An Efficient Ant Colony System Based on Receding Horizon Control for the Aircraft Arrival Sequencing and Scheduling Problem

Zhi-Hui Zhan, *Student Member, IEEE*, Jun Zhang, *Senior Member, IEEE*, Yun Li, *Member, IEEE*, Ou Liu, S. K. Kwok, W. H. Ip, and Okyay Kaynak, *Fellow, IEEE*

***Abstract*—The aircraft arrival sequencing and scheduling (ASS) problem is a salient problem in air traffic control (ATC), which proves to be nondeterministic polynomial (NP) hard. This paper formulates the ASS problem in the form of a permutation problem and proposes a new solution framework that makes the first attempt at using an ant colony system (ACS) algorithm based on the receding horizon control (RHC) to solve it. The resultant RHC-improved ACS algorithm for the ASS problem (termed the RHC-ACS-ASS algorithm) is robust, effective, and efficient, not only due to that the ACS algorithm has a strong global search ability and has been proven to be suitable for these kinds of NP-hard problems but also due to that the RHC technique can divide the problem with receding time windows to reduce the computational burden and enhance the solution's quality. The RHC-ACS-ASS algorithm is extensively tested on the cases from the literatures and the cases randomly generated. Comprehensive investigations are also made for the evaluation of the influences of ACS and RHC parameters on the performance of the algorithm. Moreover, the proposed algorithm is further enhanced by using a two-opt exchange heuristic local search. Experimental results verify that the proposed RHC-ACS-ASS algorithm generally outperforms ordinary ACS without using the RHC technique and genetic algorithms (GAs) in solving the ASS problems and offers high robustness, effectiveness, and efficiency.**

***Index Terms*—Air traffic control (ATC), ant colony system (ACS), arrival sequencing and scheduling (ASS), receding horizon control (RHC).**

NOMENCLATURE

ACO	Ant colony optimization.
ACS	Ant colony system.
ALT	Assigned landing time.
ASS	Arrival sequencing and scheduling.
ATC	Air traffic control.
COP	Combinatorial optimization problem.
FCFS	First come first served.
GA	Genetic algorithm.
JSP	Job shop-scheduling problem.
LTI	Landing time interval.
PLT	Predicted landing time.
RHC	Receding horizon control.
TAD	Total airborne delay.
TSP	Traveling salesman problem.

Manuscript received March 22, 2009; revised November 2, 2009 and February 11, 2010; accepted February 12, 2010. Date of publication April 1, 2010; date of current version May 25, 2010. This work was supported in part by the National Natural Science Foundation of China Joint Fund with Guangdong under Key Project U0835002 and in part by the National High-Technology Research and Development Program ("863" Program) of China under Grant 2009AA01Z208. The Associate Editor for this paper was M. Brackstone.

Z.-H. Zhan and J. Zhang (Corresponding author) are with the Key Laboratory of Digital Life, Ministry of Education, and the Department of Computer Science, Sun Yat-Sen University, Guangzhou 510275, China (e-mail: junzhang@ieee.org).

Y. Li is with the Department of Electronics and Electrical Engineering, University of Glasgow, G12 8LT Glasgow, U.K.

O. Liu, S. K. Kwok, and W. H. Ip are with The Hong Kong Polytechnic University, Hung Hom, Hong Kong.

O. Kaynak is with the Department of Electrical and Electronics Engineering, Bogazici University, 34342 Istanbul, Turkey.

Digital Object Identifier 10.1109/TITS.2010.2044793

TABLE I
MINIMAL LTI BETWEEN THE AIRCRAFT [5]

LTI(i, j) Seconds		Type of the Later Landing Aircraft j			
		1	2	3	4
Type of the Earlier Landing Aircraft i	1	96	200	181	228
	2	72	80	70	110
	3	72	100	70	130
	4	72	80	70	90

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

I. INTRODUCTION

ASS IS one of the most significant problems in ATC [1]–[4]. With the development of airline industry, air traffic congestion has become increasingly more serious. Economic loss as a result of flight delays has become serious enough to call for urgent solutions. Building more airports or runways is considered not a realistic option because of practical constraints and investment costs. However, a promising approach is to more optimally schedule the aircraft arrival sequence, which is the ASS that can be formulated as a minimization problem. The objective of the ASS is to minimize the TAD by generating efficient landing sequences, assigning landing times for the arrival aircraft, and satisfying a set of practical operational constraints. A straightforward approach to the ASS problem is the FCFS algorithm, which is according to the aircraft's PLT at the runway. This would be a fair and perfect schedule approach if no operational time separation is needed for any two successive landings.

However, a practical ASS problem is subject to a number of constraints, such as the LTI between any two successive landings [5], at least for safety reasons. One set of reference data has been given by Bianco *et al.* [5], as shown in Table I, on four types of aircraft with different speed, capacity, weight,

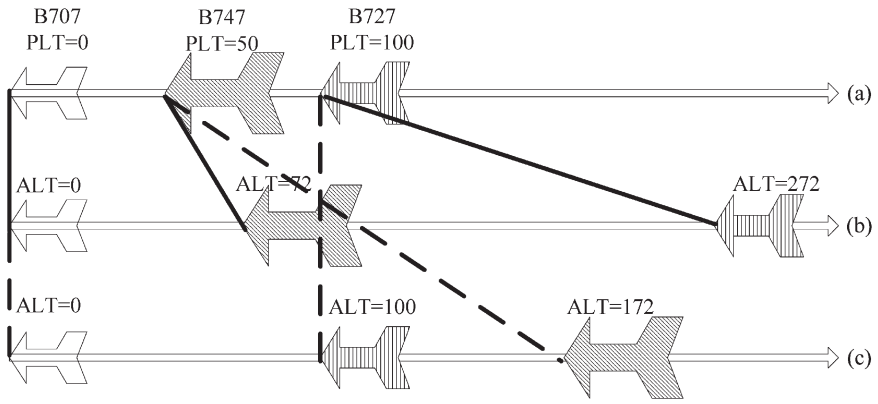


Fig. 1. Example of using position shift to improve the FCFS landing sequence. (a) Original sequence based on PLTs. (b) FCFS landing sequence by considering the LTI. Total delay is 194 s. (c) Optimal landing sequence by considering the LTI. Total delay is 122 s.

and other technical properties. As shown in the table, the LTI is not a constant because different pairs of aircraft and different relative landing sequences will both affect its value. Generally speaking, a smaller aircraft following a larger aircraft will require a longer separation than the other way around. For example, a Boeing 727 (B727) has to wait for 200 s after the landing of a Boeing 747 (B747). However, when a B747 lands after a B727, the LTI is only 72 s. These asymmetric characteristics of the LTI make the FCFS algorithm not always a good choice for the ASS problem. Fig. 1 shows an example of the FCFS approach when taking into account both PLT and LTI, where the TAD can be reduced by exchanging the landing sequence of the B747 and the B727. More substantially, the asymmetric nature of the LTI has made the ASS a nondeterministic polynomial (NP)-hard problem [5], [6]. Due to the significance and difficulty of the ASS problem, heuristic or metaheuristic optimization algorithms are in great need of investigation.

So far, many research efforts have been made, including the development of various formulations to model the problem and the use of variants of deterministic and heuristic algorithms [7]–[12]. Psaraftis [13], [14] and Bianco *et al.* [15] both modeled the ASS as a JSP and reported the use of dynamic programming approach. The TSP model was used in [5]. Beasley *et al.* [16] attempted to solve this problem by using a mixed-integer linear optimization program. Monte Carlo optimization [17] and constrained position shifting approaches [18] were also reported for solving the ASS problem. However, most of these studies schedule the sequence by assigning all the aircraft in the same process, whereas many researchers have argued that it is inefficient to consider too many aircraft at the same time because of the large search space. Moreover, scheduling all the aircraft in the same process is particularly difficult in a dynamic environment, where many inherent or unexpected disturbances may occur, such as delay of the aircraft, cancellation of flights, and emergency landing of some unanticipated aircraft [19], [20].

Therefore, it is important, as well as challenging, to develop an algorithm that could solve the ASS problem robustly, effectively, and efficiently. Recently, Hu and Chen [19] introduced the RHC concept to the study of the ASS problem. The RHC divides the ASS problem into a number of subproblems with a reduced search space and therefore can bring in a lighter computational burden and can achieve a higher solution quality. They also developed a GA based on the RHC to solve the ASS problem, leading to a further improved TAD and less computational burden [20]. This has been extended to a binary representation GA (BRGA) with an efficient crossover operator to enhance the search ability for optimal ASS [21]. The successes of applying GAs to the ASS problem [20]–[22] have advocated a strong potential of evolutionary computation algorithms in dealing with this type of NP-hard problems.

As an important branch of the evolutionary computation algorithms, ACO is an adaptive and global stochastic search and optimization algorithm. First reported in 1997, the ACS is an elaborately designed ACO approach to the TSP [23], which is a typical discrete COP. ACS has been proven to be very suitable and promising in solving various COPs. Many research reports have shown the effectiveness and efficiency of ACO/ACS in solving real-world problems, such as data mining [24], resource-constrained project scheduling [25], lithium-ion battery design [26], JSP [27], protein folding [28], fuzzy controller design [29], reclosers and distributed generators placements [30], grid workflow scheduling [31], power electronic circuit design [32], and block-layout design [33]. The ASS problem is also a COP, which can be modeled as a permutation problem similar to the TSP [5] or the JSP [13]–[15]. Therefore, ACS is very suitable and has its natural advantages to solve the ASS problem using its construction process to schedule the aircraft just like constructing the TSP tour by visiting the cities one by one, which has been demonstrated to be very effective and efficient.

This paper makes the first attempt to use the ACS algorithm to solve the ASS problem by incorporating an RHC strategy. The RHC aims to reduce the computational burden and enhance the solution quality. Moreover, the RHC helps to make the ACS algorithm tolerant to an uncertain dynamic environment with strong adaptive and global search ability. To develop the RHC-enhanced ACS algorithm for the ASS problem (RHC-ACS-ASS algorithm), several novel techniques and heuristics are studied so as to make the best use of the ACS algorithm and the problem-related information. First, an efficient heuristic information strategy is designed for the RHC-ACS-ASS algorithm.

