

An improved multi-objective evolutionary algorithm for airline crew pairing problem^{*}

LI Cong, JIANG Zhipeng[†], YANG Wenguo

(School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing 100049, China)

(Received 24 January 2024; Revised 12 June 2024)

Li C, Jiang Z P, Yang W G. An improved multi-objective evolutionary algorithm for airline crew pairing problem[J].
Journal of University of Chinese Academy of Sciences, 2026, 43(5): 590-602. DOI: 10. 7523/j.ucas. 2024. 064.

Abstract The multi-objective airline crew pairing problem tries to find a subset of feasible pairings such that all flights are covered and each objective function is minimized. For this problem, we propose two mathematical models: a novel mathematical model that contains logical relationships, and an integer programming model based on all feasible pairings enumerated by a depth-first search method. To solve the problem, we propose an improved multi-objective evolutionary algorithm, which is a non-dominated sorting genetic algorithm that distinguishes individuals using the distance between the objective vector corresponding to each solution and an adaptive evaluation vector. The proposed algorithm uses the direction of the adaptive evaluation vector and the Pareto orientation for guidance to derive Pareto solutions. We also use a repairing strategy and a local optimization strategy for deriving feasible and better solutions. For this problem, in our experimental results, the performance of the proposed algorithm is superior to those of the traditional nondominated sorting genetic algorithm II and another multi-objective genetic algorithm.

Keywords airline crew pairing; multi-objective optimization; Pareto solutions

CLC number: O221.4

Document code: A

DOI: 10. 7523/j.ucas. 2024. 064

面向航空公司机组配对问题的 改进多目标进化算法

李聪, 姜志鹏, 杨文国

(中国科学院大学数学科学学院, 北京 100049)

摘要 具有多个目标的航空公司机组配对问题目的是寻找一个可行的任务环子集, 使其覆盖所有航班, 同时每个目标函数最小化。针对该问题, 构建 2 种不同的数学模型, 一种是包含逻辑关系的模型, 另一种是整数规划模型。为求解这一目标问题, 提出一种改进的多目标进化算法,

^{*} Supported by National Natural Science Foundation of China (12071459)

[†] Corresponding author, E-mail: jiangzhipeng@ucas.ac.cn

该算法基于非支配排序遗传算法框架,通过计算每个解对应的目标向量和自适应评估向量之间的距离区分种群个体。算法主要利用自适应评估向量和帕累托方向引导搜索,以获得帕累托解。为进一步提升解的质量和可行性,还引入修复策略和局部优化策略。实验结果表明,所提出的多目标进化算法性能上优于传统的非支配排序遗传算法Ⅱ及另一种多目标遗传算法。

关键词 航空公司机组配对;多目标优化;帕累托解

The airline planning problem is a very challenging problem. However, because of the high complexity of this problem, it is often divided into four subproblems: flight arrangement problem, fleet assignment problem, aircraft routing problem, and crew scheduling problem. In fact, the optimal solution of each subproblem is the input of the next subproblem. These topics have been reviewed by Wen et al.^[1]

Because crew costs are the second largest after fuel costs for airlines, the crew scheduling problem is of great significance. However, the crew scheduling problem is an NP-hard problem, which means that it can not be exactly solved within a reasonable computational time. Thus, the crew scheduling problem usually consists of two subproblems: the crew pairing problem and the crew rostering problem. Most researchers have been concentrated on the crew pairing problem^[2-3]. Evolutionary algorithms (EAs) are often used to solve the single-objective airline crew pairing problem. To derive a solution, the solving process using evolutionary algorithms generally is segmented into two stages: the enumeration of the feasible pairings and the optimization process. Zeren and Özkol^[4] proposed a perturbation operator to improve a genetic algorithm, this operator enhanced the convergence of the genetic algorithm and the stability of the results. In addition, they also considered the case of multiple crew bases. Çetin and Deveci^[5] proposed an improved genetic algorithm with updating search space to solve the crew pairing problem with medium scale. Moreover, their study possessed characteristics of multiple objectives. But they still solved the single-objective crew pairing problem. In 2018, they also proposed two variants of a genetic algorithm and a memetic algorithm^[6]. In their experimental results, they

indicated that the performance of the memetic algorithm was better than that of the other genetic algorithms. Currently, many authors focus on the single-objective crew pairing problem, where this problem finds a subset of minimum-cost feasible pairings that cover all flights. However, for the multi-objective crew pairing problem, few authors have conducted the research.

Generally, for a multi-objective problem, traditional approaches are used to reformulate the multi-objective problem into a single objective problem. The weighted sum is one of the conventional approaches. On one side, the priority of objectives can be considered to set the weighted parameter of each objective by the weighted sum approach in the solving process. However, we often do not know their priority or what setting weighted parameters is appropriate, and the derived solution may depend on the used weighted parameters. On the other side, in the solving process, one tries to find the best solution that is often the global maximum or global minimum for a single objective problem. However, for the multi-objective problem, the best solution may not exist^[7], there exists a set of Pareto-optimal or non-dominated solutions. Therefore, we prefer to find multiple Pareto solutions by a multi-objective optimization algorithm in this paper. Multi-objective evolutionary algorithms (MOEAs) are one of the most effective tools for finding multiple Pareto solutions^[8]. In recent decades, MOEAs have been constantly studied and applied. Schaffer^[9] proposed a vector evaluated genetic algorithm (VEGA) for solving multi-objective optimization problems, and the author opened up a new research field of MOEAs. Subsequently, various MOEAs have been proposed to solve multi-objective optimization problems. Horn et al.^[10] proposed a niched Pareto genetic

algorithm (NPGA). Srinivas and Deb^[7] proposed a non-dominated sorting genetic algorithm (NSGA), and indicated that the NSGA was better than the VEGA through experimental comparisons. Deb et al.^[11] proposed a non-dominated sorting genetic algorithm with lower complexity and elitism, the algorithm is called NSGA- II. There are also many other MOEAs, for example, MOGA^[12], SPEA^[13], MOEA/D^[14], NSGA-III^[15], and LMEA^[16]. In these algorithms, in particular, NSGA- II is extensively applied in many fields. NSGA- II uses the fast non-dominated sorting method for ranking and the crowded distance for preserving diversity at each rank. NSGA- II and its variants have been applied successfully by Chen et al.^[8] to solve the integrated aircraft routing and crew pairing problems, and by Chen et al.^[17] to address the integrated short-haul airline crew scheduling problem. In addition, Chou et al.^[18] proposed a method of inequality-based multiobjective genetic algorithm (called MMGA) for solving an aircraft routing problem.

In this paper, to solve a multi-objective airline crew pairing problem (MOCPP), we first propose a novel mathematical formulation and an integer programming formulation. Secondly, we propose an improved multi-objective evolutionary algorithm, namely, a non-dominated sorting genetic algorithm with the distance between the objective vector corresponding to each solution and an adaptive evaluation vector (NSGA-AEV). In other words, we propose a novel method for preserving diversity in this improved multi-objective evolutionary algorithm, where the novel method contains the distance between the objective vector corresponding to each solution and an adaptive evaluation vector, instead of the crowded distance used in NSGA- II. Therefore, NSGA-AEV contains the direction of the adaptive evaluation vector and the Pareto orientation as guidance in the iterative process. By contrast, NSGA- II only provides the Pareto-based guidance. A repair strategy and a local optimization strategy are used to find feasible and better solutions. Finally, we will indicate that the performance of NSGA-AEV is better than those of NSGA- II and

MMGA, via the comparison of the experimental results.

1 Problem definition

For the MOCPP, we consider several aspects relevant to operational costs. The number of duties determines the number of required crew, which further determines the personnel cost structure. Overnight at the nonbase incurs additional overnight expenses for crews to perform their next duty. The deadhead not only adds extra transportation costs but also may lead to reducing crew availability.

Given a crew base and a set of flights operated by an aircraft type, the MOCPP tries to find a subset of feasible pairings such that the total number of duties, the total number of deadheads, and the total number of overnights are minimized, where each flight is covered at least once by pairings and each pairing starts and ends at the same base. A pairing contains a sequence of one or more duties separated by overnights. A duty is a time period that is composed of a sequence of flights or deadheads separated by sit time. For two successive flights in a pairing, the arrival airport of a flight must be the departure airport of the next flight. A flight is the time period from aircraft take-off to landing, it is also called flight leg or leg. If a flight in which the crew is treated as passengers transfers the crew from one airport to another for performing subsequent flights or back to base, this flight is called a deadhead. Because two successive flights within a duty need time for a turnaround, this time is called sit time or transition time. A connection between two successive duties in a pairing is called an overnight, layover or rest.

In addition, duties and pairings satisfy some constraints for safety. Each duty must be subject to the maximum number of landings (denoted as $\alpha^{\max}$), the maximum flying time (denoted as $\beta^{\max}$), and the maximum duty elapsed time (denoted as $\gamma^{\max}$). We denote by $s^{\min}$ and $s^{\max}$ the minimum and maximum of the sit time, respectively. Each pairing must be subject to the maximum number of duties

(denoted as $\delta^{\max}$) and the maximum pairing elapsed time (denoted as $\varepsilon^{\max}$). We denote by $o^{\min}$ the minimum overnight delivery.

2 Problem formulation

According to the definition of the MOCPP, in this section, we propose a novel mathematical formulation that contains logical relationships and an integer programming formulation based on all enumerated feasible pairings.

2.1 A novel mathematical formulation

Haouari et al. ^[19] proposed a digraph that contains four arc types for the daily crew pairing problem. Similarly, we also present a digraph that contains four arc types for the MOCPP. We denote by F the set of all flights, by F^D the set of flights that depart from the crew base, and by F^A the set of flights that arrive at the crew base, where each flight $j \in F$ is characterized by t_j the corresponding duration, $\hat{t}_j$ the departure time, $\hat{s}_j$ the departure airport, $\bar{t}_j$ the arrival time, and $\bar{s}_j$ the arrival airport. Let b be the crew base. We define s_{ij} and o_{ij} as the sit time between two successive flights $i \in F$ and $j \in F$ in a duty, and the overnight time between two successive flights $i \in F$ and $j \in F$ that belong to two different duties in a pairing, respectively. The sit time s_{ij} and overnight time o_{ij} are computed as follows:

- $s_{ij} = \hat{t}_j - \bar{t}_i$, if $s^{\min} \leq s_{ij} \leq s^{\max}$;
- $o_{ij} = \hat{t}_j - \bar{t}_i$, if $o^{\min} \leq o_{ij}$.

We define an acyclic digraph $G = (N, E)$ for a given flight schedule, where $N = F \cup \{o, d\}$ is the set of nodes and $E = E_D \cup E_S \cup E_O \cup E_A$ is the set of arcs. Figure 1 shows an example of the graph G . Each flight $j \in F$ is represented by a node. The source node o and sink node d represent the crew base b that a pairing starts and ends. Each legal connection between $i \in N$ and $j \in N$ is represented by an arc $(i, j) \in E$. The set E is composed of the set E_D of start arcs, the set E_S of sit-time arcs, the set E_O of overnight arcs, and the set E_A of end arcs that are defined as follows:

- An arc $(o, j) \in E_D$ if and only if $j \in F^D$. The arc $(o, j) \in E_D$ means that the first flight of a pairing starts from the crew base b .
- An arc $(i, j) \in E_S$ if and only if flights $i \in F$ and $j \in F$ are successive in a duty. Namely, the arc $(i, j) \in E_S$ must satisfy (i) $\bar{s}_i = \hat{s}_j$; (ii) $s_{ij} \in [s^{\min}, s^{\max}]$; (iii) $t_i + t_j \leq \beta^{\max}$; and (iv) $t_i + s_{ij} + t_j \leq \gamma^{\max}$.
- An arc $(i, j) \in E_O$ if and only if flights $i \in F$ and $j \in F$ are successive and belong to two different duties in the pairing. Namely, the arc $(i, j) \in E_O$ must satisfy (i) $\bar{s}_i = \hat{s}_j \neq b$; (ii) $o_{ij} \geq o^{\min}$; and (iii) $t_i + o_{ij} + t_j \leq \varepsilon^{\max}$.
- An arc $(j, d) \in E_A$ if and only if $j \in F^A$. The arc $(j, d) \in E_A$ means that the last flight of a pairing

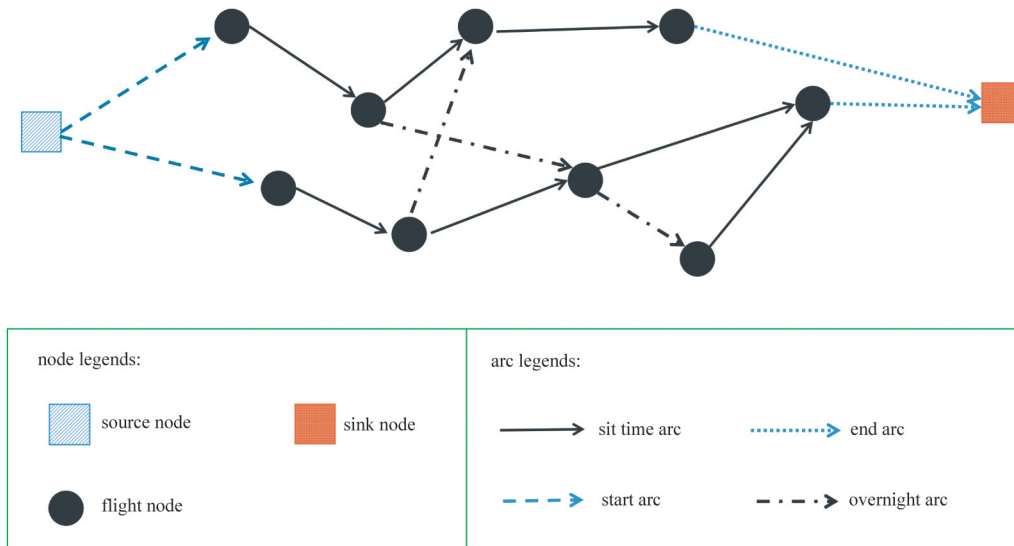


Fig. 1 An example of graph structure for crew pairing problem

ends at the crew base b .

Since each path from the source node to the sink node in G corresponds to a pairing, we denote by P^* all paths from the source node to the sink node in G . For a path $p \in P^*$ and a flight $j \in p \cap F$, we denote by α_j^p the total accumulated number of landings from the source node to the node j , by β_j^p the total accumulated duty flying time from the source node to the node j , by γ_j^p the total accumulated duty elapsed time from the source node to the node j , by δ_j^p the total accumulated number of duties from the source node to the node j , and by ε_j^p the total accumulated pairing elapsed time from the source node to the node j , where α_j^p , β_j^p , γ_j^p , δ_j^p , and ε_j^p are computed as follows:

- For an arc $(i, j) \in p$, if $(i, j) \in E_D$, then $\alpha_j^p = 1$, $\beta_j^p = t_j$, $\gamma_j^p = t_j$, $\delta_j^p = 1$, and $\varepsilon_j^p = t_j$.

- For an arc $(i, j) \in p$, if $(i, j) \in E_S$, then $\alpha_j^p = \alpha_i^p + 1$, $\beta_j^p = \beta_i^p + t_j$, $\gamma_j^p = \gamma_i^p + s_{ij} + t_j$, $\delta_j^p = \delta_i^p$, and $\varepsilon_j^p = \varepsilon_i^p + s_{ij} + t_j$.

- For an arc $(i, j) \in p$, if $(i, j) \in E_O$, then $\alpha_j^p = 1$, $\beta_j^p = t_j$, $\gamma_j^p = t_j$, $\delta_j^p = \delta_i^p + 1$, and $\varepsilon_j^p = \varepsilon_i^p + o_{ij} + t_j$.

Because the MOCPP finds a subset of pairings such that all objective functions are minimal, each flight is covered, and each pairing satisfies all restrictions and rules of pairing construction, we need to find this subset that satisfies all conditions. We denote by P' the subset of P^* . Let F_p be all flights covered by $p \in P^*$. In addition, if a path $p \in P^*$ contains the node $j \in N$, it is defined as $a_j^p = 1$; otherwise $a_j^p = 0$. The restrictions of the problem are formulated as follows:

- The number of landings per duty in a pairing restriction

$$\eta_1(P') = \sum_{p \in P'} \sum_{j \in F_p} \pi_{p,j}^{(1)},$$

where $\pi_{p,j}^{(1)} = \begin{cases} 0, & \text{if } \alpha_j^p \leq \alpha^{\max}, \\ 1, & \text{otherwise.} \end{cases}$

The number of landings per duty in a feasible pairing should not exceed $\alpha^{\max}$.

- The flying time per duty in a pairing restriction

$$\eta_2(P') = \sum_{p \in P'} \sum_{j \in F_p} \pi_{p,j}^{(2)},$$

where $\pi_{p,j}^{(2)} = \begin{cases} 0, & \text{if } \beta_j^p \leq \beta^{\max}, \\ 1, & \text{otherwise.} \end{cases}$

The duty flying time should not exceed $\beta^{\max}$ in a feasible pairing.

- The duty elapsed time per duty in a pairing restriction

$$\eta_3(P') = \sum_{p \in P'} \sum_{j \in F_p} \pi_{p,j}^{(3)},$$

where $\pi_{p,j}^{(3)} = \begin{cases} 0, & \text{if } \gamma_j^p \leq \gamma^{\max}, \\ 1, & \text{otherwise.} \end{cases}$

The duty elapsed time should not exceed $\gamma^{\max}$ in a feasible pairing.

- The number of duties in a pairing restriction

$$\eta_4(P') = \sum_{p \in P'} \sum_{j \in F_p} \pi_{p,j}^{(4)},$$

where $\pi_{p,j}^{(4)} = \begin{cases} 0, & \text{if } \delta_j^p \leq \delta^{\max}, \\ 1, & \text{otherwise.} \end{cases}$

The number of duties should not exceed $\delta^{\max}$ for a feasible pairing.

- The pairing elapsed time restriction

$$\eta_5(P') = \sum_{p \in P'} \sum_{j \in F_p} \pi_{p,j}^{(5)},$$

where $\pi_{p,j}^{(5)} = \begin{cases} 0, & \text{if } \varepsilon_j^p \leq \varepsilon^{\max}, \\ 1, & \text{otherwise.} \end{cases}$

The pairing elapsed time should not exceed $\varepsilon^{\max}$ for a feasible pairing.

- The complete including all flights restriction

$$\eta_6(P') = \sum_{j \in F} \pi_p^{(6)},$$

where $\pi_p^{(6)} = \begin{cases} 0, & \text{if } \sum_{j \in F} a_j^p \geq 1, \\ 1, & \text{otherwise.} \end{cases}$

Each flight is covered at least once by feasible pairings.

Next, the objective functions of the problem are formulated as follows:

- The total number of duties is defined as

$$\varphi_1(P') = \sum_{p \in P'} z_p,$$

where z_p is the number of duties for the pairing p .

- The total number of deadheads is defined as

$$\varphi_2(P') = \sum_{p \in P'} w_p,$$

where w_p is the number of deadheads for the pairing p .

- The total number of overnights at the nonbase is defined as

$$\varphi_3(P') = \sum_{p \in P'} o_p,$$

where o_p is the number of overnights at the nonbase for the pairing p .

Finally, we propose the following formulation

$$\text{minimize } \varphi_1(P'), \varphi_2(P'), \varphi_3(P'); \quad (1)$$

$$\text{subject to } \eta_k(P') = 0, \quad 1 \leq k \leq 6. \quad (2)$$

where P' is a subset of P^* and is also a variable. The objective function (1) finds a subset P' of P^* to minimize the total number of duties, the total number of deadheads, and the total number of overnights. In the formula (2), constraints $\eta_k(P') = 0, \quad 1 \leq k \leq 5$ ensure that each pairing in P' is feasible, in addition, constraint $\eta_6(P') = 0$ ensures that each flight is covered at least once by pairings in P' , and flights covered more than once by pairings are considered as deadheads.

The formulation is not a traditional programming formulation. In fact, it is a formulation that contains logical relationships.

2.2 Integer programming formulation

To solve the proposed formulation, we must find all paths from the source node to the sink node in G , but this can lead to a very long computational time. To reduce the computational time, we enumerate all feasible pairings by a depth-first search algorithm^[6, 20]. In addition, because of the large restrictions of pairing construction, for the computational convenience, we enumerate all feasible duties by the depth-first search algorithm. Subsequently, we enumerate all feasible pairings based on all feasible duties by the depth-first search algorithm. Next, we propose an integer programming formulation based on all feasible pairings.

We denote the set of all feasible pairings by P . Based on all feasible pairings, the MOCPP is reformulated as follows

$$\text{minimize } \sum_{p \in P} z_p x_p; \quad (3)$$

$$\text{minimize } \sum_{p \in P} w_p x_p; \quad (4)$$

$$\text{minimize } \sum_{p \in P} o_p x_p; \quad (5)$$

$$\text{subject to } \sum_{p \in P} a_j^p x_p \geq 1, \quad j \in F; \quad (6)$$

$$x_p \in \{0, 1\}, \quad p \in P. \quad (7)$$

Three objective functions (3)-(5) find a subset of feasible pairings to minimize the total number of duties, the total number of deadheads, and the total number of overnights. Constraint (6) ensures that each flight is included at least once in feasible pairings, and flights covered more than once by pairings are considered as deadheads. In the formula (7), x_p is a binary variable and means that if pairing p is selected, then $x_p = 1$; otherwise, $x_p = 0$.

3 Solution methodology

In this section, to solve the MOCPP, we propose an improved multi-objective evolutionary algorithm called NSGA-AEV. The biggest difference between MOEAs and EAs is the fitness assignment of the population. Therefore, like other genetic algorithms (GAs), the iterative process of NSGA-AEV also includes these stages: initial population generation, fitness calculation, selection, crossover, and mutation. In addition, a chromosome in NSGA-AEV is a solution of the MOCPP, which is also called an individual. The proposed NSGA-AEV is described in detail below.

Consider any two feasible solutions x and y corresponding to objective vectors X and Y , respectively. x dominates y if and only if the following expression is satisfied

$$\forall i (X_i \leq Y_i) \wedge \exists i (X_i < Y_i),$$

where $1 \leq i \leq M$, and M is the number of objective functions, it is denoted as $X < Y$ or $x < y$. If a solution can not be dominated by others, then the solution is called a Pareto-optimal solution (Pareto solution) or non-dominated solution. All Pareto-optimal solutions form a Pareto-optimal set.

Liu et al.^[21] have indicated that the convergence rate of GAs with an optimal vector treated as a guiding search direction may be more effective than that with the guidance of multiple directions. However, the centralized search direction may cause the loss of some properties in this way. To avoid this, we use the Pareto orientation and the direction of the adaptive evaluation vector (AEV)

for guidance in this paper.

3.1 The non-dominated sorting and ranking method

We use the same method used in NSGA-II for the non-dominated sorting and ranking. Please refer to Deb et al. [11] for more details on the non-dominated sorting and ranking method. The aim of the non-dominated sorting and ranking is to group all individuals in the population into several ranks. Figure 2(a) shows an example that individuals in the population are grouped into three ranks. At this example, all individuals at the same rank don't dominate each other, and each individual at a lower rank dominates each individual at a higher rank.

3.2 The proposed preserving diversity method

Because it is unknown whether an optimal solution and its corresponding objective vector exist for the MOCPP, an adaptive evaluation vector (AEV) is used for guidance. Firstly, we compute the minimum value of each objective function for the current population. Secondly, these minimum values are formed as the following vector: $AEV = (\xi_1, \xi_2, \dots, \xi_M)$. The value of AEV may vary in the iterative process. To preserve diversity, the distance between the AEV and the objective vector corresponding to each individual is computed. The pseudo-code of the proposed algorithm is given in Algorithm 1.

In step 1, we initialise $k = 1$. The value of the AEV is computed in step 2. In steps 3-5, when k is less than or equal to the maximum number K of ranks, we set all individuals $I_1, I_2, \dots, I_{n_k}$ at the rank k and set $j = 1$. When j is less than or equal to the number M of objective functions, the j th objective function values of all individuals at the rank k are computed in steps 6 and 7. In step 8, the distance between the j -th objective function value of each individual at the rank k and the j th component of the AEV is computed by the expression (8). In addition, because the denominator of equation (8) may be zero, to avoid this, we add 1 to the denominator of equation (8). In step 9, j is

Algorithm 1 Preserving diversity by the distance between the objective vector corresponding to each individual and AEV

Input: The current population PO , The maximum number K of ranks about PO , and the number M of objective functions

Output: The distances of all individuals in PO

1. Initialise $k = 1$.

2. Compute the value of AEV $= (\xi_1, \xi_2, \dots, \xi_M)$.

3. **while** $k \leq K$ **do**

4. Let $I_1, I_2, \dots, I_{n_k}$ be all individuals at the rank k ;

5. Let $j = 1$.

6. **while** $j \leq M$ **do**

7. Compute the j th objective function values $\phi_j^1, \phi_j^2, \dots, \phi_j^{n_k}$ of all individuals $I_1, I_2, \dots, I_{n_k}$ at the rank k .

8. Compute the following distance from the j th objective function value of each individual I_i at the rank k to ξ_j

$$\text{dist}_j(I_i) = \frac{|\phi_j^i - \xi_j|}{\sum_{i=1}^{n_k} \phi_j^i + 1}; \quad (8)$$

9. Let $j = j + 1$.

10. **end while**

11. Compute the summation of all distances of I_i by the following expression at the rank k

$$\text{dist}(I_i) = \sum_{j=1}^M \text{dist}_j(I_i). \quad (9)$$

12. $k = k + 1$.

13. **end while**

assigned the value of $j+1$. When all distances of each individual at the rank k are computed, the summation of all distances of each individual is computed by the expression (9) in step 11. In fact, the summation is regarded as the distance between each individual at the rank k and the AEV, so it is referred to as the distance mentioned in the following text. In Fig. 2(a), we show an example that distinguishes between two different individuals at the same rank. At this example, we denote the distance between the individual A and AEV by d_1 , and the distance between the individual B and AEV by d_2 . Here, d_1 and d_2 are different so that A and B are distinguished. In Fig. 2(b), we show that the crowded distance between the individuals i is the sum of d' and d'' , where d' and d'' are the side lengths of the moment, respectively. The crowded distance between the individuals i is related to individuals $i-1$ and $i+1$ at the same rank. By Fig. 2(a) and 2(b), we observe that the proposed

distance has direction that AEV is used for guidance, however, the crowded distance has no direction for guidance. Therefore, at each iteration, the solutions found by using the proposed distance may be better than those found by using the crowded distance.

We analyze the computational complexity of Algorithm 1. The value of the AEV is computed when this algorithm is applied to the current population. Because the j -th component of the AEV is the minimum value of the j -th objective function, the sorting algorithm is used. Therefore, the computational complexity of the algorithm is $O(MN\log(N))$, where M and N is the number of objective functions and the size of the current population, respectively.

3.3 Fitness assignment

For the fitness assignment, we use the following method. For any two different individuals I_i and I_l , if the formula

$$\begin{aligned} &(\text{rank}(I_i) < \text{rank}(I_l)) \vee (\text{rank}(I_i) \\ &= \text{rank}(I_l) \wedge (\text{dist}(I_i) < \text{dist}(I_l))), \end{aligned}$$

is satisfied, then I_i is better than I_l . The above expression signifies that an individual at a lower rank is better than those at a higher rank, or an individual close to the AEV is better than those far away when their ranks are equal. For example, in Fig. 2(a), the individual C at the first rank is better than the individual A or B at the third rank. Because the individual B and A are at the third rank and the distance d_2 is less than the distance d_1 , the

individual B is better than the individual A .

3.4 Coding and decoding of crew pairings

There are two coding types that are a binary coding and a real coding based on row and column^[6], respectively, for solutions of the MOCPP in a GA. Because the MOCPP is formulated as a 0-1 integer programming formulation, we use binary coding. The length of a chromosome using binary coding represents the number of all feasible pairings. In addition, each gene in the chromosome represents a corresponding pairing whether it is selected. If a pairing is selected, its corresponding gene will become the value of 1. If a pairing is not selected, its corresponding gene will become the value of 0. Figure 3 shows an example of a chromosome containing eight genes (pairings).

3.5 Initial population

Zeren and Özkol^[4] proposed a method to generate an initial population that includes N feasible individuals. The method can find higher quality initial individuals. It means that if no gene covers a flight in an individual, then one of all pairings covering the flight is selected for assigning the gene. Thus, the number of deadheads is reduced by the method. Although N feasible individuals are created by the method, they are not necessarily optimal or sub-optimal.

3.6 Genetic operators

Because two different individuals in the population are always compared in the fitness

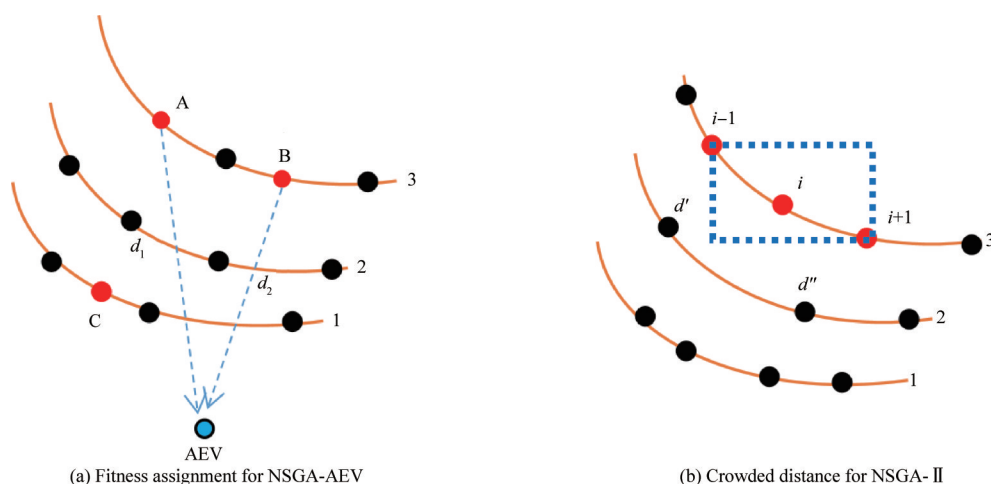


Fig. 2 Examples of fitness assignment for NSGA-AEV and crowded distance for NSGA-II

1	0	1	0	0	1	1	0
P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8

Fig. 3 An example of binary coding

assignment method, we use a binary tournament selection method with higher efficiency^[4]. However, other selection methods may be considered beyond two individuals at the population, such as, the front reservation method used in Chen et al.^[17]. For crew pairing problem, Park and Ryu^[22] proposed a greedy crossover method. Aydemir-Karadag et al.^[2] proposed a problem specific crossover procedure and an inner mutation. Çetin and Deveci^[5] used one point crossover and a bit-flip mutation approaches to generate a new offspring. Deveci and Çetin^[6] still used these approaches to generate a new offspring. Chen et al.^[17] used uniform crossover, partially matched crossover, reset mutation, and reverse mutation approaches for an integrated crew scheduling problem. In this paper, we use one point crossover approach that has better performance and a bit-flip mutation approach which avoids the algorithm falling into local search.

3.7 Repairing strategy

A feasible individual should include each flight in the set of all flights. However, the individuals in offspring generated by the crossover and mutation phases may not be ensured to be feasible. Beasley and Chu^[23] proposed a method to repair non-feasible individuals. Firstly, flights not included in an individual are identified. Secondly, the following expression is used to decide that pairings covering identified flights in the first step are added to the individual, so that non-feasible individuals are feasible

$$\frac{\text{the total pairing elapsed time}}{\text{number of non-included flights covered in the pairing}}. \quad (10)$$

The numerator of the expression (10) uses the cost of crew pairing as in Beasley and Chu^[23]. Every time period can have expenditures in a pairing, so we use the total pairing elapsed time to replace the cost of the pairing. Please refer to Beasley and

Chu^[23] for more details on the repairing strategy.

3.8 Local optimization strategy

To reduce redundant pairings for each individual, we use a local optimization strategy. The aim of the local optimization strategy is to judge whether an individual is feasible when the pairings included by the individual are removed from the individual. Please refer to Algorithm 5 described in Çetin and Deveci^[5] for more details on the local optimization strategy.

3.9 Detailed steps of NSGA-AEV

The parameters of NSGA-AEV are denoted as follows: population size N , crossover probability r , mutation probability u , and the maximum number G of generations. The pseudo-code of NSGA-AEV is given in Algorithm 2. Figure 4 shows the flowchart of NSGA-AEV.

Algorithm 2 NSGA-AEV

Input: $D^{(t)} = \{S_1^{(t)}, S_2^{(t)}, \dots, S_N^{(t)}\}$ is the set of all individuals (solutions) in generation t . $D^{(t)}$ is the offspring of $D^{(t)}$ during the iterative process.

Output: A set of all best individuals.

