14] Z.-H. Zhan, J. Zhang, Y. Li et al., "An efficient ant colony system based on receding horizon control for the aircraft arrival sequencing and scheduling problem," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 2, pp. 399–412, 2010.

[15] M. Dorigo, *Optimization, learning and natural algorithms [Ph.D. thesis]*, Elettronica e Informazi One, Politecnico di Milano, Milano, Italy, 1992.

[16] M. Dorigo and L. M. Gambardella, "Ant colonies for the travelling salesman problem," *BioSystems*, vol. 43, no. 2, pp. 73–81, 1997.

[17] M. Dorigo and L. M. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 53–66, 1997.

[18] G. Bencheikh, J. Boukachour, and A. E. H. Alaoui, "Improved ant colony algorithm to solve the aircraft landing problem," *International Journal of Computer Theory and Engineering*, vol. 3, pp. 224–233, 2011.

[19] H. Balakrishnan and B. G. Chandran, "Algorithms for scheduling runway operations under constrained position shifting," *Operations Research*, vol. 58, no. 6, pp. 1650–1665, 2010.

[20] F. Neuman and H. Erzberger, "Analysis of delay reducing and fuel sav ing sequencing and spacing algor ithms for arrival spacing," NASA Technical Report A-91203; NAS 1.15:10 3880; NASA-TM-103880, NASA Technical Reports Server, 1991.

[21] T. R. Willemain, H. Fan, and H. Ma, "Statistical analysis of intervals between projected airport arrivals," DSES Technical Report 38-04-510, Rensselaer Polytechnic Institute, Troy, NY, USA, 2004.

[22] H. Lee, *Tradeoff evaluation of scheduling algorithms for terminal-area air traffic control [M.S. thesis]*, Massachusetts Institute of Technology, Cambridge, Mass, USA, 2008.