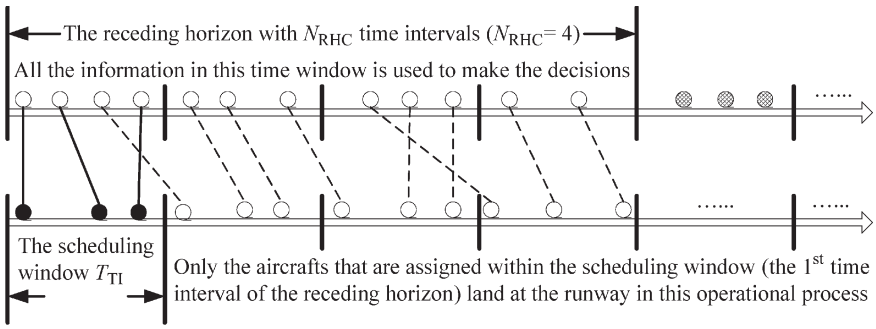


Fig. 2. Example of RHC for ASS.

Taking the PLT of the aircraft into account when selecting the next aircraft, this strategy should help the algorithm assign the most urgent aircraft to land as early as possible to reduce the TAD. Second, a two-opt exchange heuristic local search method, which is similar to that used in solving TSPs, is developed for the RHC-ACS-ASS algorithm to further enhance the solution's quality.

The rest of this paper is organized as follows. In Section II, the ASS problem formulation and the RHC concept are given, with a concise description of the ACS framework. Section III develops the RHC-ACS-ASS algorithm in detail. Experiments are carried out in Section IV, and test results are compared with other algorithms on robustness, effectiveness, and efficiency. Finally, conclusions are summarized in Section V, and future work is highlighted.

II. Background

A. ASS

The ASS problem involves a number of aircraft expected to land on the same runway on a daily basis. Assume that there are N aircraft needing to be scheduled during one operational day, and each aircraft has a respective PLT. The objective of the ASS problem is to find an optimal sequence for all the aircraft to land on the runway with a minimum TAD. In a specified landing sequence $\prod$ after scheduling, let $\prod(i)$, the ith element in sequence $\prod$, denote the order of aircraft i in the original sequence. For instance, $\prod(5) = 3$ means that the fifth aircraft in the scheduled sequence $\prod$ is the third aircraft in the original sequence before scheduling. By considering the operational constraint LTI, the ALT for each aircraft can be calculated as

$$\mathrm{ALT}\left(\prod(i)\right) = \begin{cases} \mathrm{PLT}\left(\prod(i)\right), & \text{if } i=1 \\ \max\{\mathrm{PLT}\left(\prod(i)\right), \mathrm{ALT}\left(\prod(i-1)\right) \\ \quad + \mathrm{LTI}\left(TP\left(\prod(i-1)\right), TP\left(\prod(i)\right)\right)\}, & \text{otherwise} \end{cases} \tag{1}$$

where $TP(i)$ is the aircraft type of the ith aircraft in the original sequence, and $\mathrm{LTI}(i,j)$ is the LTI of an aircraft of type j landing immediately after an aircraft of type i, refer to Table I for example. With all the ALTs determined, the TAD can be calculated as

$$\begin{aligned} \mathrm{TAD} = F &= \sum_{i=1}^{N} \left[\mathrm{ALT}\left(\prod(i)\right) - \mathrm{PLT}\left(\prod(i)\right)\right] \\ &= \sum_{j=\prod(1)}^{\prod(N)} \left[\mathrm{ALT}(j) - \mathrm{PLT}(j)\right]. \end{aligned} \tag{2}$$

The objective is thus to find an optimal $\prod_{\mathrm{OPT}}$ such that it minimizes the TAD defined by (2).

Note that another metric of objective is sometimes adopted in the literature, where the total operation time (i.e., the completion time of all aircraft for landing) is instead defined as the objective of the ASS problem [34]. The metric of the TAD emphasizes the operating cost of airlines, whereas the metric of the total operation time focuses on the best utilization of the runway (i.e., the maximum throughput of the runway). Although the two metrics are not equivalent, a minimum TAD can always offer a minimum total operation time [20]. Therefore, without loss of generality, the TAD metric is adopted in this paper.

B. RHC

Utilizing real-time optimization, RHC is a very effective online predictive control strategy [35]–[37]. The receding horizon is a sliding time frame, within which the original problem is solved as several smaller problems for a reduced computational burden. RHC can also cope with the real-time demand of uncertainties and disturbances in a dynamical environment [19]. This is due to that when there are uncertainties or disturbances in the environment, the RHC can perceive it and deals with it in the current or the following receding horizons. The RHC involves two parameters: 1) the time interval of a scheduling window T_{TI} and 2) the width of the receding horizon N_{RHC}, measured as the number of T_{TI}'s dividing the horizon. Fig. 2 illustrates how RHC works using an example of $N_{\mathrm{RHC}} = 4$. For a given horizon of interest, optimization is performed on information from the entire horizon, i.e., from all $N_{\mathrm{RHC}} = 4$ intervals. However, only scheduling decisions made for the first time interval are actually implemented. As illustrated in Fig. 2, the scheduling window is the first time interval in the receding horizon, i.e., optimization is globally made within the

horizon of interest, whereas scheduling is locally implemented in the first interval of the horizon. As the scale of each receding horizon is smaller than the whole problem, the computational burden of optimization is reduced such that it can be computed in real time. Moreover, as the solution space of each receding horizon is much smaller, the optimization approach can more efficiently perform the global search to obtain higher quality solution.

When applying the RHC strategy in an optimization problem, the problem is divided into a number of subproblems by the receding horizon. For the kth $(k = 1, 2, 3, \ldots)$ subproblem, environmental information is collected in the duration from the beginning of time interval k to the end of time interval $k + N_{\mathrm{RHC}} - 1$. The objective of optimization is for the kth time interval only. Then, the optimization process repeats for the $(k + 1)$th receding horizon until the entire problem is complete. By using the RHC strategy, the scale of each receding horizon is smaller than the whole problem, and therefore, the computational burden of optimization is reduced such that it can be computed in real time. Moreover, as the search space of each receding horizon is much smaller, the optimization approach can more efficiently perform the global search to obtain higher quality solution.

C. ACS

ACS is an effective and efficient global optimization algorithm that was first developed by Dorigo and Gambardella as a more efficient version of ACO [23]. ACS was originally designed for the TSP, and its framework is suitable for discrete COPs. In solving the TSP, a number of ants randomly start from various cities. Then, each ant constructs its tour by visiting all the cities one by one. On locating a city s, the ant chooses the next city r from the unvisited cities by considering the "pheromone" deposited on the edge of (s, r) and the heuristic information value of (s, r) together. Then, it moves from s to r and repeats this process until a complete tour is constructed. When an ant completes constructing such a potential solution, the pheromone on the edges of the solution path will evaporate by a "local pheromone updating rule." After all the ants complete their tour constructions, the best tour with the shortest length will be compared with the last historically best solution to determine the current historically best solution. The pheromone on the edges of the historically best solution is enhanced by a "global pheromone updating rule." Then, the algorithm moves to the next generation until a termination criterion is met.

III. Receding Horizon Control–Ant Colony System-Arrival Sequencing and Scheduling Algorithm for Solving the ASS

A. Aircraft for Scheduling Obtained by RHC

The first task in the scheduling process is to find all the aircraft whose PLT is within each receding horizon. Without loss of generality, considering the kth $(k = 1, 2, 3, \ldots)$ receding horizon, the time window is $\Omega(k) = [(k-1)T_{\mathrm{TI}}, (k + N_{\mathrm{RHC}} - 1)T_{\mathrm{TI}}]$, and the scheduling window is $\omega(k) = [(k-1)T_{\mathrm{TI}}, kT_{\mathrm{TI}}]$, indicating that only the aircraft whose ALTs are within this scheduling window can land on the runway during the kth receding horizon process.

As ω is always narrower than Ω (if $N_{\mathrm{RHC}} > 1$), there will be some aircraft that cannot be scheduled to land during the kth receding horizon process and, hence, need to be scheduled in the following processes. To make these aircraft available in these processes, their PLTs need to be modified if the aircraft will be missed when the receding horizon moves forward. For example, if $\mathrm{PLT}(i) \in \omega(k)$ and $\mathrm{ALT}(i) \notin \omega(k)$, i.e., if aircraft i's PLT is within the kth scheduling window but is not assigned in the kth receding horizon, then $\mathrm{PLT}(i)$ should be modified to kT_{TI}, making $\mathrm{PLT}(i) \in \Omega(k+1)$ follow so that the aircraft can be scheduled in the next receding horizon.

B. ACS Solution Construction

Here, an ACS is used to optimize the sequence of the aircraft that fall in the kth receding horizon. The objective of the ACS algorithm is to find an optimal sequence π such that

$$\min f = \sum_{i=1}^{M} \left(\mathrm{ALT}\left(\pi(i)\right) - \mathrm{PLT}^{*}\left(\pi(i)\right)\right) \tag{3}$$

is minimized, where M is the number of aircraft, and $\mathrm{PLT}^{*}(\pi(i))$ means that the original PLT of the $\pi(i)$ aircraft is used in the calculation but not that which has probably been modified in previous receding horizon processes.

During the solution construction process, techniques for how to determine the initial pheromone, how to select the first aircraft, and how to transit from one aircraft to another step by step to until completion are described below.

1) Initialization State Configurations: The initialization state configuration includes designing the initial pheromone τ_0 and determining the first scheduled aircraft. To design the initial pheromone τ_0, the FCFS approach is used to schedule the M aircraft and find a sequence π_{FCFS}. Then, the fitness of π_{FCFS} is calculated by (3), as f_{FCFS}. Similar to [23], we set $\tau_0 = (M \cdot f_{\mathrm{FCFS}})^{-1}$.

Then, the construction process randomly chooses an aircraft from M as the first aircraft to schedule. As soon as the first aircraft is determined, the ALT needs to be calculated and assigned to the aircraft. There are two conditions in this situation. The first condition is that the process is in the first receding horizon $(k = 1)$, and the second condition is that the process is not in the first receding horizon $(k > 1)$. Under the first condition, the ALT for the aircraft is just its PLT. Under the second condition, its ALT is the larger between its PLT and the LTI following the last aircraft in the previous receding horizon. Therefore, ALT calculation is defined as

$$\mathrm{ALT}(s) = \begin{cases} \mathrm{PLT}(s), & \text{if } k = 1 \\ \max\left(\mathrm{PLT}^{*}(s), \mathrm{ALT}(s^{*})\right. & \\ \qquad \left. + \mathrm{LTI}\left(TP(s^{*}), TP(s)\right)\right), & \text{otherwise} \end{cases} \tag{4}$$

where s^{*} is the last aircraft in the previous receding horizon.

