A Multi-Objective Ant Colony System Algorithm for Airline Crew Rostering Problem With Fairness and Satisfaction

Shu-Zi Zhou, *Student Member, IEEE*, Zhi-Hui Zhan, *Senior Member, IEEE*, Zong-Gan Chen, *Student Member, IEEE*, Sam Kwong, *Fellow, IEEE*, and Jun Zhang, *Fellow, IEEE*

Abstract—The airline crew rostering problem (CRP) is significant for balancing the workload of crew and for improving the satisfaction rate of crew's preferences, which is related to the fairness and satisfaction of crew. However, most existing work considers only one objective on fairness or satisfaction. In this study, we propose a new practical model for CRP that takes both fairness and satisfaction into account simultaneously. To solve the multi-objective CRP efficiently, we develop an ant colony system (ACS) algorithm based on the multiple populations for multiple objectives (MPMO) framework, termed multi-objective ACS (MOACS). The main contributions of MOACS lie in three aspects. Firstly, two ant colonies are utilized to optimize fairness and satisfaction objectives, respectively. Secondly, a new hybrid complementary heuristic strategy with three kinds of heuristic information schemes is proposed to avoid ant colonies focusing only on their own objectives. Ant colonies randomly choose one of the three schemes to help explore the Pareto front (PF) sufficiently. Thirdly, a local search strategy with two types of local search respectively for fairness and satisfaction is designed to further approach the global PF. The MOACS is applied to seven real-world monthly CRPs with different sizes from a major North-American airline. Experimental results show that MOACS generally outperforms the greedy algorithm and some other popular multi-objective optimization algorithms, especially on large-scale instances.

Index Terms—Crew rostering problem (CRP), multi-objective optimization, multiple populations for multiple objectives (MPMO), ant colony system (ACS).

I. INTRODUCTION

AIRLINE crew scheduling is of great significance to airlines, which mainly consists of two stages: crew pairing and crew rostering. In the crew pairing problem (CPP), several flights are organized into a pairing that starts from one base and eventually returns to the same base. Then the crew rostering problem (CRP) aims at assigning each pairing to the crew under a set of constraints. The crew scheduling problem is divided into these two stages mainly for two reasons [1]. For one thing, the integrated problem is too large to be handled simultaneously for large-scale instances, and for another, the objectives and constraints of CPP and CRP are different. Some surveys [2], [3] have introduced the related topic. In this study, we concentrate on the CRP due to its direct impact on the fair assignment of workload and the improvement of crew satisfaction.

To better deal with the CRP, the problem model should be firstly formulated and then the methods are designed to solve the model. For the problem model of CRP, there are different kinds of considerations. Many CRP models focus on minimizing the cost from the perspective of airlines. Zeghal and Minoux [4] considered minimizing the total number of additional flight credits for all crew members, and they also took the cost of overnights into account. Santosa *et al*. [5] regarded the variable cost of roster paid by airlines as one of the objective functions, which was represented by actual flying hours. Ezzinbi *et al*. [6] considered minimizing the total cost as the main objective of the crew scheduling problem, and the cost was calculated by the sum of different crew pairing costs.

Different from most existing models that consider the cost from the perspective of airlines, we focus on fairness and satisfaction from the perspective of crew members, which directly affects their motivation. Herein, fairness refers to the balanced allocation of pairings among all the crew members, and many studies have studied the evaluation metrics of the fairness. Doi *et al*. [7] adopted the sum of deviations from

standard working hours for each crew member as a measure of fairness, and the deviation is calculated by the linear penalty. De Armas *et al.* [8] also used the working time as the measure of work-balancing, and they proposed three criteria to describe the deviation as well, which were the standard deviation, interquartile range, and maximum-minimum range. Lučić and Teodorović [9] measured fairness using the weighted sum of average deviations between the real values and their respective ideal numbers, including monthly flight time, allowances, and the number of weekend days. Boufaied *et al.* [10] proposed balancing three objectives, i.e., the number of flight hours, layovers, and destinations' occurrences.

As for satisfaction, some researchers proposed considering crew preferences. In the modern work environment, taking employee preferences into account in workload scheduling has an important effect on service quality [11]. Gamache *et al.* [12] considered a series of weighted bids that reflected the crew preferences and the assignment had to be completed under strict qualifications. Dawid *et al.* [13] suggested generating a set of work lines for each crew member based on their requests like days off and preferred flights, and only one of them can be selected. Maenhout and Vanhoucke [14] considered that the crew could convey their preferences for certain attributes of rosters, such as specific pairings, reserve duties, and even the time when they wanted to be scheduled. Kasirzadeh *et al.* [15] proposed using constraint functions to limit the minimum number of preferred flights and preferred vacations to be satisfied.

However, there are still some defects in the above works. Firstly, some works only optimized a single objective on fairness or satisfaction. In this case, the solution may be good at one objective but extremely poor at the other one, which may lack comprehensiveness and cannot meet the actual needs of the industry [16]. Secondly, many studies only used flight time as the evaluation metric of fairness, which was not comprehensive enough. Thirdly, most researches that considered crew preferences only took it as part of the single objective function. This weighted-sum method for multiple objectives is difficult to determine the weights and only one solution can be obtained.

Based on the considerations mentioned above, we propose a new CRP model from the perspective of crew members, which takes both fairness and satisfaction of crew into account simultaneously. The multi-objective model is solved to get a group of non-dominated solutions so that decision-makers can balance all objectives in different situations. A solution is non-dominated if the other obtained solutions cannot be better than it on both objectives. Specifically, this study uses three attributes to measure fairness comprehensively, including monthly flight time, duty time, and overnight time away from home. Herein, the flight time is the real working time, which is usually regarded as the only measure of fairness in most studies. However, the duty time for the crew to stay in the airport may be also long if the paring is loosely scheduled. Therefore, it is also necessary to consider duty time as another measure of fairness. Moreover, crew members prefer to rest at their own bases rather than in hotels away from home.

Therefore, overnight time away from home (i.e., the rest time in hotels) is also used as a measure of fairness. In terms of satisfaction, our proposed model takes crew satisfaction with the preferences for flights and vacations directly as an objective, rather than as an item in a single objective function.

The formulated multi-objective CRP is an NP-hard problem [7] with many practical constraints, including daily restrictions, weekly restrictions, and monthly restrictions. Different countries or airlines may require different constraints to ensure the adequate rest of crew members. The constraints in our model are based on the latest government document [17] from the ministry of transport in China. These constraints involve the limits on the connection between pairings, accumulative flight time for one calendar month, duty time for seven consecutive days and one calendar month, and an additional requirement for the rest time. These complex constraints make it difficult to find feasible solutions in CRP.

Since our proposed CRP model is a challenging discrete combinatorial optimization problem with complex constraints, it is important to choose a suitable optimizer to solve it. Traditional mathematical methods are inappropriate to such an NP-hard problem or cannot obtain the optimal solution in polynomial time. Ant colony optimization (ACO) [18], [19] is a kind of swarm intelligence approach that has been widely used to obtain the optimal or near-optimal solution in discrete combinational optimization problems [20]–[24]. In particular, the ant colony system (ACS) [18] is an efficient and widely used variant of ACO. Hence, we study the ACS-based algorithm for CRP in this paper. The ACS optimizer constructs a solution step by step, i.e., the pairings are assigned in turn. Each pairing is assigned to a crew member satisfying a set of constraints in each step, thus the feasibility of the final solution can be guaranteed. Therefore, the ACS-based algorithm can find feasible solutions easily, even on a large-scale instance. As the original ACS is a single objective optimization algorithm, we combine the ACS optimizer with the multiple populations for multiple objectives (MPMO) framework [25], [26] to handle the multi-objective CRP.

Therefore, based on the MPMO framework, we develop a multi-objective ACS (MOACS) algorithm to solve the multi-objective CRP. MOACS uses two ant colonies to optimize fairness and satisfaction objectives, respectively. As each ant colony only focuses on its own objective, it may reduce population diversity to explore the central part of the Pareto front (PF). Hence, a hybrid complementary heuristic (HCH) strategy with three kinds of heuristic information schemes is put forward to improve the search ability of ant colonies. Ant colonies randomly choose one of the three schemes to help explore the PF sufficiently. Furthermore, we design a local search strategy to further approach the global PF, which contains two types of local search for fairness and satisfaction, respectively. In this study, MOACS is tested on real-world monthly CRP with various sizes from a major North-American airline [15]. The experimental results indicate that MOACS outperforms some representative multi-objective optimization algorithms in solving CRP, especially on large-scale instances.

TABLE I
EXAMPLE OF FLIGHTS INFORMATION

LegNO.	Airport Dep	Date Dep	Hour Dep	Airport Arr	Date Arr	Hour Arr
LEG_01_0	BASE1	01-01	12:00	AIR1	01-01	13:13
LEG_01_1	AIR1	01-01	14:05	BASE2	01-01	15:19
LEG_01_2	AIR3	01-01	17:45	BASE2	01-01	18:27
LEG_01_3	BASE2	01-01	19:24	AIR2	01-01	20:15
LEG_01_4	AIR5	01-01	20:35	BASE2	01-01	21:49
LEG_01_5	BASE2	01-01	23:11	AIR4	01-02	00:22
LEG_01_6	BASE2	01-01	14:21	AIR6	01-01	16:54

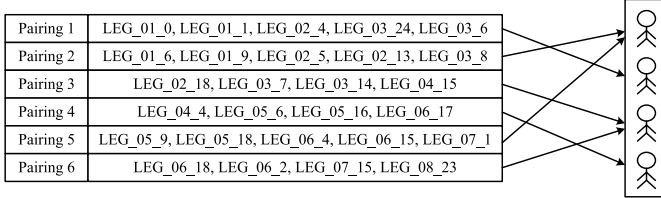| Pairing 1	LEG_01_0, LEG_01_1, LEG_02_4, LEG_03_24, LEG_03_6
Pairing 2	LEG_01_6, LEG_01_9, LEG_02_5, LEG_02_13, LEG_03_8
Pairing 3	LEG_02_18, LEG_03_7, LEG_03_14, LEG_04_15
Pairing 4	LEG_04_4, LEG_05_6, LEG_05_16, LEG_06_17
Pairing 5	LEG_05_9, LEG_05_18, LEG_06_4, LEG_06_15, LEG_07_1
Pairing 6	LEG_06_18, LEG_06_2, LEG_07_15, LEG_08_23

Fig. 1.   Diagram of crew rostering scheme.

To sum up, the contributions of this study are as follows:

1) We propose a new practical multi-objective CRP model from the perspective of crew members that takes both fairness and satisfaction into consideration.