1. Let $t = 0$; initial population $D^{(t)}$ is generated.
 2. Apply non-dominated sorting and ranking method and the proposed preserving diversity method for $D^{(t)}$.
 3. **while** $t \leq G$ **do**
 4. Carry out selection, crossover, and mutation operations to generate the offspring $D^{(t)}$ on $D^{(t)}$.
 5. Apply the repairing strategy for non-feasible individuals in $D^{(t)}$.
 6. Apply the local optimization strategy for $D^{(t)}$.
 7. Apply non-dominated sorting and ranking method and the proposed preserving diversity method for $D^{(t)} \cup D^{(t)}$.
 8. Let $D^{(t+1)} = D^{(t)} \cup D^{(t)}$ be the replacement.
 9. Let $t = t + 1$.
 10. **end while**
-

4 Computational experiments

We use a real-world data set from 2021 China Graduate Mathematical Modeling Competition. The dataset includes 206 flights and a base. The convergence of NSGA-AEV is analyzed. The comparison of convergence of NSGA-AEV, NSGA-II and MMGA is discussed under the same

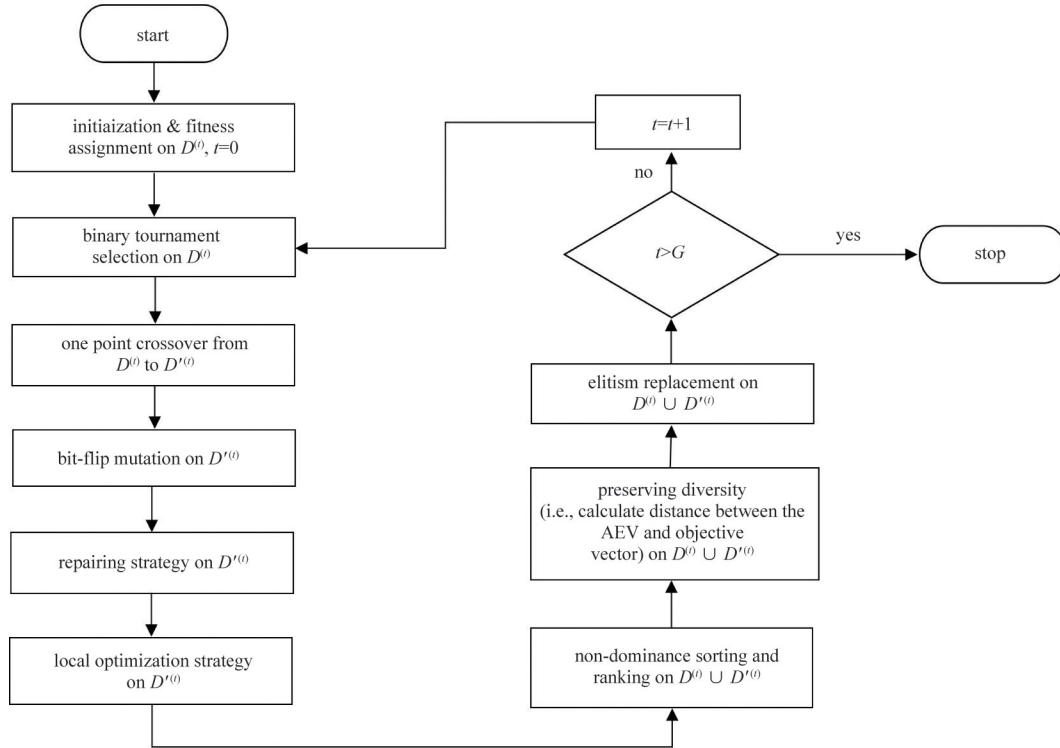


Fig. 4 Flowchart of NSGA-AEV

circumstances. The genetic parameters are set as follows: population size 50, crossover probability 0.9, mutation probability 0.2, and the maximum generation 1 000 (the total number of iteration is 1 001 from 0 to 1 000 generation) through repeated experiments. The specifications of the hardware environment and programming language are 13th Gen Intel (R) Core (TM) i9-13900K at 3.00 GHz and Python, respectively.

4.1 Convergence of NSGA-AEV

Figure 5 shows that the case of solutions derived by NSGA-AEV with the repairing strategy and without a repairing strategy is very different. With the repairing strategy, the solutions derived by NSGA-AEV are feasible from 0 to 1 000 generations. On the contrary, without a repairing strategy, we derive many non-feasible solutions, and the number of non-feasible solutions may increase in the iterative process. Therefore, for the MOCPP, if there is no the search direction of feasible solutions, this algorithm may not be convergent or derive many non-feasible solutions.

Figures 6 and 7 show the minimum sum of all objective function values and the case of the number

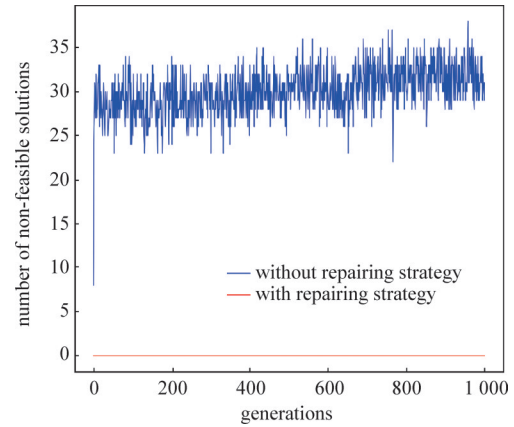


Fig. 5 Number of non-feasible solutions derived by NSGA-AEV with or without repairing strategy

of iterations reaching convergence in ten runs for NSGA-AEV with the local optimization (LOS) and without a local optimization strategy, respectively. The minimum sum of all objective function values means that the minimum sum of objective vector components is given at each iteration. The quality of solutions derived by NSGA-AEV with the local optimization strategy is higher than that of solutions derived by NSGA-AEV without a local optimization strategy in Fig. 6. On the other hand, in Fig. 7, we observe that the convergence rate of NSGA-AEV without a local optimization strategy is slower than

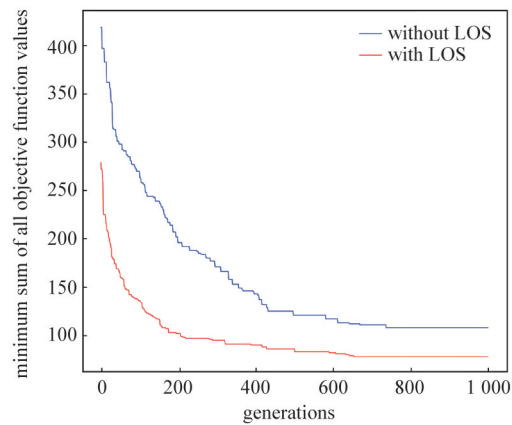


Fig. 6 Minimum sum of all objective function values of NSGA-AEV with or without local optimization strategy

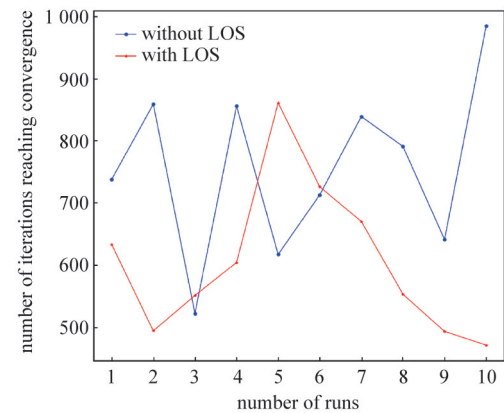


Fig. 7 Number of iterations reaching convergence of NSGA-AEV with or without local optimization strategy (in 10 runs)

that of NSGA-AEV with the local optimization strategy for most runs excluding runs 3, 5, and 6. Thus, if NSGA-AEV does not use the local optimization strategy, then it may fall into local Pareto solutions.

4.2 Comparison of convergence of NSGA-AEV, NSGA- II , and MMGA

We use the number of iterations reaching convergence (NIRC) and the performance measure function $C(X', X'')$ ^[13] for the comparison of convergence of NSGA-AEV, NSGA- II and MMGA.

Firstly, in Table 1, we see that the convergence rate of NSGA-AEV is faster than those of NSGA- II and MMGA in most runs. The main reason is that the search direction of NSGA-AEV is more concentrated than those of NSGA- II and MMGA. However, the average CPU time (i. e. runtime) of NSGA-AEV is longer than those of

NSGA- II and MMGA in Table 1. The main reason is that the CPU time spends computing AEV in the proposed preserving diversity algorithm of NSGA-AEV so that the CPU time of NSGA-AEV is increased.

Secondly, the function $C(X', X'')$ is defined as follows

C(X', X'') = (|{x'' in X'': exists x' in X', x' <= x''}|) / |X''|,

where X' and X'' are two sets of Pareto solutions and $x' \leq x''$ means that $x' = x''$ or $x' < x''$. The equation $C(X', X'') = 1$ means that the existing solutions in X' dominate or are equal to all solutions in X'' . On the contrary, the equation $C(X', X'') = 0$ means that non-existent solutions in X' dominate or are equal to any solution in X'' . Since $C(X', X'')$ and $1 - C(X'', X')$ may not be equal, we have to consider both $C(X', X'')$ and $C(X'', X')$. The

Table 1 Number of iterations reaching convergence and runtime of NSGA-AEV, NSGA- II , and MMGA, respectively, in 20 runs

run number	NSGA-AEV		NSGA- II		MMGA	
	NIRC	CPU/s	NIRC	CPU/s	NIRC	CPU/s
1	923	1 349.1	923	1 294.4	<u>672</u>	1 275.9
2	<u>633</u>	1 331.3	800	1 294.0	701	1 283.7
3	<u>494</u>	1 335.3	651	1 272.8	931	1 280.4
4	551	1 330.4	396	1 292.8	867	1 266.5
5	604	1 331.3	915	1 278.0	813	1 268.8
6	861	1 335.4	318	1 307.2	<u>814</u>	1 281.1
7	<u>726</u>	1 300.7	995	1 279.4	910	1 282.8
8	670	1 331.7	720	1 292.0	888	1 257.8
9	553	1 346.4	527	1 315.2	831	1 283.5
10	<u>493</u>	1 335.3	632	1 294.6	793	1 278.1
11	<u>471</u>	1 323.7	518	1 282.8	830	1 274.3
12	<u>352</u>	1 346.2	434	1 309.3	692	1 277.9
13	769	1 328.3	632	1 268.9	<u>534</u>	1 280.7
14	<u>566</u>	1 295.0	837	1 274.1	690	1 277.3
15	<u>562</u>	1 335.5	444	1 286.8	892	1 272.2
16	925	1 349.7	623	1 291.6	<u>666</u>	1 285.1
17	<u>735</u>	1 322.0	712	1 271.5	976	1 267.3
18	<u>586</u>	1 307.3	671	1 280.5	868	1 273.1
19	<u>458</u>	1 337.1	544	1 314.8	814	1 273.8
20	<u>685</u>	1 305.3	794	1 282.1	696	1 285.6
average		1 328.8		1 289.1		1 276.3

Note: The initial population of NSGA-AEV, NSGA- II , and MMGA are the same in each run.

qualities of solutions are exposed by the result analysis of $C(X', X'')$ and $C(X'', X')$. Mostly, the true Pareto-optimal set for a real-world problem is usually intractable. However, the function $C(X', X'')$ does not require a true Pareto-optimal set, so it is used to measure the solution quality for a real-world problem. Table 2 shows the results of performance measure by $C(X', X'')$ approach for NSGA-AEV and NSGA- II in 20 runs. The results indicate that the performance of NSGA-AEV outperforms that of NSGA- II for the MOCPP in most runs. In addition, Table 3 shows the results of performance measure by $C(X', X'')$ approach for NSGA-AEV and MMGA in 20 runs. The results indicate that the performance of NSGA-AEV outperforms that of MMGA for the MOCPP in most runs. In fact, we only derive a Pareto solution by NSGA-AEV or NSGA- II in each run, so the values of $C(X', X'')$ are 1 or 0. Although, at each run, we may derive multiple Pareto solutions by MMGA, the Pareto solution derived by NSGA-AEV can or can not dominate or

Table 2 Results of performance measure by $C(X', X'')$ approach for NSGA-AEV and NSGA- II				
run number	$C(X', X'')$	$C(X'', X')$	$\frac{C(X', X'')}{C(X'', X')} > 1$	$\frac{C(X'', X')}{C(X', X'')} > 1$
1	1	1		
2	1	0	True	
3	1	1		
4	1	0	True	
5	1	0	True	
6	1	0	True	
7	1	0	True	
8	1	1		
9	1	0	True	
10	0	1		True
11	0	1		True
12	1	1		
13	0	1		True
14	1	1		
15	1	0	True	
16	0	0		
17	1	0	True	
18	1	0	True	
19	1	0	True	
20	1	0	True	
total better runs			11	3

Note: X' and X'' are the Pareto-optimal sets obtained by the NSGA-AEV and NSGA- II with the same initial population, respectively.

Table 3 Results of performance measure by $C(X', X'')$ approach for NSGA-AEV and MMGA				
run number	$C(X', X'')$	$C(X'', X')$	$\frac{C(X', X'')}{C(X'', X')} > 1$	$\frac{C(X'', X')}{C(X', X'')} > 1$
1	0	1		True
2	1	0	True	
3	1	0	True	
4	1	0	True	
5	1	1		
6	1	0	True	
7	1	0	True	
8	0	0		
9	1	0	True	
10	0	1		True
11	1	0	True	
12	1	0	True	
13	0	1		True
14	1	0	True	
15	1	0	True	
16	0	1		True
17	1	0	True	
18	1	0	True	
19	0	1		True
20	1	0	True	
total better runs			13	5

Note: X' and X'' are the Pareto-optimal sets obtained by the NSGA-AEV and MMGA with the same initial population, respectively.

be dominated by all Pareto solutions by MMGA so that the values of $C(X', X'')$ are also 1 or 0.

5 Conclusion

The MOCPP has been solved in this paper. For the MOCPP, we consider some practical constraints and three objective functions which are the total number of duties, the total number of deadheads, and the total number of overnights. We propose two mathematical models and a novel algorithm, namely, NSGA-AEV to solve the MOCPP.

Finally, we indicate that NSGA-AEV not only explores high quality non-dominated solutions and improves the convergence by the repairing strategy and the local optimization strategy, but also performs better than NSGA- II and MMGA under performance indices.

The contribution of three aspects is included in this paper: 1) we propose a novel mathematical formulation, and an integer programming model that is based on all feasible pairings enumerated by a

depth-first search method; 2) we propose a novel algorithm NSGA-AEV that preserves the diversity of the population by the distance between the objective vector corresponding to each solution and an adaptive evaluation vector; 3) we use a repairing strategy and a local optimization strategy to explore high quality non-dominated solutions.

References

- [1] Wen X, Sun X T, Sun Y G, et al. Airline crew scheduling: models and algorithms[J]. *Transportation Research Part E: Logistics and Transportation Review*, 2021, 149: 102304. DOI: 10.1016/j.tre.2021.102304.
- [2] Aydemir-Karadag A, Dengiz B, Bolat A. Crew pairing optimization based on hybrid approaches[J]. *Computers & Industrial Engineering*, 2013, 65(1): 87-96. DOI: 10.1016/j.cie.2011.12.005.
- [3] Crawford B, Castro C, Monfroy E. A hybrid ant algorithm for the airline crew pairing problem[M]//*Lecture Notes in Computer Science*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006: 381-391. DOI:10.1007/11925231_36.
- [4] Zeren B, Özkol İ. An improved genetic algorithm for crew pairing optimization[J]. *Journal of Intelligent Learning Systems and Applications*, 2012, 4(1): 70-80. DOI: 10.4236/jilsa.2012.41007.
- [5] Çetin Demirel N, Deveci M. Novel search space updating heuristics-based genetic algorithm for optimizing medium-scale airline crew pairing problems[J]. *International Journal of Computational Intelligence Systems*, 2017, 10(1): 1082-1101. DOI: 10.2991/ijcis.2017.10.1.72.
- [6] Deveci M, Çetin Demirel N. Evolutionary algorithms for solving the airline crew pairing problem[J]. *Computers & Industrial Engineering*, 2018, 115: 389-406. DOI: 10.1016/j.cie.2017.11.022.
- [7] Srinivas N, Deb K. Multiobjective optimization using nondominated sorting in genetic algorithms[J]. *Evolutionary Computation*, 1994, 2(3): 221-248. DOI:10.1162/evco.1994.2.3.221.
- [8] Chen C H, Chou F I, Chou J H. Multiobjective evolutionary scheduling and rescheduling of integrated aircraft routing and crew pairing problems[J]. *IEEE Access*, 2020, 8: 35018-35030. DOI: 10.1109/ACCESS.2020.2974245.
- [9] Schaffer J D. Some experiments in machine learning using vector evaluated genetic algorithms[D]. Nashville, USA: Vanderbilt University, 1985.
- [10] Horn J, Nafpliotis N, Goldberg D E. A niched Pareto genetic algorithm for multiobjective optimization[C]//*Proceedings of the First IEEE Conference on Evolutionary Computation*. IEEE World Congress on Computational Intelligence. Orlando, FL, USA. IEEE, 1994: 82-87. DOI: 10.1109/ICEC.1994.350037.
- [11] Deb K, Pratap A, Agarwal S, et al. A fast and elitist multiobjective genetic algorithm: NSGA-II[J]. *IEEE Transactions on Evolutionary Computation*, 2002, 6(2): 182-197. DOI: 10.1109/4235.996017.
- [12] Fonseca C M, Fleming P J. Genetic algorithms for multiobjective optimization: formulation discussion and generalization[C]//*Proceedings of the 5th International Conference on Genetic Algorithms*. ACM, 1993: 416-423. DOI:10.5555/645513.657757.
- [13] Zitzler E, Thiele L. Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach[J]. *IEEE Transactions on Evolutionary Computation*, 1999, 3(4): 257-271. DOI:10.1109/4235.797969.
- [14] Li H, Zhang Q F. Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II[J]. *IEEE Transactions on Evolutionary Computation*, 2009, 13(2): 284-302. DOI: 10.1109/TEVC.2008.925798.
- [15] Deb K, Jain H. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints[J]. *IEEE Transactions on Evolutionary Computation*, 2014, 18(4): 577-601. DOI: 10.1109/TEVC.2013.2281535.
- [16] Zhang X Y, Tian Y, Cheng R, et al. A decision variable clustering-based evolutionary algorithm for large-scale many-objective optimization[J]. *IEEE Transactions on Evolutionary Computation*, 2018, 22(1): 97-112. DOI: 10.1109/TEVC.2016.2600642.
- [17] Chen C H, Liu T K, Chou J H. Integrated short-haul airline crew scheduling using multiobjective optimization genetic algorithms[J]. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2013, 43(5): 1077-1090. DOI:10.1109/TSMC.2012.2234943.
- [18] Chou T Y, Liu T K, Lee C N, et al. Method of inequality-based multiobjective genetic algorithm for domestic daily aircraft routing[J]. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 2008, 38(2): 299-308. DOI: 10.1109/TSMCA.2007.914784.
- [19] Haouari M, Mansour F Z, Sherali H D. A new compact formulation for the daily crew pairing problem[J]. *Transportation Science*, 2019, 53(3): 811-828. DOI: 10.1287/trsc.2018.0860.
- [20] Du X B. Research on Integrated Optimization of Airline Crew Scheduling[D]. Beijing: Beijing Jiaotong University, 2020.
- [21] Liu T K, Chen C H, Chou J H. Optimization of short-haul aircraft schedule recovery problems using a hybrid multiobjective genetic algorithm[J]. *Expert Systems with Applications*, 2010, 37(3): 2307-2315. DOI: 10.1016/j.eswa.2009.07.068.
- [22] Park T, Ryu K R. Crew pairing optimization by a genetic algorithm with unexpressed genes[J]. *Journal of Intelligent Manufacturing*, 2006, 17(4): 375-383. DOI: 10.1007/s10845-005-0011-z.
- [23] Beasley J E, Chu P C. A genetic algorithm for the set covering problem[J]. *European Journal of Operational Research*, 1996, 94(2): 392-404. DOI: 10.1016/0377-2217(95)00159-X.