2) State Transition Rule: In the RHC-ACS-ASS algorithm, the state transition rule is as follows: When an ant completes scheduling aircraft s, it then chooses the next aircraft r by applying the rule given in (5), shown at the bottom of the page.

In (5), the set J_s is the allowable aircraft that can be selected by the ant on the current aircraft s, making sure that each of the M aircraft is scheduled once and once only. The parameter $q_0 (0 \le q_0 \le 1)$ is used to control the exploitation and exploration behaviors of the ant. If a randomly generated number q in range [0, 1] is smaller than q_0, then the ant chooses the next aircraft u whose pheromone τ and heuristic η are maximal, measured by $[\tau(s,u)][\eta(s,u)]^\beta$, where β is a parameter that determines the relative importance of pheromone versus heuristic information $(\beta > 0)$ [23]. Otherwise, the next aircraft r will be determined as a random variable R using a probability selection as

$$p(s,r) = \begin{cases} \frac{[\tau(s,r)]\cdot[\eta(s,r)]^\beta}{\sum_{u\in J_s}[\tau(s,u)]\cdot[\eta(s,u)]^\beta}, & \text{if } r \in J_s \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

In the RHC-ACS-ASS algorithm, "reasonable" pheromone information τ and heuristic information η need to be designed so as to make their value ranges comparable with each other. In (5) and (6), η is the heuristic information that represents the urgency of each aircraft. The expected assigned time for the aircraft r is given in (7), whereas the heuristic information of $\eta(s,r)$ is given as

$$E(s,r) = \max\left(\text{PLT}(r), \text{ALT}(s) + \text{LTI}\left(TP(s), TP(r)\right)\right) \tag{7}$$

$$\eta(s,r) = \frac{1}{E(s,r) - \text{ALT}(s)}. \tag{8}$$

C. Pheromone Updating Rules

There are a local pheromone updating rule and a global pheromone updating rule in an ACS process. In the RHC-ACS-ASS algorithm developed here, the local pheromone updating operation is carried out on each aircraft pair (s,r) in the completed scheduled sequence as

$$\tau(s,r) = (1-\rho)\cdot\tau(s,r) + \rho\cdot\tau_0 \tag{9}$$

for each ant.

Conversely, the global pheromone updating operation is only performed on the best-so-far (historically best) solution π_{Best}. Only the pheromone on the aircraft pair of the sequence of π_{Best} is increased as

$$\begin{aligned} \tau(s,r) &= (1-\varepsilon)\cdot\tau(s,r) + \varepsilon\cdot\Delta\tau, \\ \text{where } \Delta\tau &= (f_{\pi_{\text{Best}}})^{-1}, \quad \text{if}(s,r) \in \pi_{\text{Best}}. \end{aligned} \tag{10}$$

These two pheromone updating rules are used to adjust the search behaviors of the ants. The effect of the local pheromone updating rule is to reduce the desirability of the visited edges by some ants. Evaporation of the pheromone on the visited edges makes them less attractive for the following ants. Hence, it is useful for increasing the population diversity. The effect of the global pheromone updating rule is to increase the desirability of the edges on the best-so-far solution. This reinforcement can result in a higher convergence speed.

D. Complete RHC-ACS-ASS Algorithm

The complete RHC-ACS-ASS algorithm is shown in Fig. 3 and is described in the following six steps.

- Step 1: Initialization. Set up parameters N_{RHC} and T_{TI} for the RHC, and set the current receding horizon $k = 1$.
- Step 2: Find out all the M aircraft whose PLTs belong to the kth receding horizon $\Omega(k) = [(k-1)T_{\text{TI}}, (k + N_{\text{RHC}} - 1)T_{\text{TI}}]$.
- Step 3: Schedule the M aircraft in the kth receding horizon by using an ACS.
- Step 4: Assign the aircraft whose ALTs belong to kth scheduled window $\omega(k) = [(k-1)T_{\text{TI}}, kT_{\text{TI}}]$ to land on the runway.
- Step 5: Modify the PLT for those aircraft whose PLT belongs to $\omega(k)$ but the ALT does not belong to $\omega(k)$. The modification is to set their PLT to kT_{TI}, making them belong to $\Omega(k+1)$, such that they can be scheduled in the next receding horizon.
- Step 6: Termination check. When all the aircraft have been assigned to land at the runway, the algorithm terminates. Otherwise, set $k = k + 1$ and go to Step 2 for the next receding horizon optimization.

In the preceding steps, Step 3 is the major process of the algorithm. The flowchart is illustrated on the right side of Fig. 3, and the details are given below.

- Step 3.1: Schedule the M aircraft by the FCFS approach and calculate the fitness value through (3). Calculate the initial pheromone τ_0 and set the pheromone for each aircraft pair as τ_0.
- Step 3.2: For each ant, do the following.
  - a) Determine the first landing aircraft s and construct the whole landing sequence using the state transition rule as (5) and (6).
  - b) Perform the local pheromone updating as (9).
- Step 3.3: Calculate the fitness of each ant and determine the best solution. Moreover, the current best solution is compared with the historically best solution to determine the historically best solution.
- Step 3.4: Perform the global pheromone updating as (10).

$$r = \begin{cases} \arg\max_{u\in J_s}\left\{[\tau(s,u)]\cdot[\eta(s,u)]^\beta\right\}, & \text{if } q \le q_0 \text{ (exploitation)} \\ R, & \text{otherwise (biased exploration)} \end{cases} \tag{5}$$

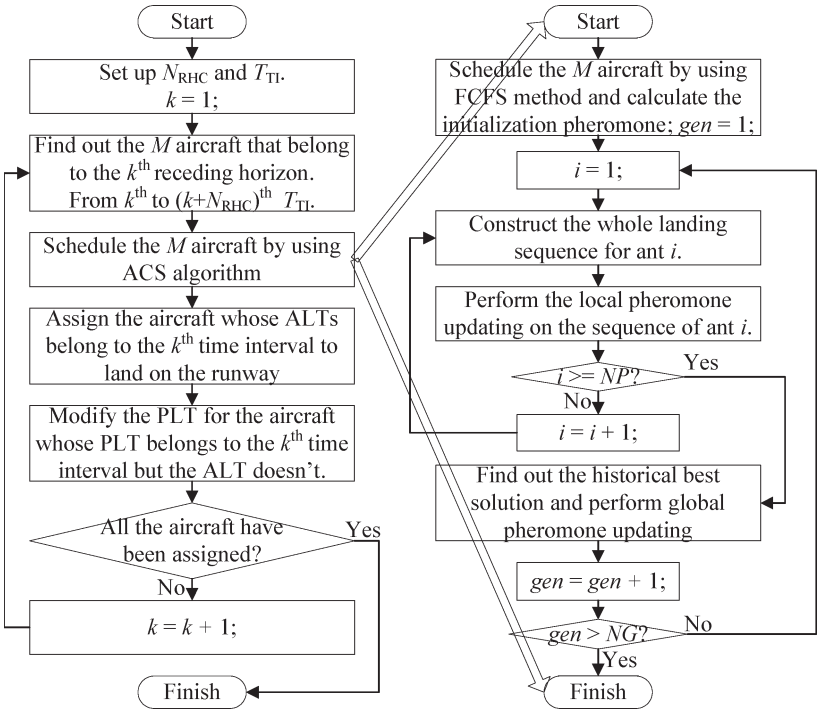


Fig. 3. Flowchart of the RHC-ACS-ASS algorithm.

Step 3.5: Termination check. If the termination criteria are met, e.g., the maximal generations, then the process stops. Otherwise, go to Step 3.2, and continue optimizing.

IV. Experiments and Comparisons

A. Test Cases

Experimental tests are carried out in this section to verify the robustness, effectiveness, and efficiency of the RHC-ACS-ASS algorithm. It is implemented in Visual C++ 6.0 on a PC running a Pentium Dual CPU at 2.0 GHz with 2.0-GB random access memory.

The test cases in [20] and [21] are used here to test the effectiveness of the RHC-ACS-ASS algorithm. The efficiency of the RHC-ACS-ASS is also tested by comparing with the corresponding results obtained by the FCFS approach and the GA-based approaches developed in [20] and [21]. The GA-based algorithms are compared because they were proven to outperform traditional approaches [20], [21]. Moreover, an ACS-ASS algorithm is implemented and compared with the RHC-ACS-ASS algorithm. The ACS-ASS algorithm directly applies the ACS algorithm to solve the ASS problem. It can be regarded that ACS-ASS is a special case of RHC-ACS-ASS, where the time interval T_{TI} is long enough to contain all the aircraft in an operational day. In other words, ACS-ASS does not use the RHC strategy and schedules all the aircraft at one time by using the ACS algorithm.

The parameter configurations are given in Table II. Both ACS-ASS and RHC-ACS-ASS use the same configurations for the ACS-related parameters. Two of the ACS-related parameters are the population size NP and the maximal generation number NG. They are set to five times the number of aircraft M in the current receding horizon. The other ACS-related parameters are $q_0 = 0.9$, $\rho = 0.9$, $\varepsilon = 0.1$, $\beta = 2.0$, and $\tau_0 = (M \cdot f_{\mathrm{FCFS}})^{-1}$. The RHC-related parameters are that $N_{\mathrm{RHC}} = 4$ and $T_{\mathrm{TI}} = 150$ s. These parameter configurations are based on empirical studies presented in Sections IV-B and C.

TABLE II
Parameter Configurations for the RHC-ACS-ASS Algorithm

ACS parameters							RHC parameters	
NP	NG	q_0	ρ	ε	β	τ_0	N_{RHC}	T_{TI}
$5M$	$5M$	0.9	0.9	0.1	2.0	$(M{\cdot}f_{\mathrm{FCFS}})^{-1}$	4	150 seconds

1) Case 1—30 Aircraft: In this test case, the PLTs for all the aircraft are adopted from [20], as presented in the first three columns of Table III. The results of FCFS, RHC-GA, ACS-ASS, and RHC-ACS-ASS are presented and compared. The results of RHC-GA given in [20] are adopted directly and compared in Table III.

