

Received 15 September 2022, accepted 3 October 2022, date of publication 10 October 2022, date of current version 14 October 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3213066

RESEARCH ARTICLE

An Improved Honey Badger Algorithm by Genetic Algorithm and Levy Flight Distribution for Solving Airline Crew Rostering Problem

BIN DENG 

School of Mathematics and Statistics, Yunnan University, Kunming 650504, China

e-mail: dengbin96@126.com

This work was supported by the Postgraduate Research and Innovation Foundation of Yunnan University under Grant 2021Z089.

ABSTRACT Airline crew rostering problem contains a variety of rules and constraints, and there are almost countless possible scheduling schemes. It is the most complex and important link in the entire crew scheduling plan. In this paper, we build a model that includes qualification constraints. In this paper, we consider two models with qualification constraints with different objective functions, namely minimizing the total cost of the airline and balancing flight utility among pilots as much as possible. To solve this model, the Levy flight is used to improve the ability of the Honey Badger Algorithm (HBA) to jump out of local optima, and the crossover and mutation operators in the Genetic Algorithm (GA) are used to improve the quality of the solution. This improved HBA algorithm significantly improves convergence and solution accuracy. In addition to this, we verified the improved HBA algorithm on 6 instances, of which 4 instances do not contain any qualifications, and 2 instances contain high-qualification flight pairings. The good results of the improved HBA show that it has excellent performance in both objective functions.

INDEX TERMS Airline crew rostering problem, honey badger algorithm, genetic algorithm, levy flight.

I. INTRODUCTION

According to the statistics of major airlines in the past, the cost of crew is the second largest airline cost after fuel cost [1]. But the fuel cost is usually fixed, and the cost of the crew is largely manageable [2], [3], [4]. Table 1 lists pilot costs (annual crew fees, salaries, and benefits) for some of China's major airlines in 2019. Thus, the reasonable arrangement of the crew is very important for the airlines. In the past few decades, the problem of crew scheduling has attracted extensive research in both theory and practice, because a good scheduling scheme can save airlines hundreds of millions of dollars each year. However, the crew scheduling problem is NP-hard, which means that we cannot find the optimal solution to the problem in polynomial time [6]. Due to the large scale of flights involved and the need to strictly comply with complex work rules, in order to reduce the difficulty of solving, the crew scheduling problem can usually be divided into crew pairing

problem (CPP) and crew rostering problem (CRP) to be solved in turn. CPP constructs a set of flight loops that can cover all flight segments and depart from one base and return to the same base, while CRP assigns the constructed flight loops to each crew member to generate a personal schedule. In some surveys [4], [7], the contents of these two parts have been presented in detail. In this paper, we mainly study CRP, because CRP affects the fairness of the crew. The purpose of this study is to generate a set of low-cost and highly fair schedules.

CRP has been a problem that has puzzled researchers in the industry for many years. In combinatorial optimization, CRP is also divided into NP-hard problems [8]. This problem can be described as orderly assigning all flight pairings to pilots on the basis of satisfying laws and regulations and optimizing the objective function. The objective function of the airline crew roster problem usually has the following types: 1. Minimize the cost of the airline. 2. The income balance among pilots. 3. Minimize the amount of fatigue between pilots. In the method of solving non-deterministic polynomial (NP)-hard problems, some intelligent heuristic


The associate editor coordinating the review of this manuscript and approving it for publication was Shun-Feng Su .

TABLE 1. Cabin crew costs for major Chinese airlines in 2019 [5].

Airline	Number of captain	Number of co-pilot	Pilot expenses(billion)
China Eastern	2,483	2,343	2.55
Air China	2,423	2,749	2.6
China Southern Airlines	3,376	3,003	3.42
Sichuan Airlines	826	1,023	1.15
XIAMEN AIR	849	973	1.16
Hainan Airlines	1,110	1,779	1.68
Spring Airlines	489	580	0.97
Juneyao Airlines	410	503	0.87

algorithms are widely used, such as genetic algorithm and particle swarm algorithm, etc. In the existing studies, most of the literature assumes that the assignment of flight pairings to pilots is not constrained by qualifications. However, in real life, the influence of qualifications is very large. For example, if a pilot lacks a certain qualification, he cannot assign the flight pairings that require this qualification to this pilot.

Therefore, this paper proposes an improved honeypot optimization algorithm, that is, in the process of updating the solution, adding Levy flight, and then using the two algorithms in the genetic algorithm to further improve the quality of the solution. This reduces the chance of getting stuck in a local optimum during the solution. In addition to this, we have developed a mathematical model that takes into account the qualification issue. Compared with other scholars' research, this paper has the following contributions: 1. A mathematical model that is more suitable for the real world is proposed. 2. In order to solve CRP, we propose an improved HBA algorithm, which reduces the possibility of the algorithm falling into local optimum. At the same time, a discrete version of the HBA algorithm was also designed. 3. Verified the effectiveness and feasibility of the improved HBA algorithm in solving large-scale CRP.

A. RELATED WORK

In recent years, many scholars have studied CRP. In the following, we will review the literature in terms of mathematical models and solution methodology.

Freling et al. [9] formulated CRP as an ensemble partition model to minimize the total cost of airlines, and employed a branch-and-bound algorithm to solve a monthly instance. Gamache et al. [10] formulate CRP as an ensemble partition model to minimize total airline costs, and employ a branch-and-bound algorithm to solve a monthly instance crew scheduling problem. The crew scheduling problem of Kasirzadeh et al. [11] is formulated as a set coverage problem and solved with a column-generating algorithm on a real dataset. Medard and Sawhney [12] introduced the crew scheduling problem and the crew recovery problem, in order to solve the crew recovery problem, they proposed a new method based on tree search and column generation algorithm. Based on European airlines, Nissen et al. [13] formulated a new mathematical model for the airline crew rearrangement problem, which covered various laws and regulations in a resource-constrained manner. Then the model

is solved by branch pricing, and the calculation results show that a new solution can be quickly provided when the scheduling is disturbed. Saddoune et al. [14] integrated the crew scheduling problem, that is, the crew scheduling problem is no longer divided into two stages of CPP and CRP to solve. To obtain shorter computation time, they propose a dual dynamic constrained aggregation method. This approach not only improves the solution quality of the scheme, but also reduces the average computation time by a factor of 2.3. Souai and Teghem [16] established a model and two scheduling networks using the real data of an airline in Taiwan, and solved this problem by using a column generation algorithm. Doi et al. [17] proposed a CRP based on fair working hours, and in order to satisfy various hard constraints, a two-level decomposition algorithm was proposed. The calculation results show that this method can effectively obtain the solution of this problem.

In the above studies, the column generation algorithm or the derivative algorithm of the column generation algorithm is used to solve the problem. However, in addition to column generation algorithms, some intelligent optimization algorithms are also widely used when solving CRP. Compared with the column generation algorithm, the intelligent optimization algorithm has the characteristics of fast solution speed. Souai and Teghem [16] described the disadvantages of dividing the crew scheduling problem into two sub-problems, CPP and CRP, therefore, they proposed 3 heuristics based on genetic algorithms to solve these two sub-problems simultaneously. Considering both CPP and CRP, Saemi et al. [18] proposed a new formula and designed a heuristic algorithm based on ant colony optimization to solve it. The calculation results show that the ant colony optimization algorithm has a good effect in solving the problem of synthesizing CPP and CRP.

In the previous studies, all focus on the single-target situation. Most of them set the objective function to minimize the airline's expenditure, and a few set the objective function to consider the balance of working hours. In recent years, some scholars have begun to turn their research direction to multi-objective situations. In CRP, Zhou et al. [19] considered both the workload of the crew and the satisfaction of the crew, and proposed a more useful multi-objective model. In order to solve this multi-objective model, a multi-objective ant colony algorithm was proposed to effectively solve the problem. Chutima and Arayikanon [20] optimized four objectives at

the same time. In order to solve this model, a hybrid algorithm of multiobjective evolutionary algorithm based on decomposition(MOEA/D) and honey-bees mating optimization algorithm (HBMO) was proposed. The calculation results show that the hybrid algorithm outperforms these two algorithms.

It is clear from the literature review above that the CRP problem has received extensive attention from researchers over the past two decades. But only a few articles directly address this issue. Because the captain and co-pilot not only need to depend on basic laws and regulations, but also need to consider their various qualifications and matching, etc. For example, the qualification of the captain in charge is C1, only the co-pilot with C1 qualification can match with it.

As far as the objective function is concerned, we can know that the objective function can be divided into single objective and multi-objective. As far as constraints are concerned, most literatures only consider some of the most basic constraints, while a small number of literatures consider constraints such as language. However, in this paper, we further consider the qualification constraint, which is an essential part of real world airlines. In fact, most constraints can be described as qualification constraints, such as language constraints, etc.

Therefore, in this paper, we propose a model that is closer to the real world. To solve this model, we propose an improved honey badger algorithm by genetic algorithm and levy flight distribution and we call this improved algorithm the Honey Badger Algorithm-Genetic Algorithm(HBA-GA). And for CRP, we also designed a discrete version of this hybrid algorithm. In addition, we conduct a comparative analysis of the proposed hybrid algorithm and various other well-known metaheuristics and hybrid algorithms on several different datasets. Simulation experiments show that HBA-GA is an effective hybrid heuristic algorithm for solving CRP.

II. AIRLINE CREW ROSTERING PROBLEM

In our research problem, the aim is to assign pilots various training tasks while identifying flight tasks while minimizing cost and maximizing fairness among pilots. Some important features of CRP are as follows: (1) The pilot's departure station and end station should be at the same airport; (2) If a flight pairing is assigned to a pilot, the pilot cannot assign other tasks during this time period.

A. RELATED DEFINITIONS

To make it easier for those who are less familiar with the aviation industry, we first introduce some related terms.

Time: Airline operations span time and space. The time expression in this article consists of the year, month, day, hour and minute, and the precision below the second is not counted.

Airport: The departure and arrival of flights, as well as the departure and arrival of crew members, are all based on the airport. Two flight loops can only be assigned to the same pilot if their landing and departure airports are the same.

Crew: The research in this paper is only for pilots, but is fully applicable to flight attendants and marshals. Piloting of modern civil aircraft usually requires two pilot qualifications: captain and co-pilot. Each crew pilot has the following attributes:

1. Each crew member has a fixed base, expressed by the airport.
2. Each crew member has one and only one primary qualification, but may also have other alternate qualifications.

Flight: Refers to a takeoff and landing of an aircraft. Also called a leg when the flight applies to crew scheduling. Each flight includes the following attributes:

1. Every flight has a given departure time, and an arrival time.
2. Every flight has a given departure and arrival airport.
3. Each flight has a given minimum crew composition.

Duty: A duty consists of a series of leg (flight or boarding) and connecting times. A duty is shown in Figure 1, where A1, A2, A3 and A4 represent 4 airports respectively, and 6142, 6143, 4133 and 4134 represent 4 flights. These 4 flights and the interval between flights together form a duty. The following relationships are satisfied between legs on the same duty:

1. The arrival of the previous leg is the same as the departure airport of the next leg.
2. The interval between two adjacent flight segments must satisfy the shortest connection time constraint.
3. Each leg in the same duty must take off on the same day, but the arrival time is not limited by this.
4. Connection time between legs counts toward duty time, but not flight time. The start time of duty is calculated from the departure time of the first leg performed on the day, and the end time is calculated according to the arrival time of the last leg.

Pairing: A pairing consists of a sequence of duty and rest periods, starting from and eventually returning to one's own base.

Roster: Each crew has a roster in each scheduling cycle, which consists of a series of pairings and vacations, and a certain number of vacation days are met between the pairings.

B. OBJECTIVES AND CONSTRAINTS

Each flight pairing has start time, end time, duty time, flight time and duration, etc. In airlines, there are often a variety of pilots with different qualifications, and pilots with different qualifications can fly different flight pairings. In this section, we build a model that takes into account the following constraints:

1. In a scheduling cycle, the flight time and duty time of each pilot shall not exceed the maximum flight time and maximum duty time stipulated by the Civil Aviation Administration.
2. The interval between all flight pairings performed by the pilot is greater than the minimum rest period.
3. One captain and one co-pilot must be assigned to each flight loop.

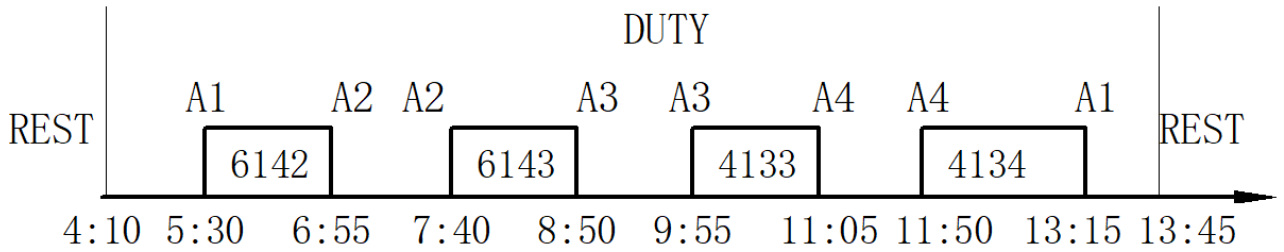


FIGURE 1. Pilot's day schedule. Duty time=7h45min. Flight time = 5h10min.

4. A flight pairing can only be assigned to a pilot if the pilot's qualifications meet the requirements of the flight pairing.

We first define the sets, parameters and decision variables that are needed in mathematical programming. The detailed symbol definitions are shown in Table 2.

1) OBJECTIVE

In this paper, we consider two objectives and will solve for each of them separately. Although there are many goals that airlines focus on, these two goals are the most concerned by airline decision makers. The detailed explanation and mathematical expression of these two goals can be described as follows.

Objective 1: Balance of benefits between pilots of the same qualification. This goal is mainly to minimize the salary difference between pilots, so as to prevent pilots' salaries from being divided into two levels and causing pilots to change jobs. Therefore, the objective function 2 can be formulated as the following:

$$\min f_1 = \sqrt{\frac{\sum_{i \in R_1} (v_i^* - v_{g1(i)})^2 + \sum_{j \in R_2} (v_j^* - v_{g2(j)})^2}{|R_1| + |R_2|}},$$

where $v_i^* = \sum_{k \in P} v_{ik}x_{ik}$ and $v_j^* = \sum_{k \in P} v_{jk}y_{jk}$. In the above formula, v_i^* and v_j^* are the total revenue in one scheduling period.

Objective 2: Minimize airline costs. Of all costs for an airline, in addition to fixed costs, it is often possible to change the overall cost of crew duty by adjusting the allocation of flight pairings. Therefore, the objective function 1 can be formulated as the following:

$$\min f_2 = \sum_{i \in R_1} \sum_{k \in P} c_{ik}x_{ik} + \sum_{j \in R_2} \sum_{k \in P} c_{jk}y_{jk}.$$

2) CONSTRAINT

Constraint 1: Any flight pairing requires a captain and a co-pilot.

$$a_{ij}(\sum_{i \in R_1} x_{ik} + \sum_{j \in R_2} y_{jk}) = 2, \quad \forall k \in P. \quad (1)$$

Constraint 2: In one period, the flight time of the captain and co-pilot does not exceed T_1 .

$$\sum_{k \in P} f_k x_{ik} \leq T_1, \quad \forall i \in R_1 \quad (2)$$

$$\sum_{k \in P} f_k y_{jk} \leq T_1, \quad \forall j \in R_2. \quad (3)$$

Constraint 3: In one period, the captain and co-pilot's duty time shall not exceed T_2 .

$$\sum_{k \in P} d_k x_{ik} \leq T_2, \quad \forall i \in R_1 \quad (4)$$

$$\sum_{k \in P} d_k y_{jk} \leq T_2, \quad \forall j \in R_2. \quad (5)$$

Constraint 4: The qualifications of the captain and co-pilot are not lower than the minimum requirements of the flight pairing.

$$g_1(i)x_{ik} - g_2(k) \geq 0, \quad \forall i \in R_1, k \in P \quad (6)$$

$$g_1(j)y_{jk} - g_2(k) \geq 0, \quad \forall j \in R_2, k \in P. \quad (7)$$

Constraint 5: Each pilot can only perform one pairing per day.

$$w_d(k_1)x_{ik_1} + w_d(k_2)x_{ik_2} \leq 1, \quad \forall i \in R_1, \\ k_1 \neq k_2 \in P, \quad d = 1, 2, \dots, D. \quad (8)$$

$$w_d(k_1)y_{jk_1} + w_d(k_2)y_{jk_2} \leq 1, \quad \forall j \in R_2, \\ k_1 \neq k_2 \in P, \quad d = 1, 2, \dots, D. \quad (9)$$

Constraint 6: In order to ensure that the pilot has plenty of energy and reduce flight accidents. In any 144-hour period, at least 48 hours of rest is scheduled for pilots. At the same time, 2 days off must be arranged after 4 consecutive days of flying.

$$4 - w_{d+4}(k_1)x_{ik_1} - w_{d+5}(k_2)x_{ik_2} \geq w_d(k_3)x_{ik_3} \\ + w_{d+1}(k_4)x_{ik_4} + w_{d+2}(k_5)x_{ik_5} + w_{d+3}(k_6)x_{ik_6}, \\ \forall i \in R_1, k_1, k_2, k_3, k_4, k_5, k_6 \in P, \quad d = 1, 2, \dots, D. \quad (10)$$

$$4 - w_{d+4}(k_1)y_{jk_1} - w_{d+5}(k_2)y_{jk_2} \geq w_d(k_3)y_{jk_3} \\ + w_{d+1}(k_4)y_{jk_4} + w_{d+2}(k_5)y_{jk_5} + w_{d+3}(k_6)y_{jk_6}, \\ \forall j \in R_2, k_1, k_2, k_3, k_4, k_5, k_6 \in P, \quad d = 1, 2, \dots, D. \quad (11)$$

TABLE 2. Summary of notation.

Type of notation	Notation	Description
Set	P	Set of all pairings.
	R_1	Set of all captains.
Constant	R_2	Set of all co-pilots.
	c_{ik}	Cost of assigning pairing k to captain $i, \forall i \in R_1, k \in P$.
	c_{jk}	Cost of assigning pairing k to co-pilot $j, \forall j \in R_2, k \in P$.
	v_{ik}	Profit of captain i after assigning pairing k to captain $i, \forall i \in R_1, k \in P$.
	v_{jk}	Profit of co-pilot j after assigning pairing k to co-pilot $j, \forall j \in R_2, k \in P$.
	\bar{v}_t	Average profit of pilots with qualification $t, t = g_1(i)$ or $t = g_1(j)$.
	f_k (minutes)	Flight time of pairing $k, \forall k \in P$.
	d_k (minutes)	Duty time of pairing $k, \forall k \in P$.
	T_1 (minutes)	Maximum flight time in a schedule cycle.
	T_2 (minutes)	Maximum duty time in a schedule cycle.
Variable	$g_1(i) \& g_1(j)$	Qualification of captain $i \in R_1$ or co-pilot $j \in R_2$.
	$g_2(k)$	Minimum qualifications required for pairings k .
	a_{ij}	$a_{ij} = 1$ if captain i can be paired with co-pilot $j, 0$ otherwise.
	d	Day index in the period, $d = 1, 2, \dots, D$.
	$w_d(k)$	$w_d(k) = 1$ if day d is included in pairing $k, 0$ otherwise.
	x_{ik}	$x_{ik} = 1$ if pairing k is assigned to captain $i, 0$ otherwise.
	y_{jk}	$y_{jk} = 1$ if pairing k is assigned to co-pilot $j, 0$ otherwise.

In Equation 10 and Equation 11, when d is the last days in a period, it can be connected to the next period

III. BACKGROUND FOR HONEY BADGER ALGORITHM AND LEVY FLIGHT

We discuss mathematical notation and formulas for honey badger algorithm (HBA) [21] and levy flight in this section.

A. HONEY BADGER ALGORITHM

Honey badger algorithm (HBA) is a new intelligent optimization algorithm developed by Egyptian scholars Hashim et al. [21] in 2021 based on the foraging behavior of honeypots. The algorithm simulates two foraging behaviors of honeypots: either digging or following the honeypot, the first is digging mode, and the second is honey-collecting mode. In digging mode, the honeypot uses its sense of smell to locate the food, and then selects a suitable location to catch the prey. In honey-picking mode, the honeypot finds food based on the location of the guided honeypot.

1) INSPIRATION

The main inspiration of the honey badger algorithm comes from the foraging behavior of honey badger. Like other swarm intelligence algorithms, HBA is also divided into 3 stages, namely initialization stage, digging phase and honey phase. The following subsections describe the impact of these two behaviors of the honeypot on the HBA optimizer.

2) INITIALIZATION STAGE

HBA randomly generates an initial solution in the initialization phase.

$$X = \begin{bmatrix} x_{1,1} & \dots & x_{1,n-1} & x_{1,n} \\ \dots & \dots & \dots & \dots \\ x_{m-1,1} & \dots & x_{m-1,n-1} & x_{m-1,n} \\ x_{m,1} & \dots & x_{m,n-1} & x_{m,n} \end{bmatrix} \quad (12)$$

where m is the number of candidate solutions and n is the number of decision variables.

In addition to this, the odor intensity of the prey needs to be defined. The higher the odor intensity of the prey, the faster the movement speed. The odor intensity can be expressed by the following formula:

$$I_i = r_1 \cdot \frac{(x_i - x_{i+1})^2}{4\pi(x_{prey} - x_i)^2}$$

where r_1 is a random number between 0 and 1 and x_{prey} is the current optimal position. To ensure a smooth transition from exploration to exploitation. Use the formula below to define a decreasing factor α that decreases with the number of iterations to decrease randomization over time.

$$\alpha = C \cdot e^{-\frac{t}{T}}$$

where t is the current number of iterations, T is the maximum number of iterations, and C is a constant greater than 1.

3) DIGGING PHASE

During the mining phase, the honeypot moves towards the food in a heart-shaped path, which can be expressed by the following formula:

$$x_{i,j}(t+1) = x_{prey,j} + F \cdot \beta \cdot I \cdot x_{prey,j} + F \cdot r_2 \cdot \alpha \cdot (x_{prey,j} - x_{i,j}) \cdot |\cos(2\pi r_3)| \cdot (1 - \cos(2\pi r_4)) \quad (13)$$

where r_2, r_3 and r_4 are three random numbers between 0 and 1, and $\beta \geq 1$ is the ability of the honeypot to get prey. F represents the search direction, which is determined by the following formula:

$$F = \begin{cases} 1, & r_5 \leq 0.5 \\ -1, & otherwise \end{cases}$$

where r_5 is a random number between 0 and 1. During the digging phase, the location of the honeypot depends on the strength of the smell and the search impact factor α .

4) HONEY PHASE

The behavior of the honeypot during the honey phase can be simulated by the following formula:

$$x_{i,j}(t + 1) = x_{prey,j}(t) + F \cdot r_6 \cdot \alpha \cdot (x_{prey,j} - x_{i,j}) \quad (14)$$

where r_6 is a random number between 0 and 1. As can be seen from the above formula, the location of the new honeypot is near x_{prey} .

5) PSEUDO-CODE FOR STANDARD HBA

The optimization of HBA starts from randomly generating an initial solution X . In each iteration, the digging phase is used as the exploitation phase of the algorithm, and the honey phase is used as the exploration phase of the algorithm. When $r < 0.5$, the algorithm approaches the optimal solution through Eq. 13, and when $r \geq 0.5$, the algorithm converges toward the optimal solution through Eq. 14, where r is a random number between 0 and 1. Finally, when the stopping condition of HBA is satisfied, the algorithm will stop and find the current best solution. The pseudo-code of the standard HBA is shown in Algorithm 1.

B. LEVY FLIGHT DISTRIBUTION

The Levy distribution is a continuous probability distribution of a non-negative random variable. A Levy flight is a random walk with a Levy distribution for the step size, which is heavy-tailed. When defined as walking in a space of dimension greater than 1, the steps performed are isotropic random directions. Therefore, to prevent the population from getting stuck in a local optimum, we can use Levy flight to change the position of the solution. Eq.15 describes changing the position of the solution by Levy flight,

$$x_{new}(t + 1) = \begin{cases} x_{prey}(t) + Levy(D) \cdot (UB - LB), & \text{if } rand < 0.25 \\ x_{prey}(t) + Levy(D) \cdot (x_{prey}(t) - x_{i,j}), & \text{if } rand < 0.5 \\ x_{prey}(t) \cdot Levy(D), & \text{if } rand < 0.75 \\ x_{prey}(t) + Levy(D), & \text{otherwise} \end{cases} \quad (15)$$

$$Levy(D) = 0.01 \cdot \frac{rand1 \cdot \theta}{|rand2|^{\frac{1}{\beta}}},$$

where $rand1$ and $rand2$ are random numbers between 0 and 1. UB and LB are the upper and lower bounds, respectively. θ can be calculated by the Eq.16

$$\theta = \frac{\Gamma(1 + \beta) \cdot \sin(\frac{\pi \cdot \beta}{2})}{\Gamma(\frac{1+\beta}{2}) \cdot \beta \cdot 2^{\frac{\beta-1}{2}}}, \quad (16)$$

where Γ is the probability distribution function and β is a fixed constant of 1.5.

IV. THE PROPOSED HBA-GA FOR CRP

Although many population-based metaheuristics have emerged in recent years and have performed well in many

Algorithm 1 Standard HBA Algorithm

```

1: Initialize the parameters  $\beta$ ,  $C$  of the HBA algorithm, the
   population size is  $M$ , and the dimension of the solution
   is  $N$ ;
2: Randomly initialize a solution  $X = \{x_1, \dots, x_M\}$ .
3: Initialize the maximum number of iterations  $T$ .
4: while  $t < T$  do
5:   Calculate the fitness of the solution set  $X$ .
6:   Update density factor  $\alpha$ .
7:   for  $i = 1$  to  $M$  do
8:     Produces random numbers  $r, r_1, r_2, r_3, r_4, r_5$  and  $r_6$ 
       between 0 and 1.
9:     if  $r < 0.5$  then
10:      for  $j = 1$  to  $N$  do
11:        Update the position:  $x_{new,j} = x_{prey,j} + F \cdot \beta \cdot I_i \cdot$ 
           $x_{prey,j} + F \cdot r_2 \cdot \alpha \cdot (x_{prey,j} - x_{i,j}) \cdot | \cos(2\pi r_3) \cdot$ 
           $(1 - \cos(2\pi r_4)) |$ 
12:      end for
13:     else
14:      for  $j = 1$  to  $N$  do
15:        Update the position:  $x_{new,j} = x_{prey,j} + F \cdot r_6 \cdot$ 
           $\alpha \cdot (x_{prey,j} - x_{i,j})$ 
16:      end for
17:     end if
18:   end for
19:   Calculate the fitness  $f_{new}$  of the solution  $x_{new}$ .
20:   if  $f_{new} < f_i$  then
21:      $x_i = x_{new}, f_i = f_{new}$ 
22:   end if
23:   if  $f_{new} < f_{prey}$  then
24:      $x_{prey} = x_{new}, f_{prey} = f_{new}$ 
25:   end if
26: end while
27: Returns the best solution  $x_{prey}$ .
```

continuous optimizations, a single algorithm does not perform well in large-scale discrete optimization problems. Therefore, we need to improve a single algorithm to achieve a balance between solution efficiency and solution quality. Therefore, this paper proposes an improved honey badger algorithm.

This section introduces a novel optimization algorithm, which we call HBA-GA, as shown in Fig. 2. The proposed algorithm is based on HBA, which combines several operators in GA and the Levy flight method. Our proposed HBA-GA method improves the ability to jump out of local optima when dealing with complex engineering problems. In HBA-GA, two search methods are added. First, several operators in the GA algorithm are used to speed up the convergence speed of the algorithm. Second, adding Levy flight increases the search space of the algorithm. Below we describe the proposed HBA-GA method in detail.

A. ENCODING AND DECODING

In any evolutionary algorithm, one of the first things we need to do is code the problem. A sensible coding method can make the problem much simpler. In CRP, each pilot has a name or number, however, since these numbers are all in the form of strings, we cannot use these numbers directly for encoding. Therefore, we need to convert these strings into real numbers that can participate in operations. We put all pilots into a list, and the position of each list represents a pilot. For example, we put 4 pilots into a list [A001, A002, A003, A004], and then we can get the corresponding pilot's code $1 \leftarrow A001$, $2 \leftarrow A002$, $3 \leftarrow A003$ and $4 \leftarrow A004$. Since the number of each pilot is unique, we can easily establish a one-to-one correspondence between the number and the real number. After establishing a one-to-one correspondence, we can get the encoding method of the candidate solution. Suppose we have two flight pairings, two captains [A001, A002] and two co-pilots [A003, A004], at this time, a solution that can get a candidate solution is $[[1, 3], [2, 4]]$. Each position corresponds to a captain and a co-pilot, which means that a flight pairing is assigned a captain and a co-pilot.

B. DISCRETISATION OF HBA

As can be seen from the detailed description of the standard HBA and Levy flight, the two algorithms can be directly applied only when solving continuous optimization during the update process of the solution. In other words, the CRP solved in this paper is a discrete optimization problem, and these two algorithms cannot be applied to CRP. Therefore, we also need to develop a discrete version for these two algorithms.

1) GENERATE A NEW UNFEASIBLE SOLUTION

In HBA, for best solution $x_{prey}(t)$ in the population, we can generate a new solution $x_{new}(t + 1)$ according to Eq. 13 or Eq. 14.

For example, we are given solutions $x_1 = [[1, 1], [2, 2], [3, 3], [4, 4]]$, $x_2 = [[2, 1], [1, 2], [4, 3], [3, 4]]$, $x_3 = [[3, 2], [1, 4], [4, 3], [2, 1]]$ with population size 3. A position in each solution represents a pairing, and each pairing corresponds to a captain and a co-pilot. And we assume that all pairings are compatible in time. Let $F = 1$, $r_6 = 0.5$, $C = 2$, $t = 10$ and $T = 20$ in HBA. For this example, we assume $x_{prey} = x_1$. Then, using the Eq. 14, we can get $x_{new}^{HBA}(t + 1) = [[1.6, 1.6], [3.2, 3.2], [4.8, 4.8], [6.4, 6.4]]$.

Although a new solution can be generated in this way, the new solution may not be a feasible solution due to various constraints of the CRP problem. Therefore, we can take a series of further steps to make it feasible.

2) REPAIR INFEASIBLE SOLUTIONS

We can convert an infeasible solution to a feasible solution by following the steps below:

STEP 1: Round down each position of the infeasible solution to produce a vector of integers.

STEP 2: For each element that is out of bounds, we take $x_{i,j} = x_{i,j} \bmod UB$ and the elements that are not out of bounds to form a new vector, where UB is the upper bound of the j -th position.

STEP 3: For the resulting integer vector, keep the element at the first incompatible position, and set the element at other positions to 0.

STEP 4: For each 0 position, find the element that is unassigned and that occurs most frequently in the population. If such an element exists, assign the element to this position, otherwise skip this step.

STEP 5: For the remaining positions with a value of 0, randomly assign unassigned elements to these positions.

In the above example, the solution produced by HBA is $x_{new}^{HBA}(t + 1) = [[1.6, 1.6], [3.2, 3.2], [4.8, 4.8], [6.4, 6.4]]$. Because none of the elements in the vector are integers, we need to convert the infeasible solution into a feasible solution. In step 1, we round down the elements at all positions to get $x_{new}^{HBA}(t + 1) = [[1, 1], [3, 3], [4, 4], [6, 6]]$. In the step 2, the vectors become $x_{new}^{HBA}(t + 1) = [[1, 1], [3, 3], [4, 4], [2, 2]]$. In the step 3, the vectors become $x_{new}^{HBA}(t + 1) = [[1, 1], [3, 3], [4, 4], [2, 2]]$. In the last step, the vectors become $x_{new}^{HBA}(t + 1) = [[1, 1], [3, 3], [4, 4], [2, 2]]$.

C. CROSSOVER AND MUTATION OPERATORS

In genetic algorithm, crossover operator and mutation operator are two important operators in individual evolution. In the process of breeding the next generation, the exchange of genes in the same position of two different individuals is called the crossover operator. The transformation of a gene at a certain position in a single individual is called a mutation operator. In our proposed HBA-GA method, we will use these two operators to improve the candidate solutions. However, not all individuals will crossover and mutate, but the worst 10% individuals will crossover and mutate with a certain probability. In this process, we set the probability of crossover to 0.8 and the probability of mutation to 0.01. Suppose the father of the two operators selected is $father = [[1, 4], [2, 5], [3, 6]]$ and the mother is $mother = [[2, 6], [3, 4], [1, 5]]$. The length of the cross is 1. The detailed operation steps are as follows.

1) CROSSOVER OPERATOR

STEP1: The offspring first get all the genes of the father, that is, $child = [[1, 4], [2, 5], [3, 6]]$

STEP2: The point where a cross is randomly generated is $r = 2$. Since the length of the cross is 1, the position that needs to be crossed is 2.

STEP3: Swap the two individuals at position 2. At this time, the individual of the offspring can be obtained as $child = [[1, 4], [3, 4], [3, 6]]$.

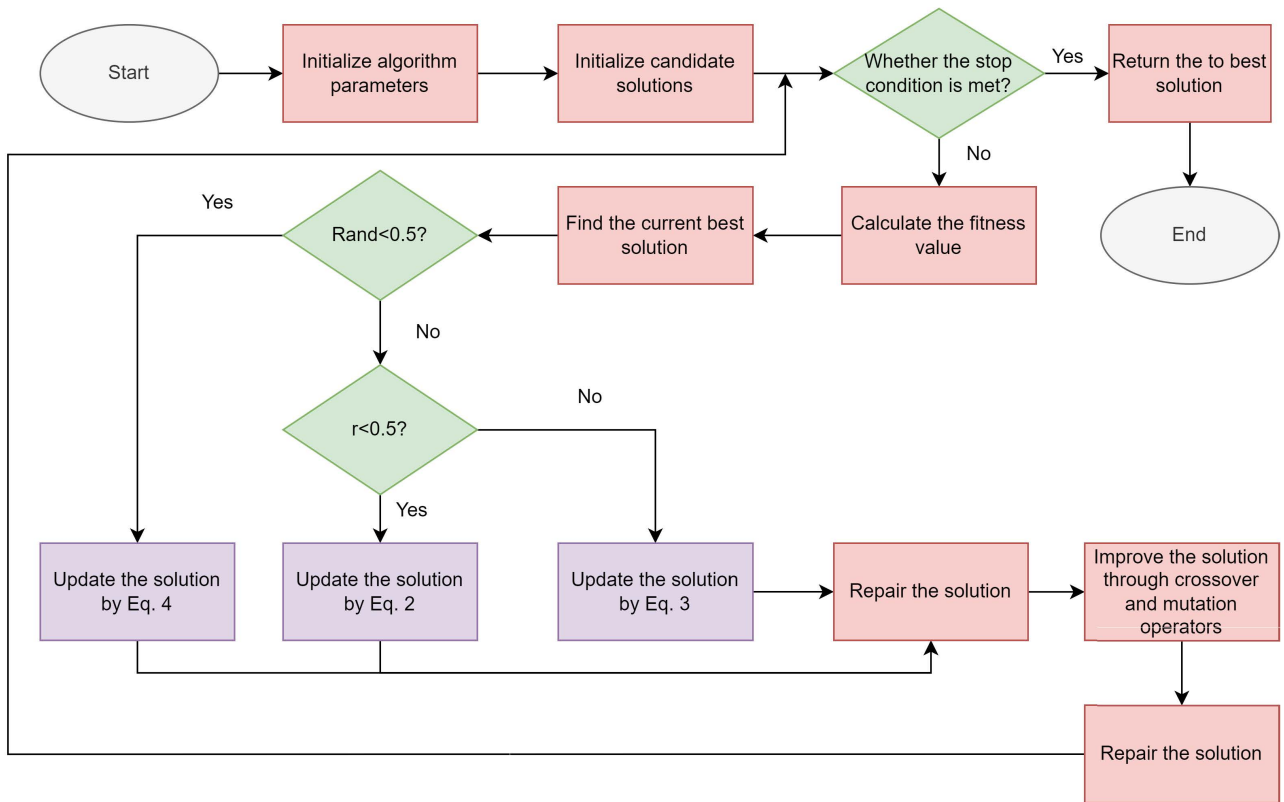


FIGURE 2. Flowchart of the proposed HBA-GA.

2) MUTATION OPERATOR

STEP4: The position where the mutation is generated randomly, here is set to 2. The position of the mutation is randomly generated, which is set to 2 here, that is, the second flight pairing is mutated. A Captain 2 and co-pilot 3 are randomly selected from all pilots. At this point, we can get $child = [[1, 4], [1, 3], [3, 6]]$.

The individual produced after the crossover and mutation operations are completed may not be a viable solution, and we also need *repair operation* to fix it.

D. DETAILED DESCRIPTION OF HBA-GA

In the proposed HBA-GA algorithm, a set of feasible initial solutions are first randomly generated. Then use HBA and Levy flight with some probability to update the current solution. When $rand < 0.5$, Levy flight is executed, otherwise, HBA is executed. Finally, the quality of the solution is further improved by the crossover operator and mutation operator. Update the current best solution until the stop condition is met. A complete description of HBA-GA is shown in Algorithm 2.

E. HBA-GA COMPUTATIONAL COMPLEXITY

In general, the time complexity of metaheuristics depends on the population size, the maximum number of iterations and the number of variables. The computational complexity of updating the solution using Equations 13, 14 and 15 is $O(N)$. Meanwhile, in the worst case, the computational complexity

of converting an infeasible solution to a feasible solution is $O(N^2)$. In our algorithm, only the worst 10% individuals are improved by crossover and mutation operators in each iteration. Then convert these infeasible individuals into feasible individuals. In this process, its computational complexity is $O(N^2 * M * 10\%)$. Therefore, the overall computational complexity of the HBA-GA algorithm is $O(T * (M * (N^2 + N) + N^2 * M * 10\%)) = O(T * M * N^2)$.

V. EXPERIMENTAL RESULTS AND DISCUSSION

To demonstrate the performance of the previously proposed HBA-GA algorithm on CRP, we conduct extensive experiments on multiple instances. Our experiments are mainly conducted under two data types: instances without any qualifications and instances with qualifications. In addition to this, we also compare the results of HBA-GA with those of several other well-known meta-heuristics: genetic algorithm (GA), arithmetic optimization algorithms (AOA) [22], particle swarm optimization algorithm (PSO) [26], honey badger algorithm (HBA) [21], whale optimization algorithm (WOA) [23], grey wolf optimizer (GWO) [24] and aquila optimizer (AO) [25]. These algorithms have been proven to have excellent results when solving engineering problems. All experiments were coded in python on a laptop((Intel Core i7-1165G7 CPU @2.8 GHz on Windows 10)).

Since these meta-heuristics are random, we run each algorithm 10 times independently on each instance for comparative accuracy. Each algorithm may have different parameters.

Algorithm 2 HBA-GA Algorithm

```

1: Initialize the parameters  $\beta$ ,  $C$  of the HBA algorithm, the
   population size is  $M$ , and the dimension of the solution
   is  $N$ .
2: Randomly initialize a solution  $X = \{x_1, \dots, x_M\}$ .
3: Initialize the maximum number of iterations  $T$ .
4: Calculate the fitness of the solution set  $X$ .
5: while  $t < T$  do
6:   if  $\text{rand} < 0.5$  then
7:     for  $i = 1$  to  $M$  do
8:       Updating the solution with Eq.15 produces a new
       solution  $x_{new}$ .
9:       Repair the new solutions  $x_{new}$ .
10:      if  $f_{new} < f_i$  then
11:         $x_i = x_{new}, f_i = f_{new}$ 
12:      end if
13:      if  $f_{new} < f_{prey}$  then
14:         $x_{prey} = x_{new}, f_{prey} = f_{new}$ 
15:      end if
16:    end for
17:  else
18:    for  $i = 1$  to  $M$  do
19:      Update density factor  $\alpha$ .
20:      Produces random numbers  $r, r_1, r_2, r_3, r_4, r_5$  and
       $r_6$  between 0 and 1.
21:      if  $r < 0.5$  then
22:        for  $j = 1$  to  $N$  do
23:          Update the solution by Eq.13.
24:        end for
25:      else
26:        for  $j = 1$  to  $N$  do
27:          Update the solution by Eq.14.
28:        end for
29:      end if
30:      Repair the new solutions  $x_{new}$ .
31:      if  $f_{new} < f_i$  then
32:         $x_i = x_{new}, f_i = f_{new}$ 
33:      end if
34:      if  $f_{new} < f_{prey}$  then
35:         $x_{prey} = x_{new}, f_{prey} = f_{new}$ 
36:      end if
37:    end for
38:  end if
39:  Improve the worst 10% individuals by crossover and
  mutation operators.
40:  Repair  $X_{new2}$  and calculate its fitness.
  Find the minimum fitness  $f_{prey}$  and its corresponding
  solution  $x_{prey}$ .
41: end while
42: Returns the best solution  $x_{prey}$ .

```

We use the parameter values recommended by the original paper during the experiment. In addition to this, we set the population size and the number of iterations to

20 and 2000 respectively, which are parameters that every algorithm has.

A. DATASET INFORMATION

In our simulation experiments, our basic data comes from question F of the 2021 Huawei Cup Graduate Mathematical Contest in Modeling. The dataset consists of two parts, Data A and Data B. Both Data A and Data B contain flight and crew information, of which Data A contains a total of 206 flights and 21 pilots, and Data B contains 13954 flights and 465 pilots. In flight information, it mainly includes 8 elements, namely flight number, departure date, departure time, departure airport, arrival date, arrival time, arrival airport, and minimum qualification configuration. In the crew information, it mainly includes 7 elements, namely the employee number, the captain, the deputy captain, the Deadhead, the base, the duty cost per unit hour, and the task pairing cost per unit hour. Data A is a small-scale dataset that spans a total of 15 days, as shown in Instance #1 in Table 3. However, Data B is a large-scale dataset that spans 30 days. In our experiments, flights within a scheduling period have been combined into flight pairs in advance. Since Data B is a super large-scale dataset, we divided flight pairings and pilots into datasets of different scales, as shown in Instance #2 to Instance #6 in Table 3. At the same time, we added qualification information in datasets Instance #5 and Instance #6. These 6 instances will be used to evaluate the effectiveness of our proposed algorithm.

B. THE PERFORMANCE OF HBA-GA ON THE OBJECTIVE FUNCTION 1

In this section, we will test the CRP problem with the objective function f_1 . The objective function f reflects the income balance of pilots of the same level and the same qualification in a scheduling cycle. We use this objective function to measure the fairness between pilots. To evaluate the performance of HBA-GA, 8 algorithms are tested on 6 instances. Each algorithm is run 10 times independently in each instance. Since evolutionary algorithms have a certain degree of randomness in each run, we separately recorded the best function value, worst function value, mean and standard deviation of each algorithm in 10 runs. In this paper, gains are made by improving two aspects and adding them to the HBA algorithm.

1. Since the global optimization ability of HBA is not strong, it is easy to fall into the local optimum. Therefore, we added Levy flight in the evolution stage, which significantly enhanced the global optimization ability of the algorithm.

2. After the evolution phase is completed, we also apply the crossover and mutation operators in the candidate solutions, which allows the algorithm to find better solutions. This search can improve the quality of the solution.

The statistical results of several classical algorithms are shown in Table 4. In Table 4, our proposed HBA-GA outperforms the other 7 well-known algorithms in best fitness,

TABLE 3. The pilots and pairings of instances.

Instances	Pairings	H Pairings	Co-pilots	H Co-pilots	Captains	H captains	Time span(days)
#1	22	0	10	0	11	0	15
#2	832	0	254	0	211	0	30
#3	595	0	203	0	168	0	30
#4	357	0	127	0	105	0	30
#5	595	199	254	169	211	140	30
#6	446	149	169	112	149	93	30

Pairings: Number of all flight pairings.
 H Pairings: Number of high-qualified flight pairings required.
 Co-pilots: Number of all co-pilots.
 H Co-pilots: Number of high-qualified co-pilots required.
 Captains: Number of all captains.
 H Captains: Number of high-qualified captains required.

TABLE 4. Results obtained from 10 independent runs of CRP with objective function f_1 in terms of mean, standard deviation and running time.

Instance	HBA-GA					AOA				
	Best	Worst	Mean	Std	Running time(s)	Best	Worst	Mean	Std	Running time(s)
#1	3,212	4,193	3,771	271	12	4,090	5,112	4,493	304	8
#2	17,495	19,532	18,088	591	785	18,526	20,636	19,508	649	426
#3	34,177	35,168	34,697	341	319	34,680	35,900	35,406	371	246
#4	65,241	66,187	65,712	301	261	65,545	66,271	65,911	245	150
#5	18,942	20,628	19,955	514	2,463	20,139	22,721	21,721	786	2,139
#6	27,125	30,316	28,994	1,090	1,437	28,086	32,631	30,261	1,353	1,228
Instance	GA					PSO				
	Best	Worst	Mean	Std	Running time(s)	Best	Worst	Mean	Std	Running time(s)
#1	3,931	5,112	4,286	312	12	3,932	4,706	4,279	220	9
#2	18,056	19,382	18,935	344	787	19,634	21,968	20,561	740	535
#3	34,779	35,960	35,420	310	433	35,076	36,853	35,911	565	259
#4	65,612	66,436	66,041	280	295	65,667	66,440	65,980	230	154
#5	20,093	21,723	20,830	452	2,641	21,107	23,154	22,121	679	2,368
#6	29,297	34,533	32,108	1,490	1,565	30,139	33,372	32,003	990	1,283
Instance	HBA					WOA				
	Best	Worst	Mean	Std	Running time(s)	Best	Worst	Mean	Std	Running time(s)
#1	3,982	4,862	4,432	256	11	4,221	5,398	4,761	377	12
#2	18,732	20,765	19,934	559	762	18,986	21,506	20,159	671	821
#3	35,013	36,861	35,695	568	300	35,848	36,927	36,192	287	391
#4	65,366	66,370	65,899	349	183	65,901	66,808	66,342	250	246
#5	20,486	22,592	21,560	727	1,868	21,908	23,173	22,525	404	2,382
#6	28,270	32,826	31,029	1,271	1,095	31,571	35,891	33,252	1,165	1,299
Instance	GWO					AO				
	Best	Worst	Mean	Std	Running time(s)	Best	Worst	Mean	Std	Running time(s)
#1	3,894	4,621	4,178	214	13	4,328	5,147	4,620	289	12
#2	17,910	19,934	19,190	609	749	19,087	20,619	19,762	496	885
#3	34,220	36,099	35,262	533	328	34,594	35,891	35,278	408	463
#4	65,321	66,452	65,829	368	182	65,263	66,590	65,828	404	272
#5	19,269	21,803	21,085	732	2,379	19,520	21,836	21,011	619	3,012
#6	27,589	32,535	30,324	1,279	1,122	27,427	35,339	30,679	1,951	2,069

Std: Standard deviation.

worst fitness and average fitness. Our proposed algorithm provides the best fitness for all instances. As can be seen from Table 4, except for the standard deviation, several other metrics are better than the rest of the 7 algorithms. HBA-GA also outperforms more than half of the algorithms in terms of standard deviation. Therefore, we can see that a fairer crew scheduling scheme can be obtained significantly using the HBA-GA method.

In terms of the convergence of HBA-GA, the figure 4 gives box-plots for 6 instances, HBA-GA has better stability in 10 runs of each instance. We also compared the best performance of each algorithm out of 10 runs. As can be seen from the figure 3, the ability of HBA-GA to jump out of the local optimum is better than the other 7 algorithms.

In the process of convergence, HBA-GA continuously conducts global exploration, and converges to the global optimal value on several instances. Meanwhile, HBA-GA shows faster convergence than several other algorithms.

Instance #1 is a very small instance spanning 15 days, so we can list the results of the crew scheduling using the HBA-GA algorithm, as shown in the Gantt chart of Figure 5. In Figure 5, the ordinate represents the pilot (including the captain and co-pilot), and the abscissa represents the date of the flight pairings. Among them, A001 to A0011 are captains, and A0012 to A0021 are co-pilots. In Figure 5, each block represents a flight pairing, and each flight pairing needs to be assigned twice, i.e. blocks of the same color describe the captain and first officer performing the same flight pairing.

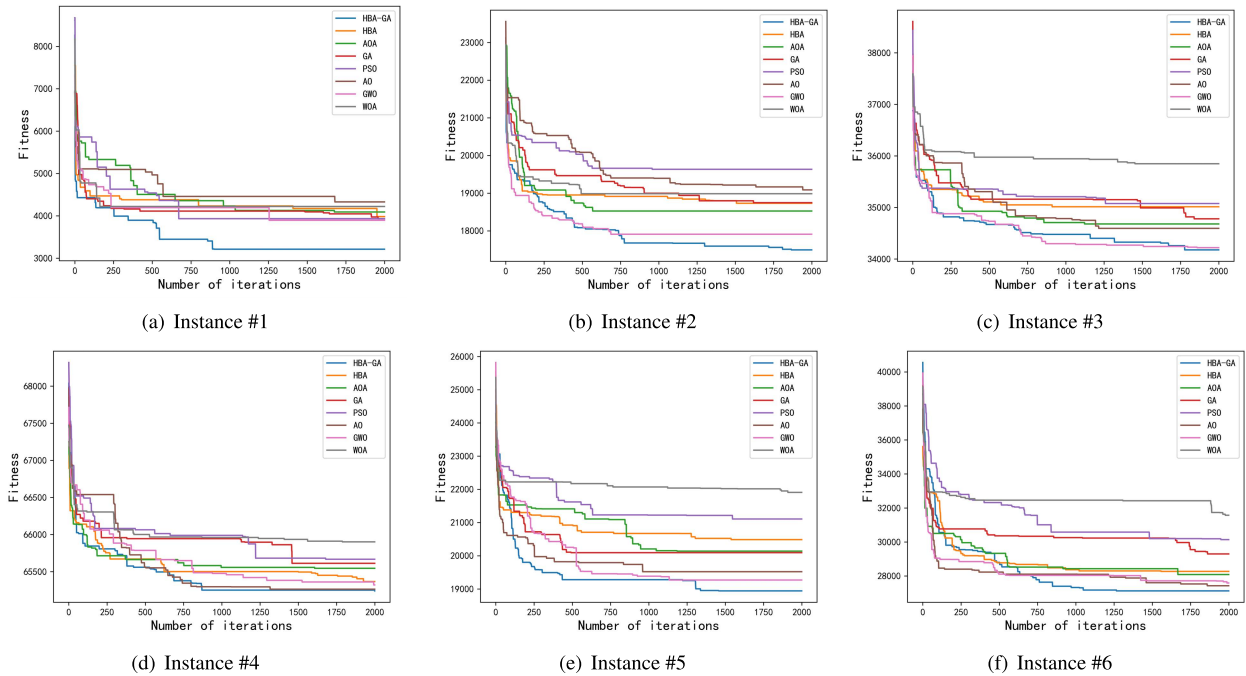


FIGURE 3. Comparison of the convergence of HBA-GA and several other well-known algorithms on f_1 .

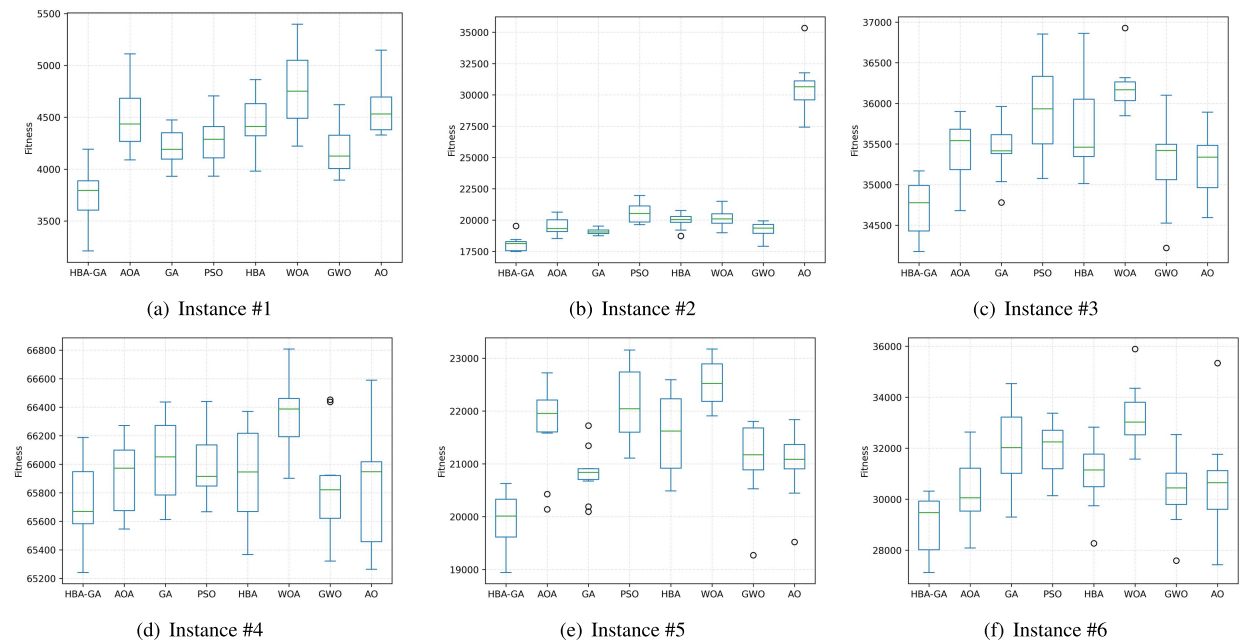


FIGURE 4. Box-plots of the best fitness values obtained from 10 runs of each algorithm on six sample instances of function f_1 .

As can be seen from Figure 5, all pilots meet the rest time and duty time constraints, ie this is a feasible allocation. Due to the large amount of data in the other 5 instances, we only list the allocation Gantt chart of instance #1.

C. THE PERFORMANCE OF HBA-GA ON THE OBJECTIVE FUNCTION 2

Objective function 2 represents the cost minimization of airlines, which is also the objective function commonly used in

most literatures. In this section, this objective is also optimized. In Table 5, the comparison of HBA-GA and several other algorithms is shown, as shown in Table 5, HBA-GA outperforms other algorithms on both the best solution and the worst solution. And it is also better than several other algorithms in terms of average fitness function. In terms of standard deviation, HBA-GA outperforms several other algorithms on instances #1, #5, #6. This shows that the stability of the algorithm of HBA-GA is only weaker than other

TABLE 5. Results obtained from 10 independent runs of CRP with objective function f_2 in terms of mean, standard deviation and running time.

Instance	HBA-GA					AOA				
	Best	Worst	Mean	Std	Running time(s)	Best	Worst	Mean	Std	Running time(s)
#1	620,766	622,352	621,470	605	11	622,216	626,450	624,216	1,399	7
#2	32,367,643	32,399,646	32,389,222	9,540	575	32,485,009	32,507,743	32,497,294	7,152	387
#3	25,962,579	25,979,356	25,971,761	5,113	213	26,058,513	26,077,150	26,065,165	5,616	189
#4	17,634,873	17,656,616	17,646,303	7,756	171	17,693,060	17,707,556	17,699,355	4,296	184
#5	27,951,641	27,972,828	27,959,705	6,113	1783	28,025,647	28,056,732	28,042,357	8,363	1543
#6	22,731,439	22,747,318	22,740,491	4,281	911	22,804,500	22,836,713	22,817,859	10,269	722
Instance	GA					PSO				
	Best	Worst	Mean	Std	Running time(s)	Best	Worst	Mean	Std	Running time(s)
#1	621,258	624,417	622,566	838	11	623,935	629,229	625,652	1,679	8
#2	32,392,213	32,420,000	32,411,112	9,486	709	32,444,150	32,505,373	32,488,160	17,333	463
#3	25,992,116	26,003,030	25,996,883	3,438	212	26,044,843	26,072,269	26,060,510	7,904	196
#4	17,659,706	17,666,213	17,663,087	1,867	198	17,697,063	17,713,293	17,704,545	4,678	174
#5	27,977,234	28,016,731	27,992,305	12,642	1,874	28,037,820	28,057,812	28,049,818	6,429	1,809
#6	22,759,836	22,778,341	22,769,688	5,875	976	22,776,354	22,798,367	22,788,869	7,354	873
Instance	HBA					WOA				
	Best	Worst	Mean	Std	Running time(s)	Best	Worst	Mean	Std	Running time(s)
#1	622,859	626,167	623,852	930	12	623,922	628,226	626,622	1,191	12
#2	32,483,230	32,514,726	32,495,651	7,862	579	32,492,173	32,513,920	32,504,019	7,454	584
#3	26,043,146	26,071,220	26,059,215	7,155	187	26,056,316	26,090,196	26,066,625	8,949	231
#4	17,695,810	17,712,320	17,702,922	4,827	153	17,692,703	17,723,766	17,707,054	8,170	187
#5	28,037,751	28,055,218	28,045,478	6,322	1,462	28,049,826	28,077,681	28,064,092	8,189	1,789
#6	22,795,621	22,821,673	22,808,287	7,890	683	22,809,278	22,834,616	22,824,252	7,704	782
Instance	GWO					AO				
	Best	Worst	Mean	Std	Running time(s)	Best	Worst	Mean	Std	Running time(s)
#1	622,226	625,793	624,122	1,086	12	622,631	625,747	624,351	917	11
#2	32,472,623	32,501,086	32,488,933	7,551	524	32,473,750	32,503,893	32,492,318	8,775	635
#3	26,042,963	26,069,496	26,058,787	8,505	229	26,044,373	26,072,023	26,059,866	8,348	237
#4	17,690,083	17,706,453	17,700,805	5,891	146	17,696,133	17,714,423	17,705,467	5,824	227
#5	28,029,878	28,054,217	28,040,742	7,147	1,763	28,026,731	28,057,321	28,044,903	8,457	2,031
#6	22,805,631	22,841,672	22,819,596	11,244	674	22,809,267	22,841,673	22,822,668	10,256	1,120

Std: Standard deviation.

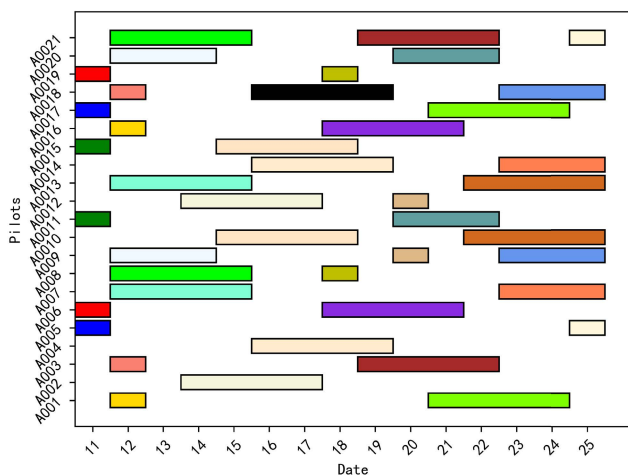


FIGURE 5. Gantt chart on instance #1.

algorithms in individual cases. This shows that the HBA-GA method performs well when solving the objective function to minimize the cost of CRP. This can be seen more clearly in the box-plot in Figure 6.

D. COMPARISON OF RUNNING TIME

In this section, we compare the computation time of several algorithms on two objective functions. As shown in Table 4 and Table 5, the running time of our proposed HBA-GA algorithm is in the middle of the 8 algorithms. From Table 4

and Table 5, we can clearly see that when the number of flight pairings and pilots in the dataset remains the same, while adding other highly qualified flight pairings and pilots, the running time of each algorithm increases significantly. Therefore, we can conclude that as flight pairing and pilot matching conditions are increased, the run time of the algorithm increases. At the same time, when the running time of other algorithms is not much different, the HBA-GA algorithm proposed by us can significantly improve the fitness value.

E. COMPARISON WITH BRANCH PRICING ALGORITHMS

In fact, our proposed model(objective function f_2) is an integer linear programming problem. At the same time, a large number of variables are included in our model. In this case, solving this model using branch pricing algorithm is a common method. Branch pricing (BP) algorithm is a hybrid optimization algorithm of branch and bound algorithm and column generation algorithm. When using column generation to solve an integer programming problem, the constraint main problem is usually relaxed into a linear programming problem, and after obtaining the optimal solution of the linear relaxation problem, integer programming is used to solve it [27]. However, the integer optimal solution is often not obtained by doing so, so it is necessary to use branch pricing to find the integer optimal solution. Therefore, the process of generating a new set of feasible solutions may take a considerable amount of time. In Table 6, we list the running

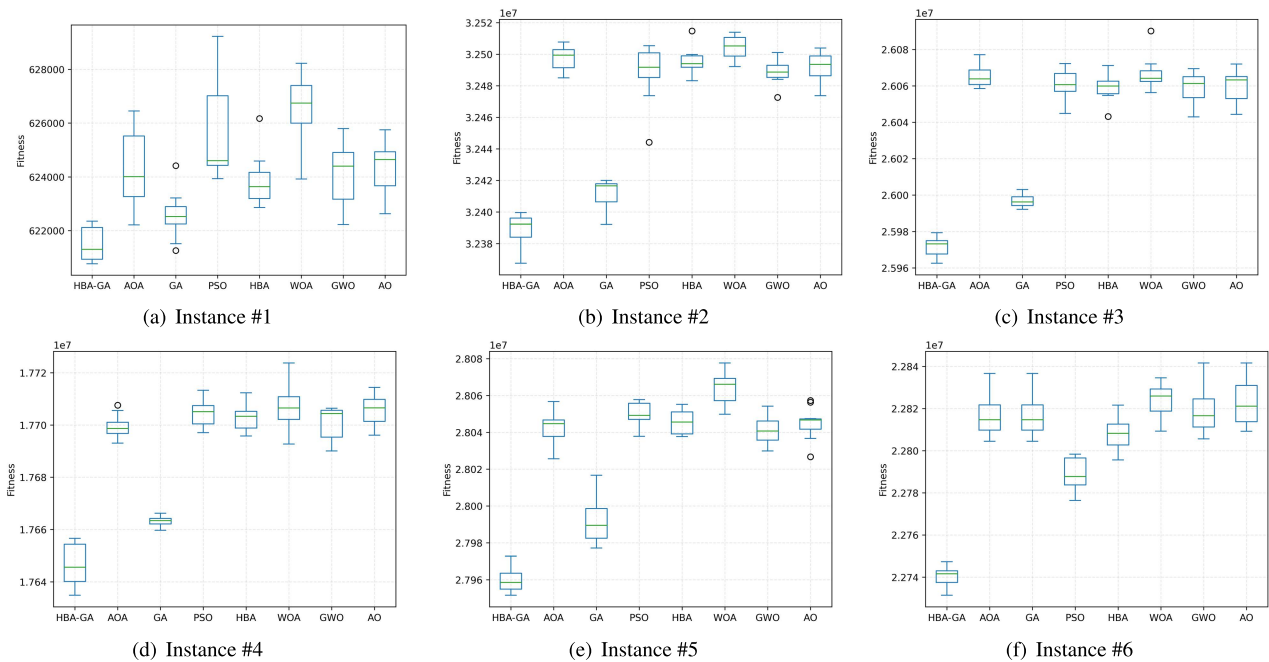


FIGURE 6. Box-plots of the best fitness values obtained from 10 runs of each algorithm on six sample instances of function f_2 .

TABLE 6. CG-VNS vs BP (f_2).

Instance	HBA-GA		BP	
	Time(s)	GAP	Time(s)	GAP
#1	11	0.01%	8	-
#2	575	0.04%	1673	-
#3	213	0.06%	731	-
#4	171	0.07%	522	-
#5	1783	0.09%	5623	-
#6	911	0.04%	2621	-

times of our proposed HBA-GA algorithm and BP algorithm on the objective function f_2 . At the same time, we also list the gap of the objective function value of HBA-GA algorithm and BP algorithm. As can be seen from Table 6, we greatly reduce the running time in large-scale instances while ensuring that the objective function value produced by the BP algorithm is not much different. When we adopt the objective function f_1 as the objective function value of the model, the model is no longer an integer linear programming, so the optimal solution of the model cannot be found using the BP algorithm. The HBA-GA algorithm can only get as close to the optimal solution as possible.

VI. CONCLUSION

Unlike most existing models, we consider a more real-world model that considers flights with high qualification requirements. Specifically, we formulate the objectives of the model from the perspectives of airlines (cost minimization) and pilots (equilibrium of benefits for pilots), taking into account qualifications. Combining HBA and GA algorithms to develop an improved HBA algorithm to solve the

CRP model. Under the HBA algorithm framework, in order to prevent the algorithm from falling into local optimum prematurely, we use a Levy flight strategy with 4 schemes. It randomly selects a scheme to update individual information, which helps to explore the global optimum. In addition, after updating the individuals, we also use the GA algorithm to optimize the worst 10% individuals in the population. Since our model is a discretized model, we design a discretized population update scheme for the CRP.

Experiments are performed on data of different scales that simulate the characteristics of data in the real world. The results show that the computational time of the algorithm is significantly improved when the flight requires highly qualified pilots to perform. Experiments are performed on data of different scales that simulate the characteristics of data in the real world. It outperforms several other swarm intelligence algorithms in most cases. At the same time, it is faster than the branch pricing algorithm in the case of a small difference in fitness. The model we propose is more suitable for the real-world pilot scheduling problem, because most constraints are qualification-related constraints, such as language constraints can also be described as a qualification with language. Therefore, we can use these solutions to guide real-world crew rostering problem. In future work, we will further consider crew rostering problem for flight cancellations due to COVID-19.

REFERENCES

[1] M. Bazargan, *Airline Operations and Scheduling*. Evanston, IL, USA: Routledge, 2016.
 [2] R. Anbil, R. Tanga, and E. L. Johnson, "A global approach to crew-pairing optimization," *IBM Syst. J.*, vol. 31, no. 1, pp. 71–78, 1992.

- [3] G.-F. Deng and W.-T. Lin, "Ant colony optimization-based algorithm for airline crew scheduling problem," *Expert Syst. Appl.*, vol. 38, no. 5, pp. 5787–5793, 2011.
- [4] 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, 2004.
- [5] *Analysis of Crew Scheduling (1)*. Accessed: 2020. [Online]. Available: <https://zhuanlan.zhihu.com/p/164552708>
- [6] D. Wedelin, "An algorithm for large scale 0–1 integer programming with application to airline crew scheduling," *Ann. Oper. Res.*, vol. 57, no. 1, pp. 283–301, Dec. 1995.
- [7] B. Gopalakrishnan and E. L. Johnson, "Airline crew scheduling: State-of-the-art," *Ann. Oper. Res.*, vol. 140, no. 1, pp. 305–337, 2005.
- [8] G. Yu, *Operations Research in the Airline Industry*, vol. 9. Cham, Switzerland: Springer, 1997.
- [9] R. Freling, R. M. Lentink, and A. P. M. Wagelmans, "A decision support system for crew planning in passenger transportation using a flexible branch-and-price algorithm," *Ann. Oper. Res.*, vol. 127, nos. 1–4, pp. 203–222, 2004.
- [10] M. Gamache, A. Hertz, and J. O. Ouellet, "A graph coloring model for a feasibility problem in monthly crew scheduling with preferential bidding," *Comput. Oper. Res.*, vol. 34, no. 8, pp. 2384–2395, Aug. 2007.
- [11] A. Kasirzadeh, M. Saddoune, and F. Soumis, "Airline crew scheduling: Models, algorithms, and data sets," *EURO J. Transp. Logistics*, vol. 6, no. 2, pp. 111–137, Jun. 2017.
- [12] C. P. Medard and N. Sawhney, "Airline crew scheduling from planning to operations," *Eur. J. Oper. Res.*, vol. 183, no. 3, pp. 1013–1027, Dec. 2007.
- [13] R. Nissen and K. Haase, "Duty-period-based network model for crew rescheduling in European airlines," *J. Scheduling*, vol. 9, no. 3, pp. 255–278, Jun. 2006.
- [14] M. Saddoune, G. Desaulniers, I. Elhallaoui, and F. Soumis, "Integrated airline crew scheduling: A bi-dynamic constraint aggregation method using neighborhoods," *Eur. J. Oper. Res.*, vol. 212, no. 3, pp. 445–454, Aug. 2011.
- [15] S. Yan and J.-C. Chang, "Airline cockpit crew scheduling," *Eur. J. Oper. Res.*, vol. 136, no. 3, pp. 501–511, Feb. 2002.
- [16] N. Souai and J. Teghem, "Genetic algorithm based approach for the integrated airline crew-pairing and rostering problem," *Eur. J. Oper. Res.*, vol. 199, no. 3, pp. 674–683, 2009.
- [17] 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, 2017.
- [18] S. Saemi, A. Komijan, R. Tavakkoli-Moghaddam, and M. Fallah, "Solving an integrated mathematical model for crew pairing and rostering problems by an ant colony optimisation algorithm," *Eur. J. Ind. Eng.*, vol. 16, no. 2, pp. 215–240, 2022.
- [19] S.-Z. Zhou, Z.-H. Zhan, Z.-G. Chen, S. Kwong, and J. Zhang, "A multi-objective ant colony system algorithm for airline crew rostering problem with fairness and satisfaction," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 11, pp. 6784–6798, Nov. 2020.
- [20] P. Chutima and K. Arayikanon, "Many-objective low-cost airline cockpit crew rostering optimisation," *Comput. Ind. Eng.*, vol. 150, Dec. 2020, Art. no. 106844.
- [21] F. A. Hashim, E. H. Houssein, K. Hussain, M. S. Mabrouk, and W. Al-Atabany, "Honey Badger Algorithm: New metaheuristic algorithm for solving optimization problems," *Math. Comput. Simul.*, vol. 192, pp. 84–110, Feb. 2022.
- [22] L. Abualigah, A. Diabat, S. Mirjalili, M. Abd Elaziz, and A. Gandomi, "The arithmetic optimization algorithm," *Comput. Methods Appl. Mech. Eng.*, vol. 376, Apr. 2021, Art. no. 113609.
- [23] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Adv. Eng. Softw.*, vol. 95, pp. 51–67, May 2016.
- [24] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Adv. Eng. Softw.*, vol. 69, pp. 46–61, Mar. 2014.
- [25] L. Abualigah, D. Yousefi, M. Abd Elaziz, A. A. Ewees, M. A. A. Al-qaness, and A. H. Gandomi, "Aquila optimizer: A novel meta-heuristic optimization algorithm," *Comput. Ind. Eng.*, vol. 157, Jul. 2021, Art. no. 107250.
- [26] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE ICNN*, vol. 4, Nov./Dec. 1995, pp. 1942–1948.
- [27] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance, "Branch-and-price: Column generation for solving huge integer programs," *Oper. Res.*, vol. 46, no. 3, pp. 316–329, 1998.



BIN DENG received the B.S. degree in surveying engineering from the Southwest University of Science and Technology, Mianyang, China. He is currently pursuing the Ph.D. degree in operations research and cybernetics with Yunnan University, Kunming, China. His research interests include aviation operations, fair distribution, and intelligent algorithms.

• • •