2) An HCH strategy is proposed to develop the MOACS combining the ACS optimizer and the MPMO framework, so as to efficiently solve the multi-objective CRP.

3) A local search strategy is designed to approach the global PF, which contains two types of local search for fairness and satisfaction, respectively.

The rest of this paper is arranged as follows. The definition and formulation of the multi-objective CRP are described in Section II. Section III introduces the details of MOACS and its innovation. Section IV presents the experimental results. Finally, the conclusion of this study is given in Section V.

II. PROBLEM DEFINITION AND FORMULATION

A. Problem Description

A base is a large airport where crew members are stationed. For convenience, the crew usually lives in the city where the base is located. Hence, each crew member is related to one base. A flight is an activity in which an aircraft travels directly from one airport to another, generally in a short period of time. A small portion of the flight schedule of one day is listed in Table I. A pairing is a connection of several flights. It starts from the base corresponding to crew members and finally returns to the base, which lasts usually 3–5 days. The CPP organizes the flights into pairings, while the CRP in our study takes pairings already obtained from the CPP as the input and assigns them to specific crew members. An example with six pairings is shown in Fig. 1, where each pairing has included several flights. The CRP aims at assigning all pairings on the left side to the crew members on the right side.

In our proposed CRP model, $(b_j, s_j, e_j, f_j, d_j, o_j)$ represents the attributes of pairing p_j, where j represents the pairing index. Three of these attributes require preprocessing
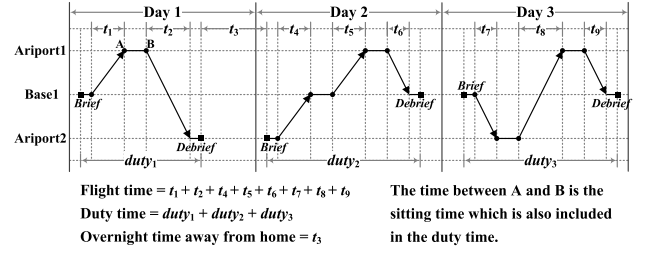

Flight time = $t_1 + t_2 + t_4 + t_5 + t_6 + t_7 + t_8 + t_9$
Duty time = $duty_1 + duty_2 + duty_3$
Overnight time away from home = t_3

The time between A and B is the sitting time which is also included in the duty time.

Fig. 2.   Schematic diagram of attribute calculation for the pairing.

to obtain, which are the $f$light time f_j, $d$uty time d_j, and $o$vernight time away from home o_j. These three attributes are used comprehensively to measure the fairness of scheduling. Combined with the flight information of pairings in Fig. 1 and the corresponding flight in Table I, the required attribute values can be calculated. The flight time f_j refers to the sum of the actual flying time of all flights included in the pairing p_j. The duty time d_j usually consists of three parts. The first part is all the flight time f_j. The second part is the sum of sitting time, i.e., the sum of intervals between flights that is less than the minimum rest time *MinRestTime*. The third part is the preparation time *Brief* before the start of the daily mission and the summary time *Debrief* after the end of the daily mission. We also consider the special cases where the duty time includes an additional fourth part. That is, in some pairings, a flight that a crew member takes only for relocation, which is called a deadhead, may be required. In such a deadhead situation, the crew member flies as a passenger and does not perform any mission, so the deadhead time is not included in the flight time but is included in the duty time because the crew member is still regarded to be on duty during a deadhead. As for the overnight time away from home o_j, it is the layover in hotels after the end of the daily mission, that is, the interval between the daily last flight and the first flight in the next day is larger than or equal to *MinRestTime*, and the arrival airport is not the home base of the crew member. The diagram of the attribute calculation is shown in Fig. 2, where the duty denotes the schedule for one day. Note that the crew member will locate in the corresponding base after performing $duty_2$, so the interval between $duty_2$ and $duty_3$ is not included in the overnight time away from home.

In addition to the three attributes mentioned above, the other three attributes are the departure base b_j, the start time s_j, and the end time e_j of pairing p_j, respectively. s_j is equal to the departure time of the first flight included in pairing p_j minus the pre-flight preparation time *Brief*. Similarly, e_j is equal to the sum of the arriving time of the last flight included in pairing p_j and the summary time *Debrief*.

Moreover, each crew member can apply for their preferred flights and vacations in advance. Note that the preferred flights of the crew member are associated with their base. There are two ways to improve the satisfaction of crew. One is assigning the pairing that includes preferred flights to the corresponding crew member. The other is avoiding assignments which are during the preferred vacations of crew.

In our CRP model, $(cb_i, pf_i, pv_i, cf_i, cd_i, co_i)$ represents the attributes of the crew member c_i. The first three attributes

cb_i, pf_i, and pv_i denote the *c*rew *b*ase, *p*referred *f*lights, and *p*referred *v*acation of crew member c_i, respectively. The remaining three attributes cf_i, cd_i, and co_i respectively refer to the *c*umulative *f*light time, *d*uty time, and *o*vernight time away from home of crew member c_i during a month. Note that we only consider the assignment of pilots, and assume that only one pilot is required per flight in our model.

To make our paper clearer and easier to read, the basic notations and terminologies for the CRP model mentioned above and used following are listed in Table S.I of the supplementary material.

B. Objective Formulation

Different from most existing CRP models that consider the cost from the perspective of airlines, we propose the model for the first time taking both fairness and satisfaction objectives into account from the perspective of crew members. This consideration can greatly enhance the enthusiasm and satisfaction of crew members, thereby increasing the quality of flights.

The fairness objective refers to the balanced allocation of workload among all the crew members. As the standard deviation is the most popular indicator to measure the balance of different members, it is used in our paper to measure the fairness objective. Herein, the workload of a crew member includes three attributes, i.e., monthly flight time, duty time, and overnight time away from home. Note that the fairness of crew in each base should be calculated separately because crew members in different bases cannot perform the same pairing. Therefore, the fairness objective is formulated as the sum of standard deviation of the three workload attributes of the crew members in all the bases, as

$$g_1(X) = \sum_{b=1}^{BN} \left(\sqrt{\frac{1}{|E_b|} \sum_{i \in E_b} dev_{b,i}} \right) \quad (1)$$

where

$$dev_{b,i} = \left(cf_i - \overline{f}_b\right)^2 + \left(cd_i - \overline{d}_b\right)^2 + \left(co_i - \overline{o}_b\right)^2 \quad (2)$$

$$cf_i = \sum_{j=1}^{n_i} \hat{f}_{i,j} \quad (3)$$

$$\overline{f}_b = \frac{1}{|E_b|} \sum_{i \in E_b} cf_i \quad (4)$$

X is the schedule of all the crew, BN denotes the number of bases, and E_b represents the set of all crew members in base b. cd_i and co_i represent the monthly duty time and overnight time away from home of member c_i, respectively, and they are calculated in a similar manner to cf_i as shown in (3). $\hat{f}_{i,j}$ denotes the flight time of the jth pairing in the schedule of crew member c_i. n_i is the total number of pairings to be performed by c_i. $\overline{d}_b$ and $\overline{o}_b$ represent the average duty time and overnight time away from home of all pairings in base b, respectively, and they are calculated in a similar manner to $\overline{f}_b$ as shown in (4). A smaller g_1 indicates a more balanced allocation of workload among all the crew members.

In terms of the satisfaction objective, a crew member is satisfied with the pairing assignment if their preferred flights are included and the assignment does not conflict with their preferred vacation. Therefore, these two attributes are considered in the satisfaction objective. In detail, the percentages of the preferred flights and preferred vacations of all crew members that are satisfied are calculated separately and the sum of these two percentages is used to represent the satisfaction of the pairing assignment for the crew members. Thus, the satisfaction objective is formulated as

$$g_2(X) = \frac{spfn}{PFN} + \frac{spvn}{PVN} \quad (5)$$

where *spfn* denotes the *n*umber of *s*atisfied *p*referred *f*lights and *PFN* denotes the total number of preferred flights required from all the crew. Note that some flights may be preferred by more than one crew member. In such cases, each of these preferred flights is counted only once in the *PFN*. Similarly, *spvn* represents the *n*umber of *s*atisfied *p*referred *v*acations and *PVN* represents the total number of preferred vacations required from all the crew. A larger g_2 indicates a more satisfying pairing assignment scheme.

In our proposed CRP model, both objectives are optimized at the same time. Note that $g_1(X)$ is a minimal objective, whereas $g_2(X)$ is a maximal objective. The goal of our model is defined as (6), which is a bi-objective problem.

$$\text{optimize } G(X) = \begin{cases} \min & g_1(X) \\ \max & g_2(X) \end{cases} \quad (6)$$

C. Constraint Formulation

With regard to constraints, different countries or airlines may have different regulations. Our constraints here are based on the latest government document CCAR121-R5 [17] in China. First of all, the connection between pairings should meet the rationality of time and space. Specifically, the rest time between the connected pairings cannot be less than 10 hours, and the pairing should be assigned to the crew member of the corresponding base. Therefore, the constraints of the reasonable connection between linked pairings in time and space are defined as (7) and (8), respectively, where $\hat{s}_{i,j}$, $\hat{e}_{i,j}$, and $\hat{b}_{i,j}$ denote the start time, the end time, and the departure base of the jth pairing in the schedule of crew member c_i, respectively.

$$\hat{s}_{i,j+1} - \hat{e}_{i,j} \geq 10, \quad \forall i \in \{1, 2, \ldots, CN\} \; \forall j \in \{1, 2, \ldots, n_i - 1\} \quad (7)$$

$$\hat{b}_{i,j} = cb_i, \quad \forall i \in \{1, 2, \ldots, CN\} \; \forall j \in \{1, 2, \ldots, n_i\} \quad (8)$$

Second, there are restrictions on cumulative flight time. In any calendar month, the flight time of a crew member cannot exceed 100 hours, which is formulated as

$$\sum_{d=1}^{31} cfd_{i,d} \leq 100, \quad \forall i \in \{1, 2, \ldots, CN\} \quad (9)$$

where $cfd_{i,d}$ refers to the flight time of the crew member c_i on the dth day.