It can be observed from the table that RHC-ACS-ASS obtains the best solution to this test case for a TAD of 3721 s. The FCFS algorithm yields a very poor solution for a TAD of 8027 s. When the GA-based algorithm is enhanced by RHC, RHC-GA performs better than ACS-ASS, which does not use the RHC strategy. However, this does not mean that the GA-based approach is better than the ACS-based approach at solving the ASS problem. The fact that RHC-GA outperforms ACS-ASS can also be caused by the advantages of the RHC strategy. To make a better comparison, we refer to the data provided in [20] and calculate the TAD according to the sequence presented in [20, Tab. II]. The results show that the solution obtained by the conventional dynamic optimization GA (CDO-GA) therein, which is a conventional GA without the RHC strategy, takes 6058 s for the TAD. This result is worse than that obtained by our ACS-ASS algorithm, which is

TABLE III
EXPERIMENTAL RESULT COMPARISONS IN CASE 1 WITH 30 AIRCRAFT

Data			FCFS				RHC-GA				ACS-ASS				RHC-ACS-ASS			
No.	TP	PLT	No.	TP	ALT	Delay	No.	TP	ALT	Delay	No.	TP	ALT	Delay	No.	TP	ALT	Delay
1	1	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
2	1	79	2	1	96	17	2	1	96	17	2	1	96	17	2	1	96	17
3	1	144	3	1	192	48	3	1	192	48	3	1	192	48	3	1	192	48
4	2	204	4	2	392	188	5	1	288	24	5	1	288	24	5	1	288	24
5	1	264	5	1	464	200	6	1	384	64	6	1	384	64	6	1	384	64
6	1	320	6	1	560	240	4	2	584	380	4	2	584	380	4	2	584	380
7	2	528	7	2	760	232	7	2	664	136	7	2	664	136	7	2	664	136
8	1	635	8	1	832	197	9	2	744	14	9	2	744	14	9	2	744	14
9	2	730	9	2	1032	302	10	2	824	58	10	2	824	58	10	2	824	58
10	2	766	10	2	1112	346	8	1	896	261	11	1	896	106	8	1	896	261
11	1	790	11	1	1184	394	11	1	992	202	12	1	992	72	11	1	992	202
12	1	920	12	1	1280	360	12	1	1088	168	8	1	1088	453	12	1	1088	168
13	3	1046	13	3	1461	415	15	2	1288	152	13	3	1269	223	13	3	1269	223
14	4	1106	14	4	1591	485	16	2	1368	202	16	2	1369	203	17	2	1369	136
15	2	1136	15	2	1671	535	17	2	1448	215	17	2	1449	216	15	2	1449	313
16	2	1166	16	2	1751	585	13	3	1518	472	15	2	1529	393	16	2	1529	363
17	2	1233	17	2	1831	598	14	4	1648	542	14	4	1639	533	14	4	1639	533
18	1	1642	18	1	1903	261	18	1	1720	78	18	1	1711	69	18	1	1711	69
19	1	1715	19	1	1999	284	19	1	1816	101	19	1	1807	92	19	1	1807	92
20	3	1770	20	3	2180	410	20	3	1997	227	20	3	1988	218	20	3	1988	218
21	1	2074	21	1	2252	178	21	1	2074	0	21	1	2074	0	21	1	2074	0
22	1	2168	22	1	2348	180	22	1	2170	2	22	1	2170	2	22	1	2170	2
23	4	2259	23	4	2576	317	23	4	2398	139	23	4	2398	139	23	4	2398	139
24	2	2427	24	2	2656	229	24	2	2478	51	25	1	2481	0	24	2	2478	51
25	1	2481	25	1	2728	247	25	1	2550	69	26	2	2681	2	25	1	2550	69
26	2	2679	26	2	2928	249	26	2	2750	71	24	2	2761	334	26	2	2750	71
27	3	2883	27	3	2998	115	27	3	2883	0	27	3	2883	0	27	3	2883	0
28	2	2982	28	2	3098	116	28	2	2983	1	28	2	2983	1	28	2	2983	1
29	1	3046	29	1	3170	124	29	1	3055	9	29	1	3055	9	29	1	3055	9
30	1	3091	30	1	3266	175	30	1	3151	60	30	1	3151	60	30	1	3151	60
Total Delay			8027				3763				3866 (CPU Time 3671 ms)				**3721** (CPU Time **218** ms)			

TABLE IV
COMPARISONS BETWEEN ACS-ASS AND RHC-ACS-ASS IN CASE 1

Algorithm	Worst	Best	Mean	Std. Dev	(Mean-Best)/Best	CPU Time (ms)	Best Ratio
ACS-ASS	4221	**3721**	3874.76	132.823	4.13%	3634.53	35%
RHC-ACS-ASS	**4075**	**3721**	**3730.34**	**53.6348**	**0.25%**	**222.18**	**97%**

3866 s for TAD. Conversely, when both GA- and ACS-based algorithms are combined with the RHC strategy, our RHC-ACS-ASS performs better than RHC-GA. These comparisons show that the ACS-based algorithm is very promising in solving the ASS problem, particularly when the algorithm is integrated with the RHC strategy.

Table IV presents comparisons between ACS-ASS and RHC-ACS-ASS on Case 1. These are the average results of 100 independent runs for each algorithm. It appears that a TAD of 3721 s is the best solution to this problem. RHC-ACS-ASS obtains this best solution with the "Best Ratio" of 97%, whereas ACS-ASS succeeds only 35% out of the 100 trials. The mean CPU time shows that RHC-ACS-ASS is much faster than ACS-ASS, whereas the mean and standard deviation show that RHC-ACS-ASS obtains a better solution than ACS-ASS.

These advantages are mainly due to the RHC strategy. By using the RHC strategy, RHC-ACS-ASS divides the whole ASS problem into a number of subproblems by the receding horizon and solves them one by one. On one hand, as fewer aircraft are considered in each receding horizon, the population size and the maximal generation will be smaller because they are set as five times the number of the aircraft being optimized (refer to Table II for the parameter settings). Although a number of subproblems have to be solved, the total computational burden is still lighter than the one needed without using RHC. Therefore, the RHC strategy can contribute to the reduced computational burden. On the other hand, as fewer aircraft are considered in each receding horizon, the search space will be smaller, and therefore the algorithm can more efficiently perform the global search to find a better solution. Moreover, in the ASS problem, it is unlikely that a very late aircraft is assigned to land very early, whereas a very early aircraft is assigned to land very late in an optimal solution. Therefore, it will not affect the global search ability of the algorithm by scheduling the aircraft one receding horizon after another receding horizon but will contribute to lighter computational burden and higher solution quality.

2) Case 2—20 Aircraft: Table V lists the test case with 20 aircraft in the first three columns. This case was also used in [21] to test the performance of BRGA. The results obtained by the proposed ACS-ASS and RHC-ACS-ASS algorithms are presented in Table V and are compared with those obtained by the FCFS and BRGA algorithms.

Similar to the experimental results in Case 1, the results shown in Table V also reveal that the ACS-based algorithms are still very promising. RHC-ACS-ASS not only outperforms

TABLE V
EXPERIMENTAL RESULT COMPARISONS IN CASE 2 WITH 20 AIRCRAFT

Data			FCFS				BRGA				ACS-ASS				RHC-ACS-ASS			
No.	TP	PLT	No.	TP	ALT	Delay	No.	TP	ALT	Delay	No.	TP	ALT	Delay	No.	TP	ALT	Delay
1	1	1935	9	4	35	0	9	4	35	0	9	4	35	0	9	4	35	0
2	3	400	5	3	142	0	5	3	142	0	5	3	142	0	5	3	142	0
3	4	879	10	1	307	0	10	1	307	0	10	1	307	0	10	1	307	0
4	1	328	4	1	403	75	4	1	403	75	4	1	403	75	4	1	403	75
5	3	142	12	2	603	241	19	1	499	5	19	1	499	5	19	1	499	5
6	2	1980	2	3	673	273	17	1	595	30	17	1	595	30	17	1	595	30
7	2	915	19	1	745	251	12	2	795	433	12	2	795	433	18	3	776	111
8	2	1814	17	1	841	276	18	3	865	200	2	3	865	465	2	3	846	446
9	4	35	18	3	1022	357	2	3	935	535	18	3	935	270	7	2	946	31
10	1	307	3	4	1152	273	7	2	1035	120	7	2	1035	120	12	2	1026	664
11	3	1414	7	2	1232	317	3	4	1145	266	15	4	1145	192	3	4	1136	257
12	2	362	15	4	1342	389	15	4	1235	282	3	4	1235	356	15	4	1226	273
13	4	1279	14	1	1414	434	13	4	1325	46	13	4	1325	46	13	4	1316	37
14	1	980	13	4	1642	363	20	2	1408	0	11	3	1414	0	20	2	1408	0
15	4	953	20	2	1722	314	11	3	1478	64	20	2	1514	106	11	3	1478	64
16	3	1726	11	3	1792	378	14	1	1550	570	14	1	1586	606	14	1	1550	570
17	1	565	16	3	1862	136	16	3	1731	5	16	3	1767	41	16	3	1731	5
18	3	665	8	2	1962	148	8	2	1831	17	8	2	1867	53	8	2	1831	17
19	1	494	1	1	2034	99	6	2	1980	0	6	2	1980	0	6	2	1980	0
20	2	1408	6	2	2234	254	1	1	2052	117	1	1	2052	117	1	1	2052	117
Total Delay			4578				2765				2915 (CPU Time 734 ms)				**2702** (CPU Time **156** ms)			

TABLE VI
COMPARISONS BETWEEN ACS-ASS AND RHC-ACS-ASS IN CASE 2

Algorithm	Worst	Best	Mean	Std. Dev	(Mean-Best)/Best	CPU Time (ms)	Best Ratio
ACS-ASS	3308	**2702**	3146.91	**138.858**	16.47%	738.59	1%
RHC-ACS-ASS	**3266**	**2702**	**2823.38**	160.547	**4.49%**	**160.62**	**45%**

FCFS but also does better than BRGA. Even the ACS-ASS without using the RHC strategy obtains a much better solution when compared with the FCFS algorithm. These results confirm that the ACS approach is suitable and promising in solving the ASS problem.

A further comparison is made between ACS-ASS and RHC-ACS-ASS in Table VI. These results are also the average of 100 independent runs. They also demonstrate the contributions of the RHC strategy in terms of reduced computational burden and improved solution quality, similar to those shown in Table IV.

B. Analysis of ACS Parameters

The ACS parameters include the computational burden-related parameters NP and NG and the performance-related parameters q_0, β, ρ, and ε. The parameters NP and NG, on one hand, affect the solution quality and, on the other hand, determine the computational burden. Hence, tradeoffs should be considered when setting up these two parameters. Moreover, it is useful to study the influences of the performance-related parameters to the solution quality.

1) Computational Burden Related Parameters: In the preceding experiments, the population size of ants NP and the maximal generations NG are both set as five times the aircraft number M in the receding horizon being optimized. In the following investigation, NP varies from M, $3M$, $5M$, $7M$ to $9M$, and NG varies from M to $10M$, making up the combinations for tests. All the other parameters remain the same as in Section IV-A. Each combination is tested to optimize both Cases 1 and 2. For each case, 30 independent runs are carried out, and the mean results are plotted in Figs. 4 and 5, respectively.

Figs. 4(a) and 5(a) show the influences of the parameters on the solution quality, whereas Figs. 4(b) and 5(b) show the influences of the parameters on the computational burden. As anticipated, the solution quality improves with the increase of the population size NP or the number of generations NG. However, as this increases the CPU time too, it is necessary to make tradeoffs between the solution quality and the computational burden. Since the solution quality is not very sensitive to NG when $NP \geq 5M$, and the improvement in quality is insignificant when NG increases from $5M$ to $10M$, setting both NP and NG to $5M$ would be a good choice.

2) Performance Related Parameters: The investigation begins with the parameters q_0. We set q_0 from 0.1 to 0.9 with a step length of 0.1. With each parameter configuration, the algorithm is used to optimize the preceding two test cases for 100 independent trials. The mean TAD and the successful rate are calculated and plotted in Fig. 6(a) and (b), respectively. The tendency of the curves indicates that it is better to use a larger q_0 for better performance. When q_0 is set to 0.8 or 0.9, the mean TAD becomes the smallest, and the successful rate is very high. However, q_0 cannot be set as 1.0 because this configuration makes the algorithm have no exploration ability and thus perform poorly. The results of the mean TAD for $q_0 = 1.0$ are too poor to plot within the scale of Fig. 6(a).

The next parameter tested is β. As shown in Fig. 7, β should not be too large, e.g., not larger than 5. For Case 1, the best β is 3, whereas for Case 2, it is 1. Moreover, the poor performance of the algorithm when β is 0 indicates that the heuristic information plays an important role in the

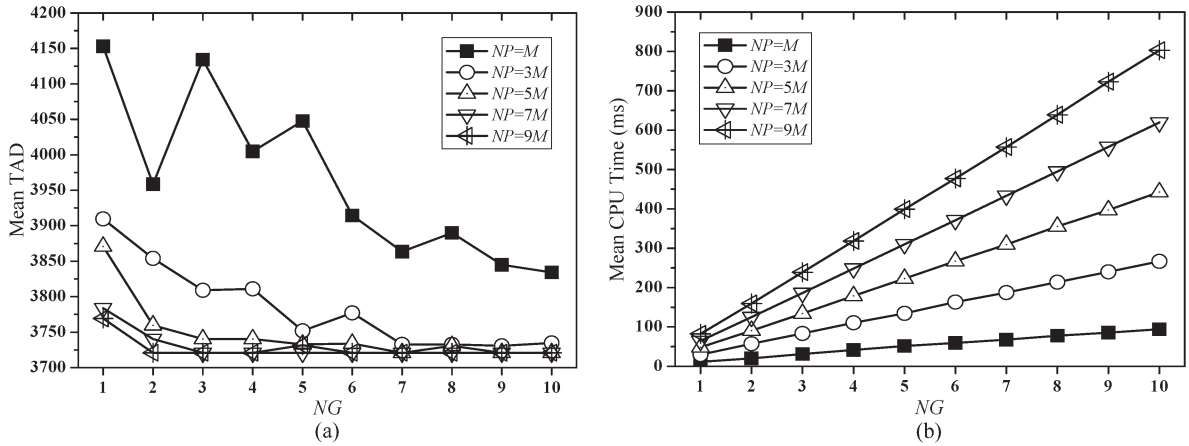


Fig. 4. Influence of the population size (NP) and the maximal generation (NG) on RHC-ACS-ASS in Case 1. (a) Mean TAD. (b) Mean CPU time (in milliseconds).

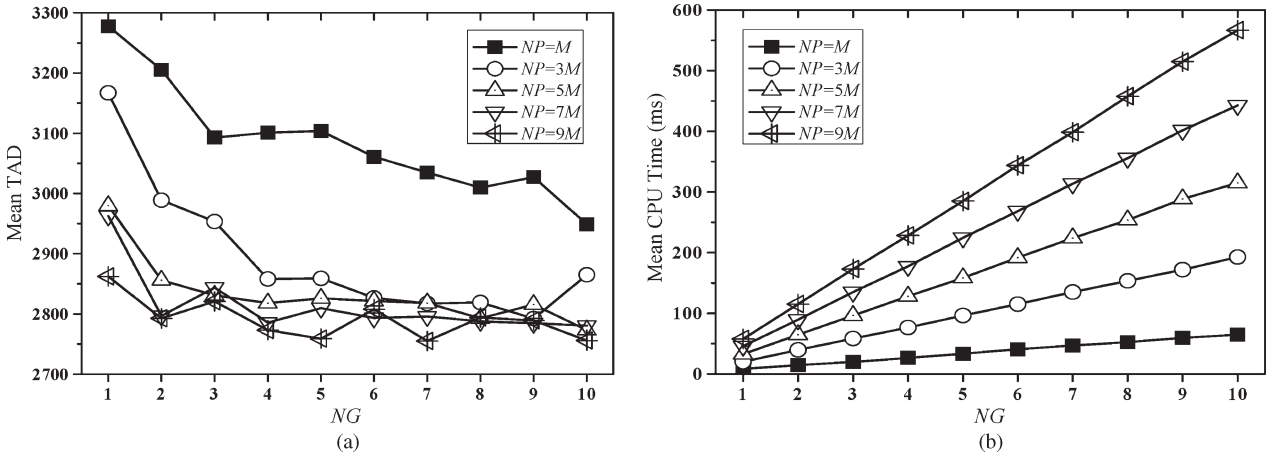


Fig. 5. Influence of the population size (NP) and the maximal generation (NG) on RHC-ACS-ASS in Case 2. (a) Mean TAD. (b) Mean CPU time (in milliseconds).

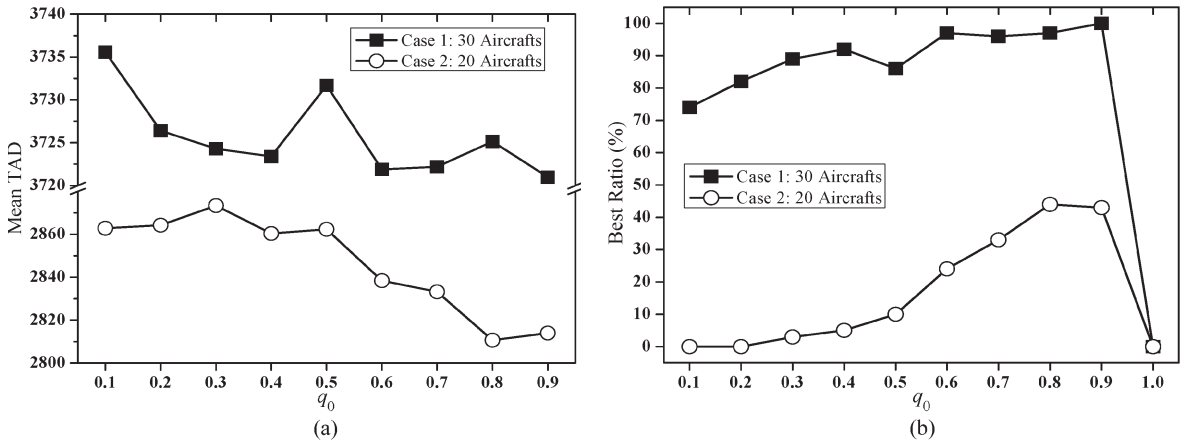


Fig. 6. Influence of the parameter q_0 on RHC-ACS-ASS. (a) Mean TAD. (b) Successful rate in reaching the best solution.

RHC-ACS-ASS algorithm. Similarly, the results for $\beta = 0$ are too poor to plot within the scale of Fig. 7(a).

Finally, the parameter ρ for local updating and the parameter ε for global updating are investigated. The results are plotted in Fig. 8, where Fig. 8(a) shows the solution quality for Case 1 and Fig. 8(b) for Case 2. It can be seen that for all values of ε, the TAD decreases when ρ increases in both Cases 1 and 2. As the parameter ρ represents the pheromone evaporation on the visited edges, a larger ρ reduces the accumulation of pheromone on the visited edges and increases the population diversity. In contrast, the figures show that the parameter ε for global updating is better to be small. Both a large ρ and a small ε support the functionality to let the ants construct as many different solutions as possible. This is useful for the

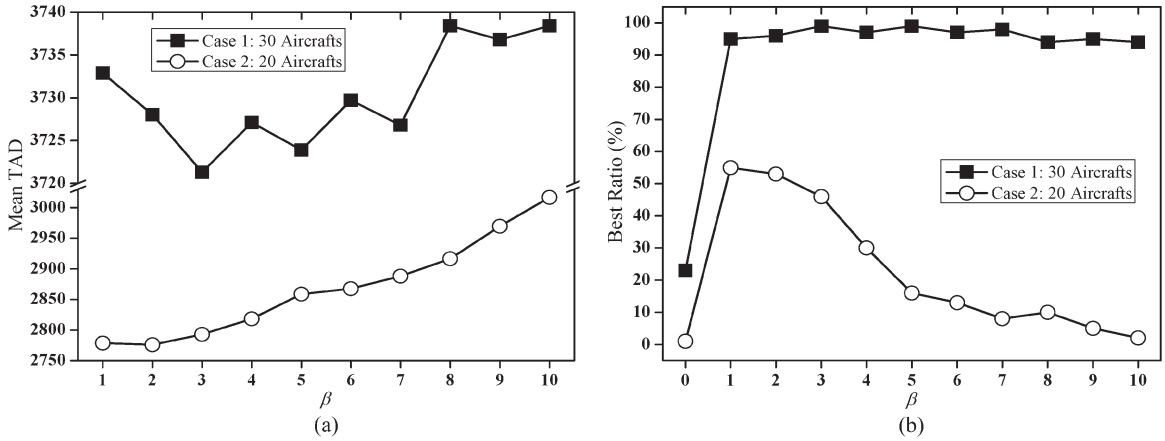


Fig. 7. Influence of the parameter β on RHC-ACS-ASS. (a) Mean TAD. (b) Successful rate in reaching the best solution.

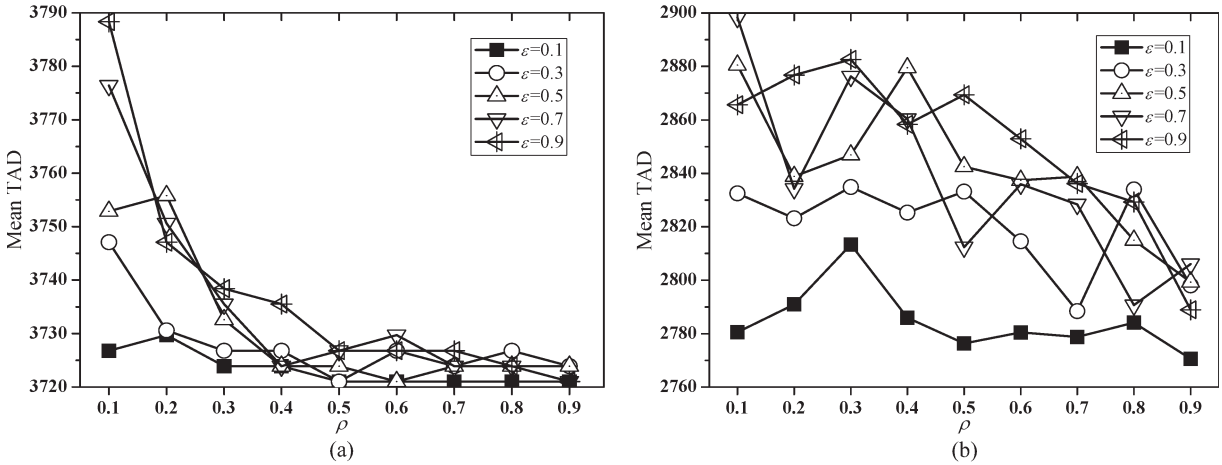


Fig. 8. Influence of the parameters ρ and ε on RHC-ACS-ASS. (a) Mean TAD in Case 1. (b) Mean TAD in Case 2.

algorithm to avoid being trapped in the local optima of a complex problem, such as the ASS problem. Hence, both Cases 1 and 2 need a large ρ and a small ε.

C. Analysis of RHC Parameters

There are two RHC-related parameters N_{RHC} and T_{TI} that can affect the performance of RHC-ACS-ASS. The receding horizon length N_{RHC} determines the sight of the algorithm. If N_{RHC} is too small, then little information can be used to decide the schedule. In contrast, a too large N_{RHC} may increase the computational burden. The time interval duration T_{TI} can affect the number of aircraft that are scheduled in the current receding horizon. These two parameters are investigated on both Cases 1 and 2. We carried out 100 independent runs for each case, and the mean results are plotted in Fig. 9.

Fig. 9(a) and (b) shows the results for Case 1, whereas Fig. 9(c) and (d) shows the results for Case 2. In this paper, we consider combinations for $T_{\mathrm{TI}} \in \{100, 150, 200\}$ and $N_{\mathrm{RHC}} \in \{1, 2, 3, 4, 5, 6\}$, whereas all the other parameters remain the same, as in Section IV-A.

From Fig. 9(b) and (d), we can see that the CPU time increases as the N_{RHC} increases. In addition, it consumes more CPU time when T_{TI} is larger. For the solution quality shown in Fig. 9(a) and (c), the mean TAD is better (smaller) when T_{TI} is 150 s. The algorithm yields very good results when N_{RHC} is set to 4. This is particularly evident in Fig. 9(c). Therefore, $N_{\mathrm{RHC}} = 4$ and $T_{\mathrm{TI}} = 150$ s are recommended for use.

D. Algorithm Robustness Tests

Although the advantages of the proposed RHC-ACS-ASS algorithm have been studied in the two test cases when compared with the FCFS and GA-based algorithms, it is still beneficial to test the algorithm robustness because RHC-ACS-ASS is a probabilistic algorithm. Robustness measures how well an algorithm manages to consistently obtain the global optimum, i.e., during all runs, on various test cases. Therefore, we examine the solution quality obtained by the proposed algorithm against the true optimal solutions. We randomly generate ten traffic data with each data set having ten aircraft. The type of each aircraft is randomly assigned, and its PLT is randomly generated within the time range [0 s, 500 s]. As the aircraft number is 10 in each test case, the true optimal solution can be obtained by an exhaustive search. For each test case, 100 independent runs are conducted under the RHC-ACS-ASS, and the BRGA algorithms and the experimental results are compared in Table VII. For RHC-ACS-ASS, the parameters are

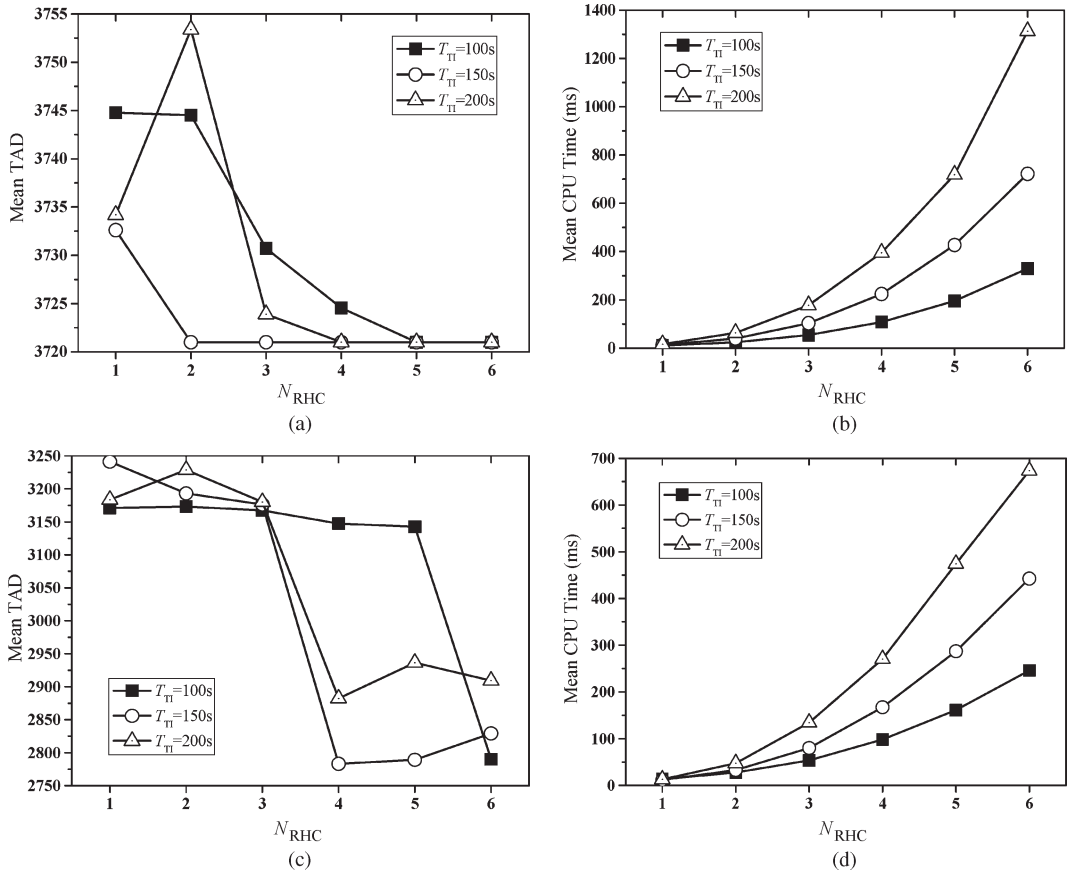


Fig. 9. (a) Mean TAD in Case 1 with different N_{RHC} and T_{TI}. (b) Mean CPU time (in milliseconds) in Case 1 with different N_{RHC} and T_{TI}. (c) Mean TAD in Case 2 with different N_{RHC} and T_{TI}. (d) Mean CPU time (in milliseconds) in Case 2 with different N_{RHC} and T_{TI}.

TABLE VII
EXPERIMENTAL RESULTS OF RHC-ACS-ASS AND BRGA COMPARED WITH THE OPTIMAL RESULTS

Cases	RHC-ACS-ASS			BRGA			OPT
	Mean	Ratio	Error	Mean	Ratio	Error	
case 1	2407	100%	0%	2427.48	69%	0.85%	2407
case 2	1945	100%	0%	2042.22	55%	5.00%	1945
case 3	1901	100%	0%	2140.62	36%	12.60%	1901
case 4	2500	100%	0%	2521.99	77%	0.88%	2500
case 5	1955	100%	0%	2031.57	80%	3.92%	1955
case 6	1779	100%	0%	1779.90	99%	0.05%	1779
case 7	931	100%	0%	944.39	96%	1.44%	931
case 8	2930	100%	0%	3043.19	46%	3.86%	2930
case 9	1733	100%	0%	1734.89	97%	0.11%	1733
case 10	1588	100%	0%	1609.86	81%	1.38%	1588

Error=(Mean –OPT)/OPT

the same as those used in Section IV-A, whereas for BRGA, the parameters are the same as those used in [21].

Table VII shows that RHC-ACS-ASS can obtain the optimal solutions in all ten cases with 100% successful rate in the 100 runs. In contrast, BRGA has experienced difficulties in consistently obtaining optimal solutions on all the test cases and even results in error values no smaller than 5% in Cases 2 and 3. Overall, the experimental results confirm that the RHC-ACS-ASS algorithm developed in this paper is more robust than BRGA in reaching the global optimal solution. The good performance of RHC-ACS-ASS also demonstrates the effectiveness of the parameter settings investigated in the preceding sections.

E. Algorithm Efficiency Tests

Due to the stochastic optimization nature of the ACS algorithm, only statistical conclusions can be made by evaluating the algorithm effectiveness and efficiency using comprehensive experiments and comparisons. In this section, we carry out experiments on an uncongested situation, a normal situation, and a congested situation with 30 aircraft as well as with 60 aircraft. Hence, six situations are simulated in total. In each situation, to reduce the stochastic influence of the test case, 20 sets of traffic data are generated. Each aircraft is assigned with a random type and a random PLT. For the 30 aircraft traffic data, the PLT is within the time range of [0 s, 4500 s], [0 s, 3000 s], and [0 s, 1500 s] for the uncongested, normal, and congested situations,

TABLE VIII
EXPERIMENTAL RESULTS OF DIFFERENT ALGORITHMS ON DIFFERENT CONGESTED SITUATIONS WITH 30 AND 60 AIRCRAFT

Algorithms	30 Aircraft			60 Aircraft		
	Uncongested	Normal	Congested	Uncongested	Normal	Congested
FCFS	3286.85	10145.50	26871.50	8174.10	30798.50	103404.48
BRGA	**2356.49**	6973.77	20164.69	5744.94	17414.95	77497.74
RHC-ACS-ASS	2366.18	**6616.62**	**18343.49**	**5679.90**	**16060.31**	**67720.50**
R1	28.31%	31.26%	24.96%	29.72%	43.46%	25.05%
R2	28.01%	34.78%	31.74%	30.51%	47.85%	34.51%
R = R2 – R1	-0.29%	3.52%	6.78%	0.80%	4.40%	9.46%

R1=(FCFS – BRGA)/FCFS, R2=(FCFS – RHC-ACS-ASS)/FCFS

TABLE IX
EXPERIMENTAL RESULTS OF RHC-ACS-ASS WITH AND WITHOUT LOCAL SEARCH ON DIFFERENT CONGESTED SITUATIONS WITH 30 AND 60 AIRCRAFT

RHC-ACS-ASS	30 Aircraft			60 Aircraft		
	Uncongested	Normal	Congested	Uncongested	Normal	Congested
Without local search	2366.18	6616.62	18343.49	5679.90	16060.31	67720.50
With local search	**2300.88**	**6451.20**	**17464.53**	**5534.70**	**15517.64**	**66002.42**
Improved R	2.76%	2.50%	4.79%	2.56%	3.38%	2.54%

R=(Without local search – With local search)/Without local search

respectively, whereas the PLT for the 60 aircraft traffic data is within the time range of [0 s, 9000 s], [0 s, 6000 s], and [0 s, 3000 s] for the uncongested, normal, and congested situations, respectively. In the simulations, both BRGA and RHC-ACS-ASS run 100 independent times on each traffic data, and the average TAD of the 100 runs is calculated. Therefore, we obtain 20 average TAD values for each situation (i.e., uncongested, normal, and congested situations with 30 and 60 aircraft) because 20 traffic data are randomly generated for test in each situation. The mean of these 20 values is calculated and is presented in Table VIII. In addition, the mean of the TAD obtained by the FCFS approach on the 20 traffic data of each situation is presented in Table VIII.

Table VIII shows that both BRGA and RHC-ACS-ASS can obtain better results than FCFS. To make a more comprehensive comparison, we use two metrics $\mathrm{R1} = (\mathrm{FCFS} - \mathrm{BRGA})/\mathrm{FCFS}$ and $\mathrm{R2} = (\mathrm{FCFS} - \mathrm{RHC\text{-}ACS\text{-}ASS})/\mathrm{FCFS}$ to evaluate the performance improved by BRGA and RHC-ACS-ASS, respectively. The data show that BRGA can improve the solution by at least 24.96% when compared with FCFS, whereas RHC-ACS-ASS can improve the solution by at least 28.01%. The metric $\mathrm{R} = \mathrm{R2} - \mathrm{R1}$ is also presented in Table VIII to compare the performance of BRGA and RHC-ACS-ASS. BRGA performs slightly better than RHC-ACS-ASS on the uncongested situation with 30 aircraft, whereas RHC-ACS-ASS wins on all the other five situations. Moreover, the results demonstrate that as the situation becomes more complicated, e.g., becomes more congested and with more aircraft, the advantage of RHC-ACS-ASS becomes more evident. It should be noted that the parameters of RHC-ACS-ASS are set the same as those in Section IV-A. The results indicate that these parameter configurations still work well on the test cases here, confirming that RHC-ACS-ASS is effective and efficient in obtaining a good solution to the ASS problem.

F. Experiments With Local Search

Here, the RHC-ACS-ASS algorithm is hybridized with a two-opt exchange local search, and the results are shown in Table IX. The two-opt heuristic was originally developed for solving the TSP [38]. In general, the two-opt heuristic deletes two edges from the TSP tour and adds two new edges to form a new tour. If the exchange results in a shorter route, then keep the new edges; otherwise, try to improve the tour by deleting other edges and adding other new edges.

The two-opt heuristic designed for the TSP focuses on exchanging the edges. However, the ASS problem is not exactly the same as the TSP in that the ASS focuses on the aircraft positions. Therefore, a special two-opt exchange heuristic that is suitable for the ASS problem is designed here. This new two-opt exchange heuristic is carried out as a local search operator after the solution has been obtained by the RHC-ACS-ASS algorithm. The procedure exchanges the positions of every two aircraft based on the obtained solution to form a set of new solutions. Among all the new obtained solutions, the solution with the shortest TAD can be determined. If it is better than the original solution, then it is accepted, and the procedure continues to exchange the positions of every two aircraft according to this new obtained best solution. The procedure terminates until no better solution can be obtained by exchanging the positions of the aircraft.

We carried out experiments based on the traffic data generated in Section IV-E by using the RHC-ACS-ASS algorithm with the two-opt exchange heuristic local search. The results are compared with those obtained by the RHC-ACS-ASS algorithm without the two-opt exchange heuristic local search in Table IX, revealing that the solution quality can be improved by this local search. For the 30 aircraft test cases, the effect of the local search is most evident in the congested situation. This is probably because of the high complexity of this situation Therefore, if RHC-ACS-ASS encounters difficulty in obtaining the best possible solution, the local search can help improve the solution. Note that in the 60 aircraft test cases, the effect of the local search appears to be most evident in a normal situation.

V. CONCLUSION

This paper has modeled the ASS problem in the form of a permutation problem and has, hence, proposed a new solution framework. An efficient ACS has been developed to solve

this NP-hard problem by incorporating an RHC strategy. The resulting RHC-ACS-ASS algorithm exhibits very good global search ability and solves the ASS problem well. With the help of the RHC strategy, RHC-ACS-ASS can not only reduce the computational burden but also improve the solution accuracy. The proposed algorithm has been described in detail and tested on a number of simulation cases. Extensive experimental results show that the RHC-ACS-ASS algorithm not only outperforms GA-based algorithms but also ACS-based algorithm without an RHC strategy. For complex cases, the algorithm performance can further be enhanced by incorporating a two-opt exchange heuristic local search. Investigations into the influences of the ACS and RHC parameters on the performance of RHC-ACS-ASS confirm the robustness, effectiveness, and efficiency of the proposed algorithm.

Further research work includes the following: 1) using real-time data and test cases from the airport to further test and refine the algorithm; 2) applying the algorithm to the dynamical environment with uncertainties and disturbances; and 3) extending the algorithm to solving the ASS problem with multiple runways.

ACKNOWLEDGMENT

The authors would like to thank the associate editor and reviewers for their valuable comments and suggestions that have enhanced the quality of this paper.

REFERENCES

[1] M. Pelegrin, *Towards Global Optimization for Air Traffic Management*, 1994, AGARD-AG-321.

[2] A. Bicchi and L. Pallottino, "On optimal cooperative conflict resolution for air traffic management systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 1, no. 4, pp. 221–232, Dec. 2000.

[3] K. Treleaven and Z. H. Mao, "Conflict resolution and traffic complexity of multiple intersecting flows of aircraft," *IEEE Trans. Intell. Transp. Syst.*, vol. 9, no. 4, pp. 633–643, Dec. 2008.

[4] Y. Wan and S. Roy, "A scalable methodology for evaluating and designing coordinated air-traffic flow management strategies under uncertainty," *IEEE Trans. Intell. Transp. Syst.*, vol. 9, no. 4, pp. 644–656, Dec. 2008.

[5] 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. New York: Springer-Verlag, 1997, pp. 139–167.

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

[7] 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, Oct. 1999.

[8] A. Andreussi, L. Bianco, and S. Ricciardelli, "A simulation model for aircraft sequencing in the near terminal area," *Eur. J. Oper. Res.*, vol. 8, no. 4, pp. 345–354, Dec. 1981.

[9] J. Milan, "The flow management problem in air traffic control: A model of assigning priorities for landings at a congested airport," *Transp., Planning Technol.*, vol. 20, no. 2, pp. 131–162, Feb. 1997.

[10] R. G. Dear and Y. S. Sherif, "The dynamic scheduling of aircraft in high density terminal areas," *Microelectron. Reliab.*, vol. 29, no. 5, pp. 743–749, 1989.

[11] C. S. Venkatakrishnan, A. Barnett, and A. R. Odoni, "Landings at Logan airport: Describing and increasing airport capacity," *Transp. Sci.*, vol. 27, no. 3, pp. 221–227, Aug. 1993.

[12] J. E. Beasley, J. Sonander, and P. Havelock, "Scheduling aircraft landings at London Heathrow using a population heuristic," *J. Oper. Res. Soc.*, vol. 52, no. 5, pp. 483–493, May 2001.

[13] H. N. Psaraftis, *A Dynamic Programming Approach to the Aircraft Sequencing Problem*. Cambridge, MA: MIT Press, 1978.

[14] H. N. Psaraftis, "A dynamic programming approach for sequencing groups of identical jobs," *Oper. Res.*, vol. 28, no. 6, pp. 1347–1359, Nov./Dec. 1980.

[15] L. Bianco, S. Ricciardelli, G. Rinaldi, and A. Sassano, "Scheduling tasks with sequence-dependent processing times," *Nav. Res. Logist.*, vol. 35, no. 2, pp. 177–184, Apr. 1988.

[16] J. E. Beasley, M. Krishnamoorthy, Y. M. Sharaiha, and D. Abramson, "Scheduling aircraft landings—The static case," *Transp. Sci.*, vol. 34, no. 2, pp. 180–197, May 2000.

[17] A. Lecchini Visintini, W. Glover, J. Lygeros, and J. Maciejowski, "Monte Carlo optimization for conflict resolution in air traffic control," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 4, pp. 470–482, Dec. 2006.

[18] H. Balakrishnan and B. Chandran, "Scheduling aircraft landings under constrained position shifting," presented at the AIAA Guidance, Navigation Control Conf. Exhib., Keystone, CO, Aug. 21–24, 2006, Paper AIAA 2006-6320.

[19] X. B. Hu and W. H. Chen, "Receding horizon control for aircraft arrival sequencing and scheduling," *IEEE Trans. Intell. Transp. Syst.*, vol. 6, no. 2, pp. 189–197, Jun. 2005.

[20] X. B. Hu and W. H. Chen, "Genetic algorithm based on receding horizon control for arrival sequencing and scheduling," *Eng. Appl. Artif. Intell.*, vol. 18, no. 5, pp. 633–642, Aug. 2005.

[21] X. B. Hu and E. D. Paolo, "Binary-representation-based genetic algorithm for aircraft arrival sequencing and scheduling," *IEEE Trans. Intell. Transp. Syst.*, vol. 9, no. 2, pp. 301–310, Jun. 2008.

[22] J. V. Hansen, "Genetic search methods in air traffic control," *Comput. Oper. Res.*, vol. 31, no. 3, pp. 445–459, Mar. 2004.

[23] M. Dorigo and L. M. Gambardella, "Ant colony system: A cooperative learning approach to the traveling salesman problem," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 53–66, Apr. 1997.

[24] R. S. Parpinelli, H. S. Lopes, and A. A. Freitas, "Data mining with an ant colony optimization algorithm," *IEEE Trans. Evol. Comput.*, vol. 6, no. 4, pp. 321–332, Aug. 2002.

[25] D. Merkle, M. Middendorf, and H. Schmeck, "Ant colony optimization for resource-constrained project scheduling," *IEEE Trans. Evol. Comput.*, vol. 6, no. 4, pp. 333–346, Aug. 2002.

[26] Y.-H. Liu, J.-H. Teng, and Y.-C. Lin, "Search for an optimal rapid charging pattern for lithium-ion batteries using ant colony system algorithm," *IEEE Trans. Ind. Electron.*, vol. 52, no. 5, pp. 1328–1336, Oct. 2005.

[27] J. Zhang, X. M. Hu, X. Tan, J. H. Zhong, and Q. Huang, "Implementation of an ant colony optimization technique for job scheduling problem," *Trans. Inst. Meas. Control*, vol. 28, no. 1, pp. 93–108, 2006.

[28] X. M. Hu, J. Zhang, J. Xiao, and Y. Li, "Protein folding in hydrophobic-polar lattice model: A flexible ant colony optimization approach," *Protein Pept. Lett.*, vol. 15, no. 5, pp. 469–477, 2008.

[29] C. Juang, C. Lu, C. Lo, and C. Wang, "Ant colony optimization algorithm for fuzzy controller design and its FPGA implementation," *IEEE Trans. Ind. Electron.*, vol. 55, no. 3, pp. 1453–1462, Mar. 2008.

[30] L. Wang and C. Singh, "Reliability-constrained optimum placement of reclosers and distributed generators in distribution networks using an ant colony system algorithm," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 38, no. 6, pp. 757–764, Nov. 2008.

[31] W. N. Chen and J. Zhang, "Ant colony optimization approach to grid workflow scheduling problem with various QoS requirements," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 39, no. 1, pp. 29–43, Jan. 2009.

[32] J. Zhang, S. H. Chung, W. L. Lo, and T. Huang, "Extended ant colony optimization algorithm for power electronic circuit design," *IEEE Trans. Power Electron.*, vol. 24, no. 1, pp. 147–162, Jan. 2009.

[33] B. R. Ke, M. C. Chen, and C. L. Lin, "Block-layout design using MAX–MIN ant system for saving energy on mass rapid transit systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 10, no. 2, pp. 226–235, Jun. 2009.

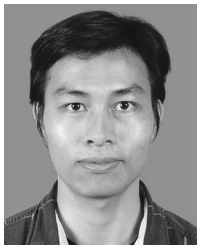
[34] D. Harikiopoulo and N. Neogi, "Polynomial time feasibility condition for multi-class aircraft sequencing on a single runway airport," in *Proc. AIAA 1st Intell. Syst. Tech. Conf.*, Chicago, IL, Sep. 20–22, 2004, pp. 1–13.

[35] D. Q. Mayne and H. Michalska, "Receding horizon control of nonlinear systems," *IEEE Trans. Autom. Control*, vol. 35, no. 7, pp. 814–824, Jul. 1990.

[36] H. Kashiwagi and Y. Li, "Nonparametric nonlinear model predictive control," *Korean J. Chem. Eng.*, vol. 21, no. 2, pp. 329–337, Mar. 2004.

[37] X. B. Hu, W. H. Chen, and E. D. Paolo, "Multiairport capacity management: Genetic algorithm with receding horizon," *IEEE Trans. Intell. Transp. Syst.*, vol. 8, no. 2, pp. 254–263, Jun. 2008.

[38] S. Lin and B. W. Kernighan, "An effective heuristic algorithm for the traveling salesman problem," *Oper. Res.*, vol. 21, no. 2, pp. 498–516, Mar./Apr. 1973.



Zhi-Hui Zhan (S'09) received the Bachelor's degree in computer science and technology in 2007 from Sun Yat-Sen University, Guangzhou, China, where he is currently working toward the Ph.D. degree.

His current research interests include optimization algorithms, ant colony optimization, particle swarm optimization, genetic algorithms, and differential evolution and their intelligent applications in real-world problems.

Jun Zhang (M'02–SM'08) received the Ph.D. degree in electrical engineering from the City University of Hong Kong, Kowloon, Hong Kong, in 2002.

From 2003 to 2004, he was a Brain Korean 21 Postdoctoral Fellow with the Department of Electrical Engineering and Computer Science, Korea Advanced Institute of Science and Technology, Daejeon, Korea. Since 2004, he has been with the Sun Yat-Sen University, Guangzhou, China, where he is currently a Professor and a Ph.D. Supervisor with the Department of Computer Science. He has authored seven research books and book chapters and over 80 technical papers in these research areas. His research interests include computational intelligence, data mining, wireless sensor networks, operations research, and power electronic circuits.

Dr. Zhang is currently an Associate Editor of the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS. He is currently the Chair of the IEEE Beijing (Guangzhou) Section, Computational Intelligence Society Chapter.

Yun Li (S'87–M'90) received the B.Sc. degree in radio electronics science from Sichuan University, Chengdu, China, in 1984, the M.Eng. degree in electronic engineering from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 1987, and the Ph.D. degree in computing and control engineering from the University of Strathclyde, Glasgow, U.K., in 1990.

From 1989 to 1990, he was with the U.K. National Engineering Laboratory and with Industrial Systems and Control Limited, Glasgow. In 1991, he was a Lecturer with the University of Glasgow. In 2002, he was a Visiting Professor with Kumamoto University, Kumamoto, Japan. He is currently a Senior Lecturer with the University of Glasgow and a Visiting Professor with UESTC. In 1996, he independently invented the "indefinite scattering matrix" theory, which opened up a groundbreaking way for microwave feedback circuit design. From 1987 to 1991, he carried out leading work in parallel processing for recursive filtering and feedback control. In 1992, he achieved first symbolic computing for power electronic circuit design without needing to invert a matrix: complex-numbered or not. Since 1992, he has pioneered the design automation of control systems and the discovery of novel systems using evolutionary learning and intelligent search techniques. He has supervised 15 Ph.D. students in this area and has over 140 publications.

Dr. Li is a Chartered Engineer and a member of the Institution of Engineering and Technology. He established the IEEE Computer-Aided Control System Design (CACSD) Evolutionary Computation Working Group and the European Network of Excellence in Evolutionary Computing Workgroup on Systems, Control, and Drives in 1998.

Ou Liu received the Ph.D. degree in information systems from the City University of Hong Kong, Kowloon, Hong Kong, in 2006.

Since 2006, he has been with The Hong Kong Polytechnic University, Hung Hom, Hong Kong, where he is currently an Assistant Professor with the School of Accounting and Finance. His research interests include business intelligence, decision support systems, ontology engineering, genetic algorithms, ant-colony systems, fuzzy logic, and neural networks.

S. K. Kwok received the B.Eng. and Ph.D. degrees in manufacturing engineering from The Hong Kong Polytechnic University, Hung Hom, Hong Kong, in 1991 and 1997, respectively.

He is currently a Lecturer with The Hong Kong Polytechnic University. His research interests include artificial intelligence, industrial and systems engineering, information and communication technologies, logistics-enabling technologies, and mobile commerce.

W. H. Ip received the M.Sc. degree from Cranfield University, Cranfield, U.K., in 1983, the M.B.A. degree from Brunel University, Uxbridge, U.K., in 1989, and the Ph.D. degree in manufacturing engineering from Loughborough University, Loughborough, U.K., in 1993.

He is currently an Associate Professor with the Industrial and Systems Engineering Department, The Hong Kong Polytechnic University, Hung Hom, Hong Kong. Over the span of 20 years of work in industry, education, and consulting, he has published more than 100 international journals and conference articles in the area of computer and operational research, artificial intelligence, and decision-support systems. His research interests include the modeling of applications of intelligent algorithms in planning and scheduling, evolutionary computing, and computer vision.

Dr. Ip is a member of the Hong Kong Institution of Engineers, a member of the Institution of Engineering and Technology, and a member of the Institution of Mechanical Engineers.

Okyay Kaynak (M'80–SM'90–F'03) received the B.Sc. (first-class honors) and Ph.D. degrees in electronic and electrical engineering from the University of Birmingham, Birminghan, U.K., in 1969 and 1972, respectively.

From 1972 to 1979, he held various positions within industry. In 1979, he joined the Department of Electrical and Electronics Engineering, Bogazici University, Istanbul, Turkey, where he is currently a Full Professor, holding the UNESCO Chair in Mechatronics. He has hold long-term (near or more than a year) Visiting Professor/Scholar positions at various institutions in Japan, Germany, the U.S., and Singapore. He has authored three books and edited five and authored or coauthored almost 300 papers that have appeared in various journals and conference proceedings. His current research interests are in the fields of intelligent control and mechatronics.

Dr. Kaynak is active in international organizations, has served on many committees of IEEE, and was the President of the IEEE Industrial Electronics Society from 2002 to 2003. He serves as the Coeditor-in-Chief of the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS.