Mathematical Biosciences and Engineering

MBE, 19(12): 13007–13027.
DOI: 10.3934/mbe.2022607
Received: 13 June 2022
Revised: 26 July 2022
Accepted: 30 August 2022
Published: 05 September 2022

http://www.aimspress.com/journal/MBE

Research article

Improved artificial bee colony algorithm for air freight station scheduling

Haiquan Wang[1], **Hans-Dietrich Haasis**[2], **Menghao Su**[3,*], **Jianhua Wei**[4], **Xiaobin Xu**[5], **Shengjun Wen**[1], **Juntao Li**[6] **and Wenxuan Yue**[7]

[1] Zhongyuan-Petersburg Aviation College, Zhongyuan University of Technology, Zhengzhou 450007, China
[2] Chair in Maritime Business and Logistics, University of Bremen, Max-von-Laue-Str. 1, 28359 Bremen, Germany
[3] Faculty of Information Engineering, Zhengzhou University of Industrial Technology, Xinzheng 451100, China
[4] Zhengzhou Tiamaes Technology Co., Ltd, Zhengzhou 450007, China
[5] School of Automation, Hangzhou Dianzi University, Hangzhou 310018, China
[6] Beijing Key Laboratory of Intelligent Logistics Systems, Beijing Wuzi University, Beijing 100000, China
[7] Faculty of Electrical and Engineering, Zhongyuan University of Technology, 41 Zhongyuan Road, Zhengzhou 450007, China

* **Correspondence:** Email: sumenghao1228@outlook.com; Tel: +8618838978169.

Abstract: Aiming at improving the operating efficiency of air freight station, the problem of optimizing the sequence of inbound/outbound tasks meanwhile scheduling the actions of elevating transfer vehicles (ETVs) is discussed in this paper. First of all, the scheduling model in airport container storage area, which considers not only the influence of picking sequence, optimal ETVs routing without collision, but also the assignment of input and output ports, is established. Then artificial bee colony (ABC) is proposed to solve the above scheduling issue. For further balancing the abilities of exploration and exploitation, improved multi-dimensional search (IMABC) algorithm is proposed where more dimensions will be covered, and the best dimension of the current optimal solution is used to guide the evolutionary direction in the following exploitation processes. Numerical experiments show that the proposed method can generate optimal solution for the complex scheduling problem, and the proposed IMABC outperforms original ABC and other improved algorithms.

Keywords: air freight station; dual elevating transfer vehicles; cargo flow schedule; artificial bee colony algorithm; improved multi-dimensional search strategy

1. Introduction

The economy of China is expanding rapidly with the increasing demand for air freight services. Since 2008, China's air cargo and postal transport volume maintains sustained growth, from 4.076 million tons in 2008 to 8.632 million tons in 2020 [1], meanwhile the products needed to be transported become more diversified, a large amount of high value-added goods including fresh goods, seasonal or precious commodities appear. In other words, the situations of multi batch, small batch, fast transportation are more prominent, and higher real-time requirement is presented compared with other logistics industries.

In order to improve the efficiency of air freight transport system, an efficient automated multi-dimensional warehouse with ETVs needs to be established, and as the basis of the system, a proper scheduling strategy considering the assignment of cargo locations as well as the sequence of inbound/outbound tasks should be designed.

From limited references in the past few years, the relevant research mainly focus on deciding the cargoes' transportation sequence. Guo [2] and Qiu et al. [3] studied the inbound and outbound cargoes scheduling problem with single ETV and solved it with particle swarm optimization (PSO) algorithm. Different from the above single ETV scheduling problem, Lei [4] discussed the task scheduling problem with dual ETVs considering collision avoidance and introduced expert system to improve the operation efficiency. Ding [5] proposed task chain generation algorithm with improved shared fitness strategy and applied it to assign two ETVs to different cargo areas.

The works mentioned above developed different scheduling models with considering picking sequence and ETV routing, but they neglected to discuss the problem of assignment of input and output ports (I/O ports) if the air freight station has several entrances and exits. On the other hand, previous scheduling problems have been solved with different intelligent algorithms, but a more efficient algorithm needs to be designed for improving the efficiency and accuracy as the problems become more complex. In our research, ABC algorithm which possesses strong global optimization ability and few parameters [6–9] is first introduced to solve the scheduling problem in air freight transport system.

Actually, as one of the most recently defined bio-inspired algorithms, ABC is motivated by the swarm intelligence behaviors of gregarious colony, and it has been proved to be useful in solving various engineering problems, such as civil engineering design [10], aerospace industry [11], software testing [12], logistics warehouse management [13], manufacturing production [14], communication problem [15] and so on. However, it often suffers from the problems of poor exploitation and slow convergence, so developing new search mechanisms is crucial for complex optimization problems. Recently, other metaheuristic algorithms were introduced and combined with traditional ABC to improve its performance. Ghanem et al. [16] incorporated monarch butterfly optimization into ABC. The firework explosion search mechanism was introduced to explore the potential food sources of ABC in [17]. Gaidhane et al. [18] mixed ABC with grey wolf optimizer to balance the exploration and exploitation capabilities. Liang et al. [19] introduced differential operator in employed bee phase and revised the selection probability to be a step function. Moreover, another improving idea was to replace the single search strategy in original ABC with two or more search strategies. Xu et al. [20] introduced differential evolution strategy in employed bee phase to accelerate its convergence and adopted the global best position to guide the following updating processes in onlooker bee phase which could enhance the local search ability. Literature [21] proposed two search strategies with the help of the information of the best-so-far solution and the mean of two random solutions. In [22], three search strategies with different characteristics were employed to construct a strategy candidate pool, and the Parzen window method was applied to select the best candidate individuals. Two new search strategies were adopted based on the k-neighborhood of best solution in [23] which didn't need to calculate the selection probability. Different from the above literatures, from the perspective of improving the search mechanism, Zhang et al. [24] proposed a new search mechanism named full-dimensional ABC algorithm (fdABC), it was executed with all dimensions of each solution thus the search area could be expanded and the probability of obtaining the optimal solution could be improved. Although the proposed algorithm possessed better optimization performance especially for high-dimensional optimization problems, the time cost increased substantially because more dimensions needed to be updated compared with traditional ABC. In this paper, improvement on the search mechanisms is introduced and applied to solve the scheduling problem in air freight station.

The main features that distinguish this study from previous studies are presented below. a) For solving the tasks scheduling problem in airport container storage area with several I/O ports, the model considering the assignment of I/O ports, as well as the optimization of picking sequence and ETVs' travelling route is first established. The proposed modeling method could be applied to all scheduling problems of storage systems in fast moving consumer goods industry, e-commerce industry and logistics industry; b) In order to balance the abilities of exploration and exploitation of ABC algorithm, two improvements on search mechanisms are proposed. Different from original ABC that only one dimension is randomly updated, improvement is realized by saving the most valuable dimension of current solution and guiding the direction of subsequent exploration. Related improvement could be adopted to solve all optimization and scheduling problems in engineering fields.

The rest of this paper is organized as follows: Section 2 introduces the scheduling model with dual ETVs in the freight station of Luoyang Beijiao Airport. Then ABC and IMABC algorithms are proposed in Section 3, and their optimization performance is verified on benchmark functions. In Section 4, the improved algorithms are applied to the dual ETVs scheduling problem and their effectiveness is proved. Finally, the above work is summarized.

2. Modeling of the scheduling problem

2.1. System description

The air freight station in Luoyang Beijiao Airport consists of three parts, which are container storage area, bulk cargo storage area, and unhandled cargo area respectively as shown in Figure 1.

As the core of the whole system, the container storage area is used for handling the containerized cargoes, which are unloaded from aircraft in the airside or inbounded from the bulk cargo storage area in the landside. It is a three-dimensional warehouse with two rows of shelves and 16 I/O ports, each row has eight layers and 60 columns, the total slots are $60 \times 8 \times 2 = 960$. Two ETVs are employed for handling cargoes between the 14 I/O ports in airside and the two I/O ports in landside, and each ETV is responsible for half of the shelf and I/O ports.

The operational process of the two ETVs is depicted as Figure 2 (some columns and I/O ports are omitted). ETVs start from R_1 and R_3 respectively, the 1# ETV picks up cargo at entrance R_2, and stores it at I_1. And then, it retrieves outbound cargo from O_2 and delivers it to the exit C_1. The same actions are executed with 2# ETV from R_3 to C_2.

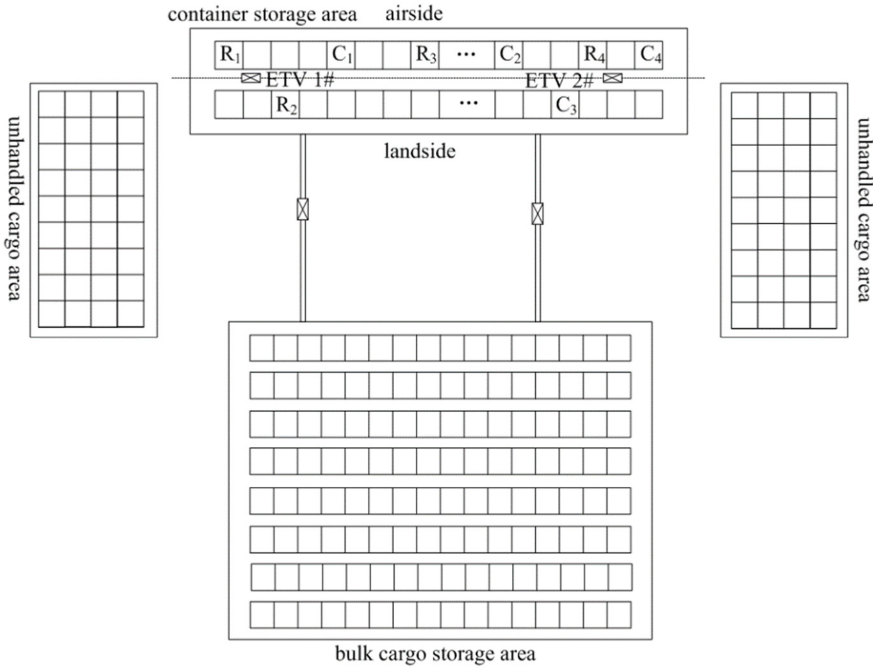

Figure 1. The structure of air freight station.

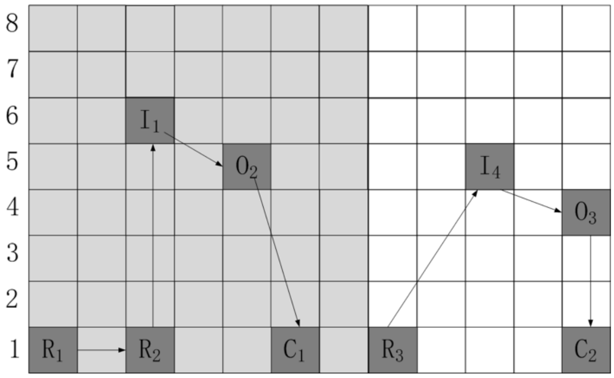

Figure 2. Operational process of dual ETVs.

Obviously, in order to improve the efficiency of cargo transportation in the container storage area, the sequence of inbound/outbound tasks as well as the corresponding I/O ports need to be scheduled, meanwhile the actions of ETVs should be optimized to obtain the minimum running time.

2.2. Model formulations

For solving the scheduling problem, the objective function should be established firstly based on the notations defined in Table 1.

Table 1. Definition of notations.

Indices	Description
x	The horizontal direction
y	The vertical direction
c	The number of columns between two slots in the shelf
l	The number of layers between two slots in the shelf
i	Task number, $i = 1, 2, \ldots, n$
j	Number of ETV, $j = 1, 2$
Parameters	Description
V_x	the horizontal travelling speeds of ETV, $V_x = 120(m/\,min)$
V_y	the vertical travelling speeds of ETV, $V_y = 20(m/\,min)$
a_x	the horizontal acceleration of ETV, $a_x = 0.5(m/s^2)$
a_y	the vertical acceleration of ETV, $a_y = 0.3(m/s^2)$
T_{xa}	the time needed to travel horizontally from static to the maximum speed and immediately decreases to 0, $T_{xa} = 2 \times \frac{V_x}{a_x} = 8(s)$
T_{ya}	the time needed to lift vertically from static to the maximum speed and immediately decreases to 0, $T_{ya} = 2 \times \frac{V_y}{a_y} = 2.2(s)$
D_x	the total travelling distance needed to travel from static to the maximum speed or from maximum speed to 0, $D_x = \frac{1}{4} \times a_x \times {T_{xa}}^2 = 8(m)$
D_y	the total lifting distance needed to travel from static to the maximum speed or from maximum speed to 0, $D_y = \frac{1}{4} \times a_y \times {T_{ya}}^2 = 0.36(m)$
w	width of the storage location, $w = 3.75(m)$
h	height of the storage location, $h = 3.75(m)$
δ	the execution time for ETV to load or unload cargoes, $\delta = 25(s)$
Variables	Description
T_{xc}	the time needed to move between two positions whose interval is c in horizontal direction
T_{yl}	the time taken to move between two positions whose interval is l in vertical direction
H_{i_o}	for the *i-th* task, the running time needed for travelling from the current position to the nearest I/O port
H_{i_1}	The time of the *i-th* task for travelling from the current position to the scheduled target.
M^1	working area of 1# ETV, $1 \le M^1 \le 30$
M^2	working area of 2# ETV, $31 \le M^2 \le 60$
fit_j	the total time of the *j*-th ETV needed to finish the scheduled task

If there are several inbound and outbound tasks in the storage area, the objective of optimized problem which is named as fitness function is to minimize the total execution time *H*H_{max} between two ETVs under the constraints in Eq (2).

$$\text{Minimize } \; H_{max} = \max\{fit_1, fit_2\} \tag{1}$$

s. t.

$$\begin{cases} |M^1 - M^2| \ge 4 \\ set = set1 \cup set2, set1 \ne set2 \end{cases} \tag{2}$$

$$fit_j = \sum_{i=1}^{n} (H_{i_0} + H_{i_1}) + 2n\delta \tag{3}$$

The first constraint in Eq (2) ensures that the columns between two ETVs at any time are no less than four, which could avoid collision with each other. As the task set is divided into two equal parts and assigned to two ETVs, the second constraint avoids repeated allocation. The time fit_j in Eq (3) is defined as the summation of execution time H_{io} and H_{i_1} of n tasks assigned to the specific ETV, it is the total time cost for picking up, releasing as well as transporting all assigned cargoes. Here the execution time H_{io} and H_{i_1} corresponding to different tasks can be obtained from Table 2 (Only the values corresponding to the first five layers and six columns are listed), which are calculated from Eqs (4) and (5). The first row and first column of Table 2 respectively represent the number of difference of rows and columns between the start position and destination in the shelf.

$$\begin{cases} T_{xc} = 2 * \sqrt{\frac{w*c}{a_x}},\ w*c \le D_x; \\ T_{xc} = T_{xa} + \frac{w*c-D_x}{V_x},\ \ w*c > D_x. \end{cases} \tag{4}$$

$$\begin{cases} T_{yl} = 2 * \sqrt{\frac{h*l}{a_y}}, h*l \le D_y; \\ T_{yl} = T_{ya} + \frac{h*l-D_y}{V_y}, h*l > D_y. \end{cases} \tag{5}$$

Table 2. The execution time matrix of ETV.

	1	2	3	4	5	6
1	0	5.47	7.74	9.62	11.50	13.37
2	11.62	11.62	11.62	11.62	11.62	13.37
3	22.87	22.87	22.87	22.87	22.87	22.87
4	34.12	34.12	34.12	34.12	34.12	34.12
5	45.37	45.37	45.37	45.37	45.37	45.37

For solving the above nonlinear scheduling problem, an effective optimization algorithm should be introduced.

3. ABC algorithm and its improvements

3.1. ABC algorithm

ABC is an optimization algorithm based on the intelligent foraging behavior of honeybee swarm, where the bee colony consists of three groups: employed bees, onlooker bees and scout bees. The position of a food source represents an optimal solution of the specific problem, and its quality could be evaluated through the fitness value of the corresponding solution [9,25].

The algorithm starts iterative optimization from employed bee phase, it executes a crossover and mutation process with one randomly chosen companion, and the new solution x'_{mk} is updated based on x_{mk} as shown in Eq (6). Then the fitness value of each solution *fitness*$_m$ could be calculated, and the onlooker bee randomly chooses to exploit or not around corresponding employed bee with the probability *Pm* defined as Eq (7). If the current *mth* solution to be exploited cannot improve for several iterations, it will be abandoned, and a scout bee corresponding to a new randomly produced solution will replace it.

$$x'_{mk} = x_{mk} + rand(-1,1) * (x_{mk} - x_{nk}) \tag{6}$$

$$P_m = \frac{fitness_m}{\sum_{m=1}^{SN} fitness_m} \tag{7}$$

where m, n represent the indices of specific solution in the population, $m, n \in \{1,2,\dots,N\}$ $m \neq n$. k is the dimension of the population, $k \in \{1,2,\dots,D\}$. $rand(-1,1)$ is a random number between [-1, 1]. SN is the number of food sources as well as the number of employed bees.

3.2. The improved ABC algorithm

In ABC algorithm, for each solution, only one dimension is randomly selected and updated according to Eq (6) in employed bee phase and onlooker bee phase. In this case, the updated dimension may be different in each iteration and the optimal dimension obtained in the previous iteration is likely to be omitted in the following iterations. Thus, the search toward the possible optimal solutions is unable to be continued, the optimization accuracy and the convergence speed will be affected.

In [23], full dimensional search strategy (fdABC) is introduced to traverse all dimensions of the solution and select the optimal dimension for further exploration, therefore the search could be extended and the possibility of obtaining optimal solution will be improved, but the optimization time increases inevitably. Another improvement called random multi-dimensional artificial bee colony algorithm (RmdABC) is mentioned in [26], the key improvement of the strategy is to randomly select several different dimensions from $\{1,2,\dots,D\}$ for one solution, and execute the updated process with Eq (6) in the employed bee and onlooker bee phases. Obviously, it randomly traverses any several dimensions of the solution in each iteration, and fewer dimensions are updated compared with fdABC, as the result, its time complexity could be greatly improved.

In order to further balance the abilities of exploration and exploitation, IMABC strategy is proposed where more valuable dimensions of solution will be picked out and saved in the external set which is used to guide the subsequent exploration. The operations of IMABC are shown as Figure 3 and outlined as follows:

1) In the first iteration, all dimensions of the solution are searched, and new solutions are generated with Eq (6) in employed bee and onlooker bee phases.

2) The fitness values of the generated solutions need to be compared with the optimal one, and if the new solution is superior to the old one, the solution in that dimension could be recognized as having the potential to be optimized, thus the optimal solution should be substituted with the generated one, the valuable dimension will be recorded in the external set and the flag is set to be one.

3) In employed bee and onlooker bee phases, after updating all dimensions of the solution, the value of flag will be checked. If the flag *sign* is equal to one, it means there is at least one dimension has been updated. If flag *sign* is zero, it demonstrates that the exploration is failed, and the number of iteration *trial* should increase by 1.

$$\text{if } sign == 1 \text{ then } trial = 0;$$

$$\text{else } trial = trial + 1$$

When the whole cycle is greater than or equal to the value of *MaxCycle*, if *trial* < *Limit*, the above updating operations based on the stored optimal dimensions should be performed iteratively. If *trial* > *Limit*, the value of *trial* resets back to zero and a new random solution will be generated to replace the old one.

Obviously, more dimensions will be explored in IMABC, it could cover more solution space and the ability of exploration could be improved compared with ABC algorithm. Meanwhile, with the help of the stored optimal dimension searched so far, the speed towards the global optimal solution could be accelerated and the ability of exploitation will be enhanced compared with fdABC and RmdABC. Therefore, the performance of the exploration and exploitation could be balanced.

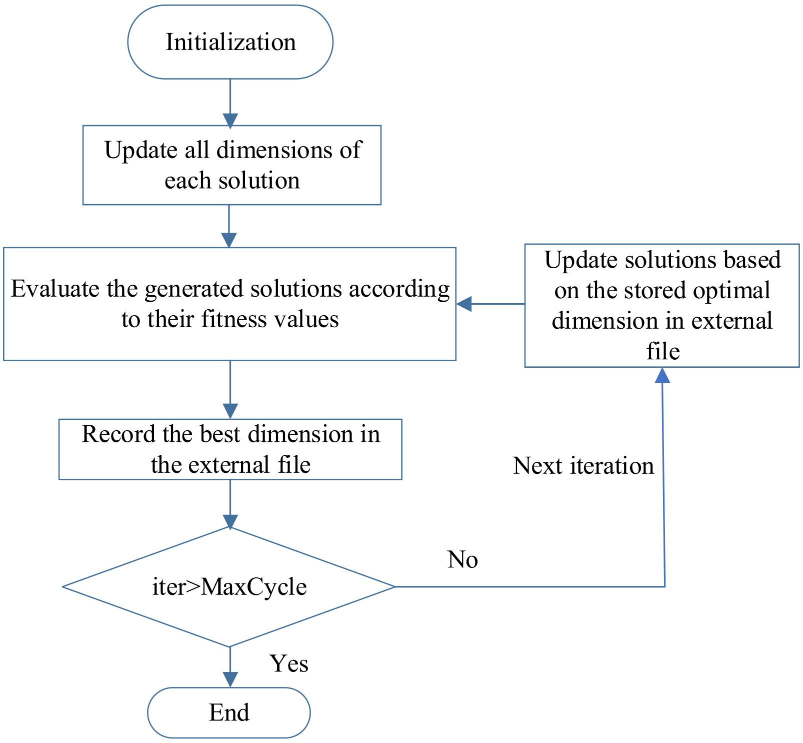
Initialization
Update all dimensions of each solution
Evaluate the generated solutions according to their fitness values
Update solutions based on the stored optimal dimension in external file
Record the best dimension in the external file
Next iteration
No
iter>MaxCycle
Yes
End

Figure 3. The flowchart of IMABC.

The pseudo-code of IMABC is given below.

```
Improved Multi-dimensional ABC
Initialization, set the maximum cycle number, the swarm size, the number of dimensions, the value of *limit*
**for** *iter* = 1 to MaxCycle
    //Employed bee phase
    **for** *m* = 1 to Number of food sources
        *flag* = 0;
        **if** *flag*(*m*) == 0, **do**
$flag(m) = D$, $Dim(m,:) = (1:D)$;
**end if**
*Temp* = 0; //*temp* is used to temporarily store the number of currently mineable dimensions
//greed strategy on multi-dimension
**for** *j* = 1 to *flag*(*m*)
```

Continued on next page

```
            Neighborhood search of employed bees with Eq (6);
            Evaluate the fitness value $fitness(Sol_m)$ with Eq (7);
            if $fitness(Sol_m) < fitness(Food_m)$, do
                $Food_m = Sol_m$; $Dim(m,temp) = Param2Change$; $sign = 1$; $temp = temp + 1$;
            end if
        end for
        $flag(m) = temp$;
        if $sign = 1$ do
            $trial = 0$;
        else
            $trial = trial + 1$;
        end if
    end for
    // onlooker bee phase
    Calculate the probability $prob_m$ if the onlooker bees choose to exploit around the specific source
    for $m = 1$ to Food Number
        $sign = 0$;
        if $rand < prob_m$, do
            $Sign = 0$;
            if $flag$(m) == 0, do
                $flag$(m) = $D$, $Dim$(m,:) = (1:$D$);
            end if
            $Temp = 0$; // $temp$ is used to temporarily store the number of currently mineable dimensions
            for $j = 1$ to $flag(m)$
                Neighborhood search of onlooker bees with Eq (6);
                Evaluate the fitness values of the generated solutions by Eq (7);
                if $fitness(Sol_m) < fitness(Food_m)$, do
                    $Food_m = Sol_m$; $sign = 1$; $temp = temp + 1$; $Dim(m,temp) = Param2Change$;
                end if
            end for
            $flag$(m) = $temp$;
            if $sign = 1$, do
                $trial = 0$;
            else
                $trial = trial + 1$;
            end if
        end if
    end for
    //scout bee phase
    if $trial > limit$, do
        $trial = 0$;
        randomly generate a new solution
    end if
end for
```

3.3. Performance evaluation

In order to evaluate the performance of the proposed IMABC algorithm, nine CEC 2017 benchmark functions as listed in Table 3 are employed, where *$f_1(x)$*, *$f_2(x)$* and *$f_6(x)$* are unimodal functions, and the others are multi-modal functions. All simulations are executed on an Intel Core i7-8750H CPU with 8G RAM, the population size is 200, the dimension of solution is set to 60, 80 and 100 respectively, the number of maximum iterations is set as being 1000, and the limit used in scout bee phase is taken 100. Independent experiments are run 20 trials, the indices including the mean and standard deviation (Mean ± std dev) which reflect the quality of solution and the stability of algorithm, the average running time (Aver-R), the shortest running time (Best-R) and fitness value of the optimal solution (Best-F) are selected to evaluate the optimization performance of different algorithms.

Table 3. The benchmark functions.

Function expression	Searching space	Minimum value	Modality
$\mathrm{f}_1(x) = x_1^2 + 10^6 \sum_{i=2}^{D} x_i^2$	$[-100, 100]$	0	Unimodal
$\mathrm{f}_2(x) = \sum_{i=1}^{D} \lvert x_i \rvert^{i+1}$	$[-100, 100]$	0	Unimodal
$\mathrm{f}_3(x) = \sum_{i=1}^{D-1}(100({x_i}^2 - x_{i+1})^2 + (x_i - 1)^2)$	$[-100, 100]$	0	Multi-modal
$f_4(x) = -20\,exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}) - exp(\frac{1}{n}\sum_{i=1}^{n} cos(2\pi x_i)) + 20 + e$	$[-5, 5]$	0	Multi-modal
$f_5(x) = \sum_{i=1}^{n}[x_i^2 - 10\,cos(2\pi x_i) + 10]$	$[-500, 500]$	0	Multi-modal
$f_6(x) = \sum_{i=1}^{n}(\lvert x_i + 0.5 \rvert)^2$	$[-100, 100]$	0	Unimodal
$f_7(x) = \sum_{i=1}^{D}(x_i^2 - 10\,cos(2\pi x_i) + 10)$	$[-5.12, 5.12]$	0	Multi-modal
$f_8(x) = sin^2(\pi w_1) + \sum_{i=1}^{D-1}(w_i - 1)^2[1 + 10\,sin^2(\pi w_i + 1)] + (w_D - 1)^2[1 + sin^2(2\pi w_D)]$ $w_i = 1 + \frac{x_i - 1}{4}, \forall i = 1, \dots, D$	$[-10, 10]$	0	Multi-modal
$f_9(x) = -20\,exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}) - exp(\frac{1}{D}\sum_{i=1}^{D} cos(2\pi x_i)) + 20 + e$	$[-32.768, 32.