Third, there are limits on the duty time for seven consecutive days and one calendar month. During any seven consecutive calendar days, the accumulative duty time cannot exceed 60 hours. In any calendar month, the cumulative duty time cannot exceed 210 hours. Therefore, the constraints of duty time within two different time ranges are defined as (10) and (11), respectively, where $cdd_{i,d}$ refers to the duty time of the crew member c_i on the dth day. The range of k in (10) corresponds to all seven consecutive days in a month.

$$\sum_{d=1+k}^{7+k} cdd_{i,d} \leq 60, \quad \forall i \in \{1, 2, \ldots, CN\} \; \forall k \in \{0, 1, \ldots, 24\} \tag{10}$$

$$\sum_{d=1}^{31} cdd_{i,d} \leq 210, \quad \forall i \in \{1, 2, \ldots, CN\} \tag{11}$$

Finally, there is an additional requirement for the rest time. During the 144-hour period prior to a mission, every crew member should be arranged at least 48 consecutive hours for rest. The constraints of additional requirements are formulated as

$$\sum_{j=2}^{n_i} checkRest_{i,j} = 0, \quad \forall i \in \{1, 2, \ldots, CN\} \tag{12}$$

where

$$checkRest_{i,j} = \begin{cases} 1, & \text{if } \sum_{k=0}^{j-1} rest_{i,j,k} = 0 \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

$$rest_{i,j,k} = \begin{cases} 1, & \text{if } \min\left(\hat{s}_{i,k+1} - \hat{e}_{i,k}, \hat{s}_{i,k+1} \right. \\ & \left. - (\hat{s}_{i,j} - 144)\right) \geq 48 \\ 0, & \text{otherwise} \end{cases} \tag{14}$$

$checkRest_{i,j} = 0$ indicates that the jth pairing of member c_i meets the additional requirements. $rest_{i,j,k}$ is used to check all the rest time within 144 hours before the start of the jth pairing of member c_i. $rest_{i,j,k} = 0$ indicates that during the 144 hours before the start of the jth pairing, the rest time between the kth and the $(k+1)$th pairing of member c_i does not exceed 48 hours. A set of constraints discussed above ensures the feasibility of scheduling and sufficient rest for crew members.

III. MOACS Algorithm for Solving CRP

Based on the MPMO framework [25], the MOACS algorithm adopts ACS [18] as the optimizer for multiple ant colonies, and each colony only optimize one objective. To adapt the characteristic of MOACS, we design suitable heuristic information for different objectives and construct solutions according to the pairings start timeline. In addition, a new HCH strategy is proposed to explore a more complete PF. Lastly, we propose a local search strategy with two types of local search to further approach the PF. The entire MOACS algorithm is described below.
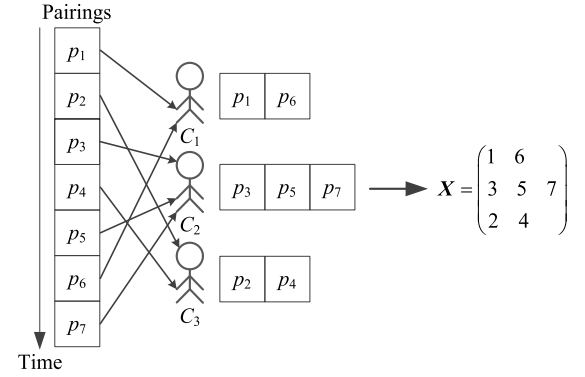

Fig. 3. Schematic diagram of the solution encoding.

A. Solution Encoding

$$X = \begin{pmatrix} x_{1,1} & x_{1,2} & x_{1,3} & \cdots & x_{1,n_1} \\ x_{2,1} & x_{2,2} & x_{2,3} & \cdots & x_{2,n_2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{CN,1} & x_{CN,2} & x_{CN,3} & \cdots & x_{CN,n_{CN}} \end{pmatrix} \tag{15}$$

In ACS, fixed-length binary encoding [27] and integer encoding [28] are commonly used. Specifically, binary encoding in CRP uses the indicator variable $y_{i,j}$ to indicate whether the crew member c_i performs the pairing p_j. The integer encoding in CRP is an integer vector $(z_1, z_2, \ldots, z_{PN})$, where z_j represents the employee index of the crew member performing the pairing p_j. Since these two encodings are fixed-length, they are easy to implement and facilitate some operators like crossover and mutation in the genetic algorithm [29]. However, the constraints involve the crew members' adjacent pairings, and all objectives and constraints are related to crew members. Therefore, it is necessary to decode the solution into a natural expression such as (15) for fitness calculation. In (15), $x_{i,j}$ denotes the pairing index of the jth pairing in the schedule of the crew member c_i whose employee index is i, CN denotes the number of crew members, and n_i represents the total number of pairings to be performed by member c_i. Each row vector represents all the pairings to be performed by a crew member in chronological order. Since the total number of pairings performed by each crew member is not fixed, the length of the row vectors is not fixed.

Based on the above considerations, we directly adopt the variable-length natural expression shown in (15) as the encoding of solutions. An example of solution encoding is shown in Fig. 3. This encoding scheme is clear and does not require a decoding process. Besides, during the solution construction (details refer to Section III-C), MOACS will handle the pairings one by one and assign only one crew member to each pairing according to the pheromone values and heuristic information, which guarantees that each pairing will be assigned to only one crew member. Moreover, according to the above encoding, it is easy to calculate the three attributes of member c_i, i.e., the sum of the flight time cf_i, duty time cd_i, and overnight time away from home co_i.

B. Multiple Colonies Framework

The key to solving multi-objective optimization problems (MOPs) with evolutionary algorithms is how to choose suitable

solutions for the next generation. Since it is hard to define an appropriate fitness metric in multi-objective functions, the MOACS algorithm borrows the concept of MPMO framework to consider each objective separately. Based on the MPMO framework, multiple ant colonies are adopted, and each ant colony only needs to focus on optimizing a single objective through a conventional optimizer. Herein, the ACS with the same population size m is adopted as the optimizer for each ant colony. As the CRP model is a bi-objective problem, we use two ant colonies named *Fcolony* and *Scolony* that set fairness and satisfaction as their optimization objective, respectively.

C. Solution Construction

Since each crew member's schedule must satisfy the time constraint, it is natural to construct schedules according to the starting timeline of pairings. Thus, we first index all the pairings in this order. The ACS imitates the foraging behavior of ant colonies in finding the shortest path to food. Ants will release chemicals named pheromones on the paths they pass and they will follow the path with higher pheromones, which forms a mechanism similar to the positive feedback. In each step during solution construction, pheromones and heuristic information are used to determine which crew member should be assigned the corresponding pairing p_j. The design of the pheromones and heuristic information will be presented in Sections III-D and III-E, respectively. The solution construction process is shown as

$$i = \begin{cases} \underset{k \in \{1,2,\dots,CN\}}{\arg\max} \left\{ \tau^{col}(k,j) \times \left[\eta^{col}(k,j) \right]^{\beta} \right\}, & \text{if } q \leq q_0 \\ I, & \text{otherwise} \end{cases}$$
$$(16)$$

where β is a parameter to balance the effect of pheromones and heuristic information, *col* is the identifier of the ant colony (i.e., the *Fcolony* or the *Scolony*), and q_0 is the probability of exploitation. $\tau^{col}(k,j)$ and $\eta^{col}(k,j)$ represent the *col* ant colony's pheromone and heuristic information between crew member c_k and pairing p_j, respectively.

For the pairing p_j, we firstly generate a random number q to determine which construction method is adopted. If q does not exceed q_0, the ant will greedily select the crew for exploitation, that is, select the crew member with the largest $\tau^{col}(k,j) \times [\eta^{col}(k,j)]^{\beta}$. Otherwise, the crew member i will be selected through the roulette wheel selection according to the probability defined as

$$p(i,j) = \frac{\tau^{col}(i,j) \times \left[\eta^{col}(i,j) \right]^{\beta}}{\sum_{k=1}^{CN} \tau^{col}(k,j) \times \left[\eta^{col}(k,j) \right]^{\beta}},$$
$$\forall i \in \{1, 2, \dots, CN\} \quad (17)$$

The heuristic information is not as fixed as traditional problems because the workload balance is constantly changing during the construction. Hence, the heuristic information should be calculated in each step during the construction. Generally speaking, the number of crew members (i.e., CN) is always fewer than the number of pairings (i.e., PN), that is, a crew member should be assigned several pairings in

a month. Therefore, in this case, we only need to calculate the heuristic information corresponding to all crew members, instead of all pairings. In contrast, the crew-based 'pilot-by-pilot' solution construction method [30] firstly completes the assignment for the first crew member, then the second, until all pairings are assigned, whose time complexity is $O(PN^2)$. However, the time complexity of our pairings-based solution construction method is $O(PN \times CN)$, which will reduce the computational complexity. Besides, we need to check all available crew members that can satisfy a set of constraints in Section II-C when they perform the current pairing.

D. Pheromone Update

1) Pheromone Initialization: In MOACS, the initial pheromone values τ_0 of *Fcolony* and *Scolony* are defined as (18) and (19), respectively, where PN is the total number of pairings, FGS is the greedy solution about fairness, and SGS is the greedy solution about satisfaction.

$$\tau_0^{Fcolony} = [PN \times g_1(FGS)]^{-1} \quad (18)$$
$$\tau_0^{Scolony} = \{PN \times [2 - g_2(SGS)]\}^{-1} \quad (19)$$

Note that different colonies perform the solution construction process independently according to their different objectives and have their own pheromone values under the MPMO framework. This is an advantage so that different scales of the objective values and pheromone values in two colonies do not lead to objective bias. Herein, objective bias means the algorithm tends to optimize more on one of the objectives.

The selection of the crew member for the pairing p_j in *FGS* is defined as

$$FGS[j] = \underset{i \in \{1,2,\dots,CN\}}{\arg\min} \left(\left| cf_{i,j} - \overline{cf_{i,j}} \right| + \left| cd_{i,j} - \overline{cd_{i,j}} \right| + \left| co_{i,j} - \overline{co_{i,j}} \right| \right) \quad (20)$$

where $cf_{i,j}$, $cd_{i,j}$, and $co_{i,j}$ denote the cumulative flight time, duty time, and overnight time away from home of member c_i, respectively, when pairings from p_1 to p_j have been assigned. $\overline{cf_{i,j}}$, $\overline{cd_{i,j}}$, and $\overline{co_{i,j}}$ refer to the average flight time, duty time, and overnight time away from home of the first j pairings, respectively. Note that the calculations of these three average values are limited to the base of the crew member c_i. According to (20), the pairing p_j is always assigned to the member with the smallest sum of absolute differences between its three attribute values and the corresponding average values.

As for *SGS*, the selection of the crew member for the pairing p_j is defined as

$$SGS[j] = \underset{i \in \{1,2,\dots,CN\}}{\arg\max} \frac{\overline{fn_{i,j}} + 1}{cv_{i,j} + 1} \quad (21)$$

where

$$\overline{fn_{i,j}} = \begin{cases} \dfrac{fn_{i,j} - fn_{i,j}^{\min}}{fn_{i,j}^{\max} - fn_{i,j}^{\min}}, & \text{if } fn_{i,j}^{\max} > 0 \\ 0, & \text{otherwise} \end{cases} \quad (22)$$

$$cv_{i,j} = \begin{cases} 1, & \text{if pairing } p_j \text{ conflicts with the vacation } pv_i \\ 0, & \text{otherwise} \end{cases}$$
$$(23)$$

$fn_{i,j}$ is the number of preferred flights of the crew member c_i included in the pairing p_j. Since the selection of the crew member for the pairing p_j considers two factors with different scales and $cv_{i,j}$ is an indicator variable, $fn_{i,j}$ needs to be normalized. Min-max feature scaling is used for normalization to eliminate the effect of scales. According to (21), the pairing p_j is always assigned to the crew member with the most preferred flights in the pairing p_j and the least violation of their preferred vacation.

2) Pheromone Local Update: The MOACS algorithm uses ACS as the optimizer, in which all ants construct solutions in parallel. Once an ant has completed a step during the construction (i.e., assigning a pairing to a crew member), it immediately performs the local update of pheromones. It aims to improve the ability of exploration and is described as

$$\tau^{col}(i, j) = (1 - \rho) \times \tau^{col}(i, j) + \rho \times \tau_0^{col} \qquad (24)$$

where ρ is a parameter in the range of (0, 1). Since τ_0^{col} is a small value as defined as (18) and (19), the pheromone value $\tau^{col}(i, j)$ between the crew member c_i and pairing p_j will be decreased after an ant assigns p_j to c_i in the ant colony *col*. Therefore, other ants will be more likely to select the other crew member different from c_i for p_j. The local update of pheromones helps improve the diversity of solutions and avoid premature convergence.

3) Pheromone Global Update: The conventional ACS for single-objective problems updates global pheromones directly through the best solution found so far. However, a set of non-dominated solutions are obtained in MOPs, and we cannot determine which one is the best solution. A different method needs to be designed to update global pheromones for MOPs. There have been some techniques for globally updating pheromones in previous studies, for example, Zhan *et al.* [25] proposed selecting a random solution from non-dominated solutions to globally update pheromones. Barán and Schaerer [31] proposed updating all the non-dominated solutions. Doerner *et al.* [32] proposed updating the best solution of respective objectives. Updating a random solution from non-dominated solutions will improve the solution diversity, but it lacks efficient search guidance. In contrast, updating all the non-dominated solutions or the best solution of respective objectives will guide ants to search better. However, the former technique may lead to premature convergence and the latter may get trapped in a narrow region of PF.

Therefore, the global update strategy [33] is adopted due to its ability to balance search efficiency and solution diversity, as described below. Firstly, we introduce a set *Archive* to save all non-dominated solutions from two ant colonies in all generations, and the global update solutions (*GUSs*) will be selected from the *Archive*. Then, we sort all solutions in the *Archive* in ascending order by the objective value g_1. Finally, *Fcolony* randomly selects the $GUS^{Fcolony}$ from the first $|Archive| \times \theta$ solutions with smaller g_1, whereas *Scolony* randomly selects its $GUS^{Scolony}$ from the last $|Archive| \times \theta$ solutions with better g_2. θ is a parameter in the range of (0, 1), which controls the balance between search efficiency and solution diversity. A small θ accelerates convergence, while a large θ improves the global search ability. The *GUS* of

an ant colony performs the local search strategy described in Section III-F below, which generates a new neighbor solution GUS_{new}. If GUS_{new} and the original GUS are non-dominated to each other, both are allowed to globally update. Otherwise, only the GUS_{new} is required in the global update. Generally, the local search strategy can improve GUS on one objective, so that GUS_{new} is not dominated by the original GUS. The global update of pheromones by $GUSs$ is shown as

$$\tau^{col}(i, j) = (1 - \varepsilon) \times \tau^{col}(i, j) + \varepsilon \times \Delta\tau(i, j) \qquad (25)$$

$$\Delta\tau(i, j) = \begin{cases} g_1(GUS^{Fcolony})^{-1}, & \text{if } (i, j) \in GUS^{Fcolony} \\ [2 - g_2(GUS^{Scolony})]^{-1}, & \text{if } (i, j) \in GUS^{Scolony} \\ 0, & \text{otherwise} \end{cases}$$

$$\qquad (26)$$

where ε is a pheromone decay parameter in the range of (0, 1). After the execution of the global update, the promising solutions $GUSs$ are allowed to leave more pheromones on their paths for increasing their attractiveness. Conversely, the pheromones on the paths of the other solutions will evaporate according to the local update to reduce their desirability.

E. Hybrid Complementary Heuristic Strategy

Heuristic information is equivalent to the self-experience of ants, which can be used to better guide the search. We first introduce how to choose reasonable heuristic information for the two ant colonies with different optimization objectives. Note that only crew members who meet a set of constraints need to calculate heuristic information, and the heuristic information of remaining members is directly set to zero. *Fcolony* aims to optimize the fairness objective. Thus, its heuristic information is devised to reduce the sum of deviations of the three attribute values, which are the flight time, duty time, and overnight time away from home. Since the solution construction restricts the heuristic information to be a maximum objective, the heuristic information uses the inverse of the sum of deviations. The smaller the sum of absolute difference between the three attribute values and their corresponding average values, the greater the heuristic information of the crew member. The definition of heuristic information about fairness objective is shown in (27), whose variables have been given in Section III-D. Notice that since solutions constructed by each ant are different, the corresponding heuristic information may be different for different ants, even if the pairing p_j is assigned to the same crew member. Therefore, it is necessary to dynamically calculate heuristic information in each step during each ant's solution construction.

$$\eta_1(i, j) = \left(\left| cf_{i,j} - \overline{cf_{i,j}} \right| + \left| cd_{i,j} - \overline{cd_{i,j}} \right| + \left| co_{i,j} - \overline{co_{i,j}} \right| \right)^{-1}$$

$$\qquad (27)$$

In addition, *Scolony* aims to obtain the schedule containing as many preferred flights and vacations as possible for the crew. Hence, its design of heuristic information involves the number of crew members' preferred flights included in the current pairing and whether the current pairing conflicts with their preferred vacations. The more a crew member's preferred flights are included in the current pairing and the pairing

does not conflict with their preferred vacation, the greater the heuristic information of the crew member. The definition of heuristic information about the satisfaction objective is shown in (28). The variables in (28) have been given in Section III-D.

$$\eta_2(i, j) = \frac{\overline{fn_{i,j}} + 1}{cv_{i,j} + 1} \quad (28)$$

Since each ant colony only focuses on optimizing a single objective optimization problem in the MPMO framework, it may lead to solutions concentrating on the margin of PF. There are two methods to enhance the exploration of the central part of PF. Chen *et al.* [33] proposed the ordinary complementary heuristic strategy that utilized the heuristic information of the external objective. The other method is to use aggregated heuristic information combining all objectives [34]. Based on the second method, we design the suitable aggregated heuristic information defined in (29), which includes two terms about fairness and satisfaction objectives, respectively. The first term $\eta_1(i, j)$ is in the range of $(0, \infty)$ and the second term ($\overline{fn_{i,j}} - cv_{i,j} + 1$) is in the range of [0, 2]. Since the two terms are in different scales, we combine them by multiplication to reduce the impact of scales.

$$\eta_3(i, j) = \eta_1(i, j) \times (\overline{fn_{i,j}} - cv_{i,j} + 1) \quad (29)$$

In order to fully explore the PF, we propose a hybrid strategy in which an ant colony randomly utilizes the heuristic information of its own objective, the other objective, or the aggregated heuristic information. These are the three heuristic information schemes in our proposed HCH strategy. The HCH strategy is defined in (30) so that the three schemes will be adopted with the same probability. In detail, we firstly generate a random number r with uniform distribution to determine the heuristic scheme. If r is less than 1/3, which corresponds to the heuristic information of their own objectives, *Fcolony* uses η_1 and *Scolony* uses η_2 as their heuristic information. If r is greater than 1/3 and less than 2/3, which corresponds to the heuristic information of the external objective, *Fcolony* uses η_2 and *Scolony* uses η_1 as their heuristic information. Otherwise, the aggregated heuristic strategy is selected, and both *Fcolony* and *Scolony* adopt η_3 as their heuristic information. The HCH strategy with three schemes can balance the search of the whole PF. In detail, the first scheme helps to explore the margin of the PF, while the other two schemes can explore the central part of the PF. Note that the random number r is generated per generation, and therefore the heuristic information scheme used in different generations may be different.

$$\left(\eta^{Fcolony}(i, j), \eta^{Scolony}(i, j)\right)$$
$$= \begin{cases} (\eta_1(i, j), \eta_2(i, j)), & \text{if } r < 1/3 \\ (\eta_2(i, j), \eta_1(i, j)), & \text{if } 1/3 \leq r < 2/3 \\ (\eta_3(i, j), \eta_3(i, j)), & \text{otherwise} \end{cases} \quad (30)$$

F. Local Search Strategy

To enhance the accuracy of solutions generated by MOACS, we design two types of local search for the two objectives respectively. On the one hand, the local search for improving

the fairness adjusts the pairings performed by two crew members with the largest deviation. On the other hand, the local search for improving crew satisfaction adjusts crew members who perform pairings during their preferred vacations. In order to approach the global PF, both *Fcolony* and *Scolony* randomly choose one of these two local searches to execute. Taking *Fcolony* as an example, if it executes the local search designed for fairness, its g_1 value is likely to decrease and the margin of PF can be explored. In the other case, if it executes the local search designed for satisfaction, its g_2 value may increase and the central part of PF can be explored. In addition to the HCH strategy mentioned above, this is another way to overcome the shortcoming that each ant colony only focuses on its own objective.

The local search strategy is executed in each ant colony in every generation after the selection of *GUS*. As only the *GUS* of each ant colony executes the local search strategy, it does not take too much time. Note that if all preferred vacations in *GUS* have been satisfied, the solution will directly perform the local search designed for fairness. Otherwise, the *GUS* will randomly select one of the two types of local search. That is, a random number r_1 within the range of [0, 1] is randomly generated. If r_1 is smaller than 0.5, the ant colony performs the local search designed for fairness. Otherwise, the other local search designed for satisfaction will be performed. These two types of local search are described in detail below. Firstly, the local search designed for fairness is described below, and its pseudocode is shown in **Algorithm 1**.

Algorithm 1 Local Search for Fairness

Input: *GUS*
Output: the newly generated solution GUS_{new}
1 **for** $b = 1$: *BN* **do**
　　/* Calculate the sum of the absolute difference *absd*
　　for all crew members in base b. 　　　　*/
2 　$maxabsd = 0$;
3 　**foreach** $i \in E_b$ **do**
4 　　$absd_i = |cf_i - \bar{f}_b| + |cd_i - \bar{d}_b| + |co_i - \bar{o}_b|$;
5 　　**if** $absd_i > maxabsd$ **then**
6 　　　$C_1 = i$;
7 　　　$maxabsd = absd_i$;
　　/* Find the matching member for C_1. 　　　　*/
8 　$mindist = +\infty$;
9 　**foreach** $i \in E_b$ **do**
10 　　$dist = |(cf_{C_1} + cf_i)/2 - \bar{f}_b| +$
　　　$|(cd_{C_1} + cd_i)/2 - \bar{d}_b| + |(co_{C1} + co_i)/2 - \bar{o}_b|$;
11 　　**if** $dist < mindist$ **then**
12 　　　$C_2 = i$;
13 　　　$mindist = dist$;
14 　TransferPairings(C_1, C_2);
15 　ExchangePairings(C_1, C_2);
16 Evaluate the newly generated neighbor solution GUS_{new};

We first calculate the sum of the *ab*solute *d*ifference *absd* between the three attribute values and their corresponding
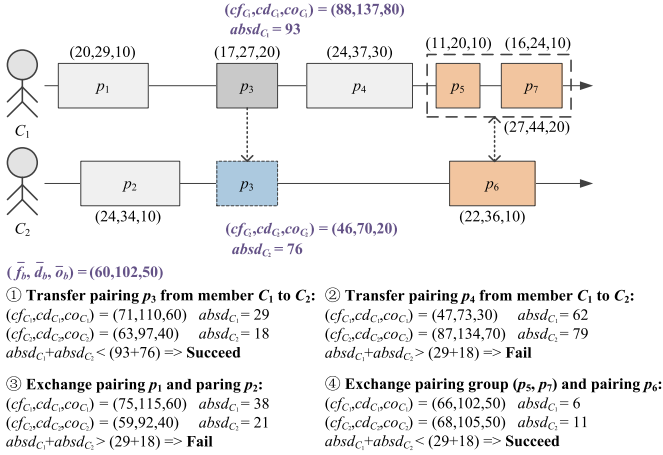
Fig. 4.   Schematic diagram of the local search for fairness.

Algorithm 2 Local Search for Satisfaction

Input: *GUS*

Output: the newly generated solution GUS_{new}

1 **for** $b = 1$: *BN* **do**

2   Randomly select a member C from all crew members in base b whose preferred vacation is not satisfied;

3   *ProcessingPool* $= \emptyset$;

4   **foreach** pairing p in the schedule of member C **do**

5   **if** pairing p conflicts with the preferred vacation of member C **then**

6   *ProcessingPool* $=$ *ProcessingPool* $\cup$ p;

7   **foreach** pairing $p \in ProcessingPool$ **do**

8   *IdleMembersPool* $= \emptyset$;

9   **foreach** member *mem* in base b **do**

10   **if** member *mem* can perform pairing p and pairing p does not conflict with his preferred vacation **then**

11   *IdleMembersPool* $=$ *IdleMembersPool* $\cup$ *mem*;

12   **if** *IdleMembersPool* $\neq \emptyset$ **then**

13   Select the idle crew member *idlemem* with the most preferred flights in pairing p;

14   TransferSinglePairing(C, *idlemem*, p);

15 Evaluate the newly generated neighbor solution GUS_{new};

---

average values for all crew members, as shown in Line 4. The crew member C_1 with the largest *absd* in base b is selected to conduct the local search. The above steps are carried out from Line 2 to Line 7. Then, the matching member C_2 for C_1 can be found from Line 8 to Line 13. The criterion for finding the matching member C_2 is that the average attribute values of the two members C_1 and C_2 are the closest to the average attribute values of all the members in the entire base. Next, we try to transfer the pairings directly as shown in Line 14. The pairings in the crew member's schedule with a large sum of three attribute values between members C_1 and C_2 are transferred to the other member's schedule. After that, we try to exchange the pairings with overlapping time in the schedules of members C_1 and C_2, as shown in Line 15. Note that the transfer and exchange of pairings require to satisfy a set of constraints, and if the sum of *absd* of the two members does not decrease after the transfer and exchange, the original schedules will be restored. After the crew members in all bases have been adjusted, the newly generated neighbor solution will be evaluated at last, as shown in Line 16.

The transfer and exchange of pairings in Line 14 and Line 15 of **Algorithm 1** are illustrated with an example in Fig. 4. The three numbers (f_j, d_j, o_j) around the pairing p_j represent the flight time, the duty time, and the overnight time away from home of pairing p_j, which is a part of its attributes. The wired arrows represent the timeline. The sum of *absd* of the two members is 169 before the local search, and it turns to 47 after the transfer of pairing p_3, so the transfer succeeds. Similarly, the sum of *absd* increases to 141 when pairing p_4 is transferred from member C_1 to member C_2. Since it is larger than 47, the transfer is not performed. Next, we try to exchange pairings that overlap in time. In this example, pairings p_1 and p_2 overlap, and the pairing group including pairing p_5 and p_7 overlaps with pairing p_6. After the exchange for pairing p_1 and p_2, the sum of *absd* slightly increases to 59 which is also larger than 47, thus the exchange is still not executed. When the pairing group and pairing p_6 exchange, the sum of *absd* decreases to 17, so this exchange is performed successfully. The transfer and exchange of pairings make the

three attribute values of member C_1 and C_2 close to the actual average of their base b.

The other local search designed for satisfaction is described below, and its pseudocode is shown in **Algorithm 2**. We first randomly select a crew member C from all crew members in base b whose preferred vacation is not satisfied to make the adjustment, as shown in Line 2. Then, we examine the pairings set *ProcessingPool* that conflicts with their preferred vacation from the schedule of crew member C, which is carried out from Line 3 to Line 6. Next, for each pairing p in *ProcessingPool*, we check all idle crew members who can perform this pairing and do not conflict with their preferred vacation. This is performed from Line 7 to Line 11. If there are idle crew members, this pairing p is transferred from the schedule of member C to the schedule of the idle crew member with the most preferred flights in pairing p, as shown from Line 12 to Line 14. After the crew members in all bases have been adjusted, the newly generated neighbor solution will be finally evaluated, as shown in Line 15.

The local search for satisfaction is illustrated with an example in Fig. 5. In the example, the preferred vacation of member C conflicts with pairing p_4 and p_6 in their schedule. Both pairings are stored in *ProcessingPool* temporarily and then check all idle members for each pairing. In this example, member c_3 and c_5 are able to perform pairing p_4, and only member c_2 can perform pairing p_6. Additionally, the number in parentheses on the member circle denotes the number of preferred flights of the corresponding member included in the above pairing. Since member c_5 applies for more preferred
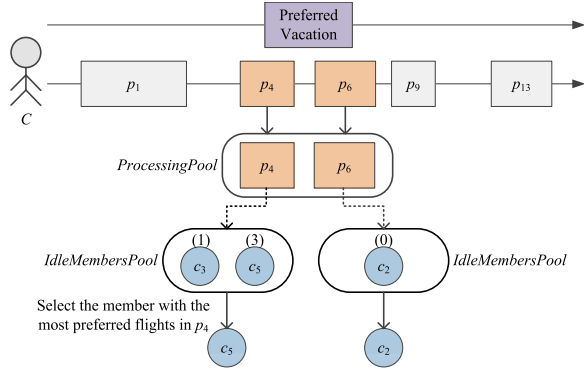
Fig. 5. Schematic diagram of the local search for satisfaction.

flights in pairing p_4, pairing p_4 is assigned to member c_5 at last. Although the preferred flights of member c_2 are not included in pairing p_6, pairing p_6 is still assigned to member c_2 because pairing p_6 has only one idle member. The local search for satisfaction may lead to an increase in the satisfaction rate of preferred flights and vacations.

G. Complete MOACS Algorithm

The overall pseudocode of the MOACS is shown in **Algorithm 3**. Firstly, the greedy solutions *FGS* and *SGS* are generated, which are used to initialize the pheromones by (18) and (19), and the global archive is also initialized. Then in every generation, the heuristic strategy is randomly selected. Two ant colonies calculate their heuristic information according to (30). In each step during the solution construction, a crew member is selected for each pairing by (16) and the pheromones are locally updated according to (24). After all ants have constructed solutions, these newly generated solutions will be evaluated and added into the global archive. All dominated solutions in the archive will be eliminated, and then $GUS^{Fcolony}$ and $GUS^{Scolony}$ are selected from it. If all preferred vacations have been satisfied, the *GUS* will directly perform **Algorithm 1**. Otherwise, it will randomly perform **Algorithm 1** or **Algorithm 2**. The global update of pheromones is conducted according to (25) and (26). If the newly generated GUS_{new} and the original *GUS* are non-dominated to each other, both are used in the global update of pheromones. Otherwise, only the GUS_{new} is used. MOACS terminates when the maximum execution time is reached and outputs all non-dominated solutions in the archive.

IV. EXPERIMENTS AND COMPARISONS

In this section, experimental tests are performed on the real-world dataset to verify the performance of MOACS in solving CRP. All the algorithms are implemented in C++ and run on a PC with a Xeon Quad CPU E3-1225 and 4.0GB of RAM.

A. Test Dataset

We use the experimental instances in [15] to test the performance of all algorithms in solving CRP. This dataset comes from a major North-American airline and provides

---

Algorithm 3 Procedure of MOACS

1 $t = 0$;
2 Generate greedy solutions *FGS* and *SGS*;
3 *Archive* $= \emptyset$;
4 Initialize the pheromone $\tau_0^{Fcolony}$ and $\tau_0^{Scolony}$ according to (18) and (19), respectively;
5 **while** $t \le MaxTime$ **do**
6    $HeurSchNo = $ rand()%3; // 0, 1, 2 represent three different heuristic schemes
7   **for** $col = 1:2$ **do**
8   **for** $j = 1:PN$ **do**
9   **for** $k = 1:m$ **do**
10   Calculate heuristic information of the kth ant in the *col*th colony based on *HeurSchNo* by (30);
11   Select a crew member by (16) and add the pairing p_j to their schedule;
12   Perform the local update of pheromone according to (24);
13   **for** $k = 1:m$ **do**
14   Evaluate the kth ant in the *col*th colony;
15   Add the kth ant in the *col*th colony to *Archive*;
16   Eliminate all dominated solutions in *Archive*;
17   **for** $col = 1:2$ **do**
18   Select the global update solution(GUS) for the *col*th colony;
19   **if** all preferred vacations in *GUS* have been satisfied **then**
20   Perform **Algorithm 1**;
21   **else**
22   Randomly perform **Algorithm 1** or **Algorithm 2**;
23   Perform the global update of pheromone for GUS_{new} according to (25) and (26);
24   **if** GUS_{new} and the original *GUS* are non-dominated to each other **then**
25   Perform the global update of pheromone for the original *GUS* according to (25) and (26);
26   Update the elapsed time t;

Output: All non-dominated solutions in *Archive*

TABLE II

SIZE OF SEVEN TEST INSTANCES

Instance	No. of pairings (*PN*)	No. of pilots (*CN*)
I1-727	172	33
I2-DC9	303	34
I3-D94	274	47
I4-D95	1079	145
I5-757	1497	247
I6-319	1187	223
I7-320	1648	305

seven instances of different sizes as shown in Table II. The last four instances are large. In our proposed CRP model, we consider monthly schedules that have more guiding value in the practical application.

TABLE III

COMPARISONS OF MOACS WITH NON-METAHEURISTIC ALGORITHMS ON TWO OBJECTIVES

Instance	GREEDF		MOACS		GREEDS		MOACS	
	g_1	g_2	g_1	g_2	g_1	g_2	g_1	g_2
I1-727	38.251	0.492	**24.893**	0.733	122.035	1.366	85.500	**1.378**
I2-DC9	49.672	0.275	**22.413**	0.785	156.633	**1.382**	50.270	1.361
I3-D94	47.222	0.125	**28.986**	1.111	165.666	**1.320**	91.160	1.307
I4-D95	34.174	0.221	**31.571**	0.780	162.970	**1.392**	72.534	1.351
I5-757	**30.049**	0.172	31.173	0.925	164.193	**1.506**	65.139	1.446
I6-319	39.867	0.230	**38.471**	0.941	176.877	**1.417**	91.184	1.375
I7-320	**39.963**	0.158	40.304	0.935	161.786	**1.449**	100.155	1.405

We use the pairings from the dataset [15] as our input of the CRP. The pairings are the solutions of the CPP. Since our proposed model further involves preferred flights and vacations, we randomly generate these data. The generator for preferred flights from the dataset recommends that each crew member prefers a certain percentage per_f of flights associated with their base. Instead of randomly selecting preferred flights from all flights, this way is more reasonable in the actual situation, because crew members generally prefer flights close to their bases. In addition, the generator for preferred vacations from the dataset suggests that a certain percentage per_v of all crew members apply for vacation. The vacation periods can occur on any day of the month with lengths between 2 and 10 days with uniform distribution. Their preferred vacations start at 00:00 and end at 23:59 by default. Here we set per_f to 10% and per_v to 30%.

B. Experimental Settings

The parameter configurations in our proposed MOACS algorithm are shown as follows. The number of ants in each ant colony m is set to 60. The CRP-related parameters in Section II are *Brief* = 30 minutes, *Debrief* = 30 minutes, and *MinRestTime* = 9 hours. The configurations of other ACS-related parameters are the same as in [18] (except for the initial pheromone value τ_0), where the parameter $\beta = 2$, $q_0 = 0.9$, $\rho = 0.1$, and $\varepsilon = 0.1$. The setting of τ_0 for *Fcolony* and *Scolony* is given in (18) and (19), respectively. The parameter θ in the global update of pheromone is set to 0.1.

MOACS is compared with three algorithms, greedy heuristic, NSGA-II [29], and P-ACO [35]. NSGA-II is a popular algorithm to solve MOPs. Based on the NSGA-II variant, Chen *et al.* [29] designed crossover and mutation operators for this problem and proposed a special repair operator to improve the feasibility of solutions. The parameters of NSGA-II [29] are set according to the corresponding reference. The population size and offspring size are set to 100 and 80, respectively. The crossover rate is set to 1.0, the mutation rate is set to 0.3, and the parameter related to the population update is set to 0.5. The maximum generations are 2000 in NSGA-II for each instance. Since the execution time of other heuristic algorithms is different from NSGA-II in every generation, all the heuristic algorithms use the same execution time as NSGA-II to ensure the fairness of comparison. P-ACO [35] is a classical multi-objective algorithm based on ACO with multiple pheromone matrices for multiple objectives, but only one heuristic matrix that integrates the information of all objectives. The parameters

of P-ACO [35] are the same as MOACS. Since P-ACO adopts only one heuristic matrix, we design the heuristic information to be the same as (29) in Section III-E for P-ACO. All the heuristic algorithms run independently 10 times.

C. Experimental Results

Although the proposed CRP model is a bi-objective problem, common performance metrics such as inverted generational distance cannot be used because the true PF is not known. Hence, the two performance metrics hypervolume (HV) and C(X, Y) [36] are adopted for performance evaluation. HV represents the volume among a solution set and a reference point. The reference point is selected by the worst objective values about fairness and satisfaction from all solutions obtained by all comparison algorithms in all runs. The larger the HV of the solution set, the better its diversity and convergence efficiency. The HV is calculated independently in each run and the average result of the 10 runs is compared.

In addition, C(X, Y) concerns the dominant relationship between two sets of solutions and is defined as

$$C(X, Y) = \frac{|\{y \in Y \,|\exists x \in X : x \preceq y\}|}{|Y|} \quad (31)$$

where $x \preceq y$ denotes solution x dominates or equal to y. It represents the ratio of the number of solutions in Y dominated by any solution in X and is in the range of [0, 1]. If C(X, Y) equals to one, it means each solution in Y is dominated by or equal to at least one solution in X. If C(X, Y) equals to zero, it represents that no solutions in Y can be dominated by any solution in X. Note that the sum of C(X, Y) and C(Y, X) is generally not equal to one. If C(X, Y) is larger than C(Y, X), it represents the solution set X is better than Y. We collect all non-dominated solutions obtained in 10 runs to calculate the C(X, Y) value.

1) Comparison With Non-Metaheuristic: The greedy heuristic is an efficient non-metaheuristic algorithm to deal with single-objective problems. The greedy algorithms for optimizing fairness and satisfaction are termed GreedF and GreedS, respectively, both of which have been described in Section III-D. The comparisons of MOACS with GreedF and GreedS are presented in Table III, where the best results are in **boldface**. Since MOACS is a multi-objective algorithm, its result is a solution set. Here, we use the best solution for the corresponding single objective in each run and use their median of the 10 runs for comparison. In other words,

the solutions with minimum g_1 and maximum g_2 in the median run, i.e., the margin of the PF, are used to compare with the greedy algorithms, which are shown in the left and right side of Table III, respectively.

As can be seen from Table III, MOACS performs much better than the greedy algorithms on g_1 on most instances. Although the average of MOACS cannot reach the best value on fairness in I5-757 and I7-320, MOACS can still obtain the better g_1 in some runs on these two instances. As for the performance in terms of satisfaction, MOACS can get the best result in I1-727, and obtain the solutions close to GreedS on g_2 on other instances. The reason for the good performance of greedy algorithm in optimizing g_2 is that the selection for a pairing has less impact on the selection for the other pairings. Thus, it tends to obtain a fairly high satisfaction rate. However, the optimization of g_1 is more complex so that the greedy algorithm may easily get trapped in a local optimum. Moreover, since MOACS optimizes both objectives g_1 and g_2 at the same time rather than simply focuses on g_2, it is reasonable that MOACS cannot obtain the best g_2 on some instances. Although MOACS cannot exceed GreedS on g_2, it has a significant advantage on another objective g_1. In a word, MOACS can get the solutions as good as or even better than that of the greedy algorithms on the corresponding objective, while it is far better than greedy algorithms on the other objective.

2) Comparison With Multi-Objective Optimization: The multi-objective algorithms adopted for comparison are the NSGA-II variant [29] and P-ACO [35]. Since the two comparison algorithms do not contain the local search, we use a MOACS variant without the local search for fair comparisons, which is termed MOACS-noLS.

Since the CRP is an NP-hard problem with various constraints, it is difficult to find a feasible solution. Firstly, we focus on the feasible rate (FR) of all comparison algorithms, which is the ratio of the number of feasible runs to the total number of runs. The comparisons of MOACS-noLS with NSGA-II and P-ACO in terms of FR are shown in Table IV, where the best results are in **boldface**. We can find that ACO-based algorithms have a significant advantage in finding feasible solutions. Both P-ACO and MOACS-noLS are able to obtain feasible solutions in all 10 runs on all instances because a set of constraints is checked and satisfied in each step during the solution construction. In contrast, NSGA-II only has a 50% probability of finding a feasible solution in I2-DC9 and even can hardly obtain a feasible solution on the last four large-scale instances. As the solutions obtained by NSGA-II are randomly generated at the beginning, it is hard to satisfy a set of constraints in such high dimensional search space.

Table IV also compares MOACS-noLS with NSGA-II and P-ACO in terms of HV, where the best results are in **boldface**. Wilcoxon rank-sum tests with significance level 0.05 are used for the significance tests. The symbols '+', '≈', and '−' mean that MOACS-noLS is significantly better than, similar to, and significantly worse than the corresponding algorithm, respectively. As can be seen from Table IV, MOACS-noLS outperforms the other two comparison algorithms on HV.

TABLE IV

COMPARISONS OF MOACS-NOLS WITH THE OTHER MULTI-OBJECTIVE OPTIMIZATION ALGORITHMS ON FR AND HV

Instance	NSGA-II		P-ACO		MOACS-noLS	
	FR	HV	FR	HV	FR	HV
I1-727	**100%**	30.583(+)	**100%**	53.276(+)	**100%**	**58.731**
I2-DC9	50%	10.892(+)	**100%**	68.796(+)	**100%**	**85.846**
I3-D94	**100%**	12.477(+)	**100%**	40.939(+)	**100%**	**43.986**
I4-D95	10%	2.391(+)	**100%**	75.445(+)	**100%**	**78.944**
I5-757	0%	2.440(+)	**100%**	89.928(+)	**100%**	**98.650**
I6-319	10%	2.604(+)	**100%**	83.762(+)	**100%**	**92.260**
I7-320	0%	1.625(+)	**100%**	87.030(+)	**100%**	**97.435**

TABLE V

COMPARISONS OF MOACS-NOLS WITH THE OTHER MULTI-OBJECTIVE OPTIMIZATION ALGORITHMS ON $C(X, Y)$

In-stance	NSGA-II		P-ACO	
	C(MOACS-noLS,–)	C(–,MOACS-noLS)	C(MOACS-noLS,–)	C(–,MOACS-noLS)
I1-727	**0.701**	0.443	**0.997**	0.181
I2-DC9	**1.000**	0.000	**0.989**	0.147
I3-D94	**1.000**	0.034	**0.905**	0.218
I4-D95	**1.000**	0.000	**0.979**	0.346
I5-757	**1.000**	0.000	**0.938**	0.342
I6-319	**1.000**	0.000	**0.935**	0.195
I7-320	**1.000**	0.000	**0.988**	0.217

Note that NSGA-II can hardly find a feasible solution on large-scale instances. MOACS-noLS performs better than NSGA-II obviously on seven instances. Compared with P-ACO, MOACS-noLS has a significantly large HV and the gap is more distinct with the growth of instance size.

Moreover, the comparison results on another performance metric $C(X, Y)$ are shown in Table V. The larger values between C(MOACS-noLS, –) and C(–, MOACS-noLS) are in **boldface**. We can find that C(MOACS-noLS, –) values are significantly larger than C(–, MOACS-noLS) values on all seven instances compared with NSGA-II and P-ACO. It indicates that MOACS-noLS outperforms the other two comparison algorithms. Specifically, most C(MOACS-noLS, –) values are very close to one, which denotes most solutions obtained by the other algorithms are dominated by at least one solution of MOACS-noLS. The C(NSGA-II, MOACS-noLS) values on most instances extremely approximate zero except for I1-727, which means very few solutions of MOACS-noLS are dominated by or equal to the solutions of NSGA-II. Although some solutions of MOACS-noLS are dominated by or equal to those found by P-ACO, C(MOACS-noLS, P-ACO) values are still obviously larger than C(P-ACO, MOACS-noLS) values on all instances.

The two performance metrics discussed above focus on different aspects of PF, and MOACS-noLS outperforms the other algorithms on both metrics. HV considers the solution quality in terms of diversity and convergence, and $C(X, Y)$ reflects the dominant relationship between solution sets. Combining these two performance metrics can provide a more convincing result, but lacks an intuitive comparison. In order to visualize the solution sets obtained by different algorithms, we plot all
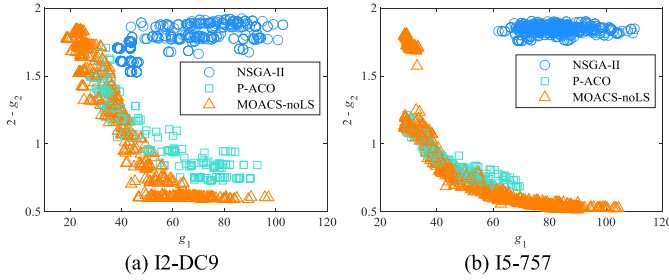
Fig. 6.   Performance of NSGA-II, P-ACO, and MOACS-noLS on two instances. (a) I2-DC9. (b) I5-757.

Fig. 7.   Performance of MOACS-noLS and MOACS on two instances. (a) I2-DC9. (b) I5-757.

the solutions found in 10 runs in Fig. 6 by taking I2-DC9 and I5-757 as examples. The figures for the remaining five instances are shown in Fig. S.1 of the supplementary material. Note that satisfaction g_2 is a maximum objective, we convert the ordinate to $2-g_2$ to observe the solution set clearly, which denotes the sum of dissatisfaction rates and is a minimum objective.

These figures further confirm that MOACS-noLS has overall better performance than the other two compared algorithms. It can be seen from the previous experiments that the solutions of NSGA-II in I2-DC9 and the last four large-scale instances are basically infeasible. Therefore, the PFs of NSGA-II on these instances are quite different from the other algorithms. Compared with P-ACO, MOACS-noLS obtains a broader PF due to its pheromone update rule and HCH strategy. The pheromone update rule increases the diversity of solutions, which helps explore the whole PF. Meanwhile, MOACS-noLS adopts the HCH strategy with three kinds of heuristic information schemes, which also leads to the exploration of the whole PF. By contrast, P-ACO merely uses one integrated heuristic matrix, which results in the solution set focusing only on the central part of the PF.

D. Effectiveness of Local Search Strategy

We propose a local search strategy with two types of local search for the two objectives respectively to help MOACS further approach the global PF. To investigate the effectiveness of the local search strategy, we compare the performance of MOACS-noLS and MOACS using the HV and C(X, Y) metrics. Both algorithms run independently 10 times. The comparison results containing two types of experiments are shown in Table S.II of the supplementary material, where the larger results are in **boldface**. Since both algorithms involve stochastic factors, Wilcoxon rank-sum tests with significance level 0.05 are used to statistically evaluate the results. The symbols '+', '≈', and '−' mean that the performance of MOACS is significantly better than, similar to, and significantly worse than that of MOACS-noLS on HV, respectively.

From the left side of Table S.II, we can see that MOACS outperforms MOACS-noLS with regard to HV on all seven instances. As for the C(X, Y) metric, the values of C(MOACS, MOACS-noLS) are larger than C(MOACS-noLS, MOACS) on all instances, which further confirms that MOACS performs better than MOACS-noLS. Both results reflect the

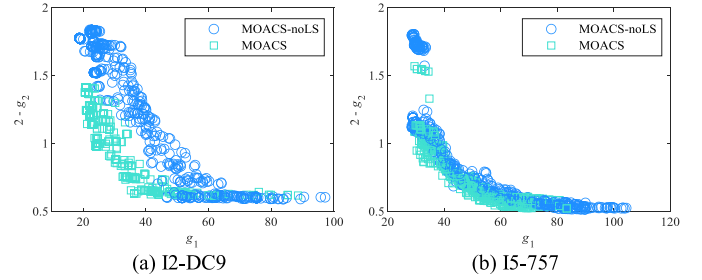effectiveness of the local search strategy. Also taking I2-DC9 and I5-757 as examples, we curve all the solutions obtained in 10 runs to compare both algorithms intuitively and the results are shown in Fig. 7. The PF distribution of MOACS is better than that of MOACS-noLS on both small-scale instance I2-DC9 and large-scale instance I5-757. Since two types of local search are performed randomly, the solutions are able to approach the global PF with better g_1 or g_2, which helps improve their quality. In conclusion, all the above results validate the effectiveness of the local search strategy.

E. Parameters Analysis

The MOACS parameters include m, β, q_0, ρ, ε, and θ. The settings of β, q_0, ρ, and ε are the typical scheme of ACS, so only m and θ are analyzed. The investigation results are plotted in Fig. S.2 of the supplementary material. The x-axis of the figure denotes the corresponding parameter settings and the y-axis denotes the average HV among 10 independent runs. Note that when testing a parameter, all other parameters are set the same as stated in Section IV-B.

Firstly, we analyze the influence of the parameter m. We set m from 20 to 60 with a step of 10. The comparison results of average HV on all seven instances with respect to the parameter m are shown in Fig. S.2(a). From Fig. S.2(a), there seems to be no significant difference in the performance of all settings. The medium-sized m such as "40" has a slightly large HV on the first three instances, but its performance is slightly worse than the other settings on most large-scale instances. On the one hand, smaller ant colonies can perform more generations in the same execution time, so the local search strategy will be executed more times, which is helpful for exploitation. On the other hand, an increase in the number of ants can also help increase the search area [37], but it will reduce the number of times that the local search strategy is executed. Since these two factors are in conflict, m is insensitive in MOACS. Finally, we select the setting "60" because it has the overall best performance.

The other parameter we study is the selection rate θ proposed in the global update of pheromones in Section III-D. We set θ from 0 to 0.4 with a step size of 0.1. Note that $\theta = 0$ represents that MOACS always selects the best solution of each objective as the *GUS*. The comparison results of average HV on all seven instances in regard to the parameter θ are shown in Fig. S.2(b). From Fig. S.2(b), we can find that

the setting "0.1" for θ obtains the best performance among other settings on most instances. On the first three small-scale instances, the larger θ can get better HV for it provides more diversity. However, an excessively large θ brings a worse HV on the large-scale instances I4-D95 and I6-319 due to the reduction in convergence efficiency. The results show that the setting "0.1" for θ is suitable for balancing the solution diversity and search efficiency. Moreover, the setting "0.1" is better than "0" on most instances, which also confirms the effectiveness of the global update strategy.

V. Conclusion

Different from most existing models that consider the cost from the perspective of airlines, this study proposes a new multi-objective CRP model from the perspective of crew members. Specifically, the fairness and satisfaction of crew are taken into account at the same time. Combining the ACS optimizer and the MPMO framework, a novel MOACS algorithm is developed to solve the multi-objective CRP efficiently. Under the MPMO framework, two ant colonies are employed to optimize fairness and satisfaction objectives, respectively. Besides, to prevent ant colonies from concentrating only on their own objectives, we propose a new HCH strategy with three kinds of heuristic information schemes. It randomly uses the heuristic information of two ant colonies and the aggregated heuristic information, which helps explore the global PF. Furthermore, we design a local search strategy with two types of local search for fairness and satisfaction objectives, respectively, which helps approach the global PF.

Experiments are conducted on monthly CRP with different sizes from the real world. The results verify the effectiveness of MOACS in solving multi-objective CRP. It generally outperforms the greedy algorithm and other multi-objective evolutionary-based algorithms, especially on large-scale instances. In conclusion, MOACS is more effective than the compared algorithms in dealing with the multi-objective CRP. In future work, the number and qualifications of crew members required for flights will be further considered in the CRP model.

References

[1] M. Deveci and N. C. Demirel, "A survey of the literature on airline crew scheduling," *Eng. Appl. Artif. Intell.*, vol. 74, pp. 54–69, Sep. 2018.

[2] N. Kohl and S. E. Karisch, "Airline crew rostering: Problem types, modeling, and optimization," *Ann. Oper. Res.*, vol. 127, nos. 1–4, pp. 223–257, Mar. 2004.

[3] B. Gopalakrishnan and E. L. Johnson, "Airline crew scheduling: State-of-the-art," *Ann. Oper. Res.*, vol. 140, no. 1, pp. 305–337, Nov. 2005.

[4] F. M. Zeghal and M. Minoux, "Modeling and solving a crew assignment problem in air transportation," *Eur. J. Oper. Res.*, vol. 175, no. 1, pp. 187–209, Nov. 2006.

[5] B. Santosa, A. Sunarto, and A. Rahman, "Using differential evolution method to solve crew rostering problem," *Appl. Math.*, vol. 1, no. 4, pp. 316–325, 2010.

[6] O. Ezzinbi, M. Sarhani, A. El Afia, and Y. Benadada, "Particle swarm optimization algorithm for solving airline crew scheduling problem," in *Proc. Int. Conf. Logistics Oper. Manage.*, Jun. 2014, pp. 52–56.

[7] T. Doi, T. Nishi, and S. Voß, "Two-level decomposition-based matheuristic for airline crew rostering problems with fair working time," *Eur. J. Oper. Res.*, vol. 267, no. 2, pp. 428–438, Jun. 2018.

[8] J. de Armas, L. Cadarso, A. A. Juan, and J. Faulin, "A multi-start randomized heuristic for real-life crew rostering problems in airlines with work-balancing goals," *Ann. Oper. Res.*, vol. 258, no. 2, pp. 825–848, Nov. 2017.

[9] P. Lučić and D. Teodorović, "Metaheuristics approach to the aircrew rostering problem," *Ann. Oper. Res.*, vol. 155, no. 1, pp. 311–338, Aug. 2007.

[10] C. Boufaied, R. Trabelsi, H. Masri, and S. Krichen, "A construction of rotations-based rosters with a genetic algorithm," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jul. 2016, pp. 2389–2394.

[11] A. T. Ernst, H. Jiang, M. Krishnamoorthy, and D. Sier, "Staff scheduling and rostering: A review of applications, methods and models," *Eur. J. Oper. Res.*, vol. 153, no. 1, pp. 3–27, Feb. 2004.

[12] M. Gamache, F. Soumis, D. Villeneuve, J. Desrosiers, and É. Gélinas, "The preferential bidding system at air Canada," *Transp. Sci.*, vol. 32, no. 3, pp. 246–255, Aug. 1998.

[13] H. Dawid, J. König, and C. Strauss, "An enhanced rostering model for airline crews," *Comput. Oper. Res.*, vol. 28, no. 7, pp. 671–688, Jun. 2001.

[14] B. Maenhout and M. Vanhoucke, "A hybrid scatter search heuristic for personalized crew rostering in the airline industry," *Eur. J. Oper. Res.*, vol. 206, no. 1, pp. 155–167, Oct. 2010.

[15] A. Kasirzadeh, M. Saddoune, and F. Soumis, "Airline crew scheduling: Models, algorithms, and data sets," *EURO J. Transp. Logist.*, vol. 6, no. 2, pp. 111–137, Jun. 2017.

[16] I. Das and J. E. Dennis, "A closer look at drawbacks of minimizing weighted sums of objectives for Pareto set generation in multicriteria optimization problems," *Struct. Optim.*, vol. 14, no. 1, pp. 63–69, Aug. 1997.

[17] Ministry of Transport of the People's Republic of China. 14th Ministerial Meeting (Aug. 29, 2017). *Large Aircraft Public Air Transport Carrier Operation Certification Rules*. [Online]. Available: http://www.caac.gov.cn/XXGK/XXGK/MHGZ/201710/P020171009385743667633.pdf

[18] 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.

[19] T. Liao, K. Socha, M. A. M. de Oca, T. Stützle, and M. Dorigo, "Ant colony optimization for mixed-variable optimization problems," *IEEE Trans. Evol. Comput.*, vol. 18, no. 4, pp. 503–518, Aug. 2014.

[20] Z. H. Zhan, J. Zhang, Y. Li, O. Liu, S. K. Kwok, W. H. Ip, and O. Kaynak, "An efficient ant colony system based on receding horizon control for the aircraft arrival sequencing and scheduling problem," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 2, pp. 399–412, Jun. 2010.

[21] J. Eaton, S. Yang, and M. Gongora, "Ant colony optimization for simulated dynamic multi-objective railway junction rescheduling," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 11, pp. 2980–2992, Nov. 2017.

[22] X. F. Liu, Z. H. Zhan, J. D. Deng, Y. Li, T. Gu, and J. Zhang, "An energy efficient ant colony system for virtual machine placement in cloud computing," *IEEE Trans. Evol. Comput.*, vol. 22, no. 1, pp. 113–128, Feb. 2018.

[23] Y.-H. Zhang, Y.-J. Gong, W.-N. Chen, T.-L. Gu, H.-Q. Yuan, and J. Zhang, "A dual-colony ant algorithm for the receiving and shipping door assignments in cross-docks," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 7, pp. 2523–2539, Jul. 2019.

[24] R. Cheng, Y. Song, D. Chen, and X. Ma, "Intelligent positioning approach for high speed trains based on ant colony optimization and machine learning algorithms," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 10, pp. 3737–3746, Oct. 2019.

[25] Z.-H. Zhan, J. Li, J. Cao, J. Zhang, H. S.-H. Chung, and Y.-H. Shi, "Multiple populations for multiple objectives: A coevolutionary technique for solving multiobjective optimization problems," *IEEE Trans. Cybern.*, vol. 43, no. 2, pp. 445–463, Apr. 2013.

[26] X.-F. Liu, Z.-H. Zhan, Y. Gao, J. Zhang, S. Kwong, and J. Zhang, "Coevolutionary particle swarm optimization with bottleneck objective learning strategy for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 23, no. 4, pp. 587–602, Aug. 2019.

[27] D. Liang, Z. H. Zhan, Y. C. Zhang, and J. Zhang, "An efficient ant colony system approach for new energy vehicle dispatch problem," *IEEE Trans. Intell. Transp. Syst.*, early access, Oct. 17, 2019, doi: 10.1109/TITS.2019.2946711.

[28] C. Y. Lu, J. H. Zhong, Y. X. Xue, L. Feng, and J. Zhang, "Ant colony system with sorting-based local search for coverage-based test case prioritization," *IEEE Trans. Rel.*, early access, Aug. 9, 2019, doi: 10.1109/TR.2019.2930358.

[29] C.-H. Chen, T.-K. Liu, and J.-H. Chou, "Integrated short-haul airline crew scheduling using multiobjective optimization genetic algorithms," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 43, no. 5, pp. 1077–1090, Sep. 2013.

[30] P. Lučić and D. Teodorović, "Simulated annealing for the multi-objective aircrew rostering problem," *Transp. Res. A, Policy Pract.*, vol. 33, no. 1, pp. 19–45, Jan. 1999.

[31] B. Barán and M. Schaerer, "A multiobjective ant colony system for vehicle routing problem with time windows," in *Proc. 21st IASTED Int. Conf. Appl. Inf.*, 2003, pp. 97–102.

[32] K. F. Doerner, R. F. Hartl, and M. Reimann, "Are CompetAnts more competent for problem solving? The case of a multiple objective transportation problem," *Cent. Eur. J. Oper. Res. Econ.*, vol. 11, no. 2, pp. 115–141, 2003.

[33] Z.-G. Chen *et al.*, "Multiobjective cloud workflow scheduling: A multiple populations ant colony system approach," *IEEE Trans. Cybern.*, vol. 49, no. 8, pp. 2912–2926, Aug. 2019.

[34] M. López-Ibáñez and T. Stützle, "The impact of design choices of multiobjective antcolony optimization algorithms on performance: An experimental study on the biobjective TSP," in *Proc. 12th Annu. Conf. Genetic Evol. Comput. (GECCO)*, 2010, pp. 71–78.

[35] K. Doerner, W. J. Gutjahr, R. F. Hartl, C. Strauss, and C. Stummer, "Pareto ant colony optimization: A metaheuristic approach to multiobjective portfolio selection," *Ann. Oper. Res.*, vol. 131, nos. 1–4, pp. 79–99, Oct. 2004.

[36] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach," *IEEE Trans. Evol. Comput.*, vol. 3, no. 4, pp. 257–271, Nov. 1999.

[37] M. Dorigo and T. Stützle, "An experimental study of the simple ant colony optimization algorithm," in *Proc. Int. Conf. Evol. Comput. (WSES)*, 2001, pp. 253–258.

Shu-Zi Zhou (Student Member, IEEE) received the B.S. degree in information and computing science from the South China University of Technology, Guangzhou, China, in 2018, where she is currently pursuing the M.S. degree.

Her current research interests include evolutionary computation, and swarm intelligence and their applications to real-world constrained optimization problems.

Zhi-Hui Zhan (Senior Member, IEEE) received the bachelor's and Ph.D. degrees in computer science from Sun Yat-sen University, Guangzhou, China, in 2007 and 2013, respectively.

He is currently the Changjiang Scholar Young Professor and the Pearl River Scholar Young Professor with the School of Computer Science and Engineering, South China University of Technology, China. His current research interests include evolutionary computation algorithms, swarm intelligence algorithms and their applications in real-world problems, and environments of cloud computing and big data.

Dr. Zhan was a recipient of the Wu Wen Jun Artificial Intelligence Excellent Youth from the Chinese Association for Artificial Intelligence in 2017 and the Outstanding Youth Science Foundation from the National Natural Science Foundations of China (NSFC) in 2018. His doctoral dissertation was awarded the IEEE Computational Intelligence Society (CIS) Outstanding Ph.D. Dissertation and the China Computer Federation (CCF) Outstanding Ph.D. Dissertation. He is listed as one of the most cited Chinese researchers in computer science. He is currently an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, *Neurocomputing*, and the *International Journal of Swarm Intelligence Research*.

Zong-Gan Chen (Student Member, IEEE) received the B.S. degree from Sun Yat-sen University, Guangzhou, China, in 2016. He is currently pursuing the Ph.D. degree in computer science and technology with the South China University of Technology, Guangzhou.

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

Sam Kwong (Fellow, IEEE) received the B.S. degree in electrical engineering from The State University of New York at Buffalo, USA, in 1983, the M.S. degree in electrical engineering from the University of Waterloo, Canada, in 1985, and the Ph.D. degree from the University of Hagen, Germany, in 1996.

From 1985 to 1987, he was a Diagnostic Engineer with Control Data Canada, where he designed the diagnostic software to detect the manufactured faults of the VLSI chips in the cyber 430 machine. He later joined the Bell Northern Research, Canada, as a Member of Scientific Staff. In 1990, he joined the Department of Electronics Engineering, City University of Hong Kong, as a Lecturer. He is currently a Chair Professor with the Department of Computer Science. His research interests include pattern recognition, evolutionary computations, and video analytics.

Prof. Kwong was elevated to an IEEE Fellow for his contributions to optimization techniques for cybernetics and video coding in 2014. He is the Vice President of the IEEE Systems, Man, and Cybernetics. He was also appointed as an IEEE Distinguished Lecturer of the IEEE SMC Society in March 2017.

Jun Zhang (Fellow, IEEE) received the Ph.D. degree in electrical engineering from the City University of Hong Kong, Hong Kong, in 2002.

He is currently a Visiting Scholar with Hanyang University, Ansan, South Korea. His current research interests include computational intelligence, cloud computing, high performance computing, operations research, and power electronic circuits.

Dr. Zhang was a recipient of the Changjiang Chair Professor from the Ministry of Education, China, in 2013, the China National Funds for Distinguished Young Scientists from the National Natural Science Foundation of China in 2011, and the First-Grade Award of the Natural Science Research from the Ministry of Education, China, in 2009. He is currently an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, the IEEE TRANSACTIONS ON CYBERNETICS, and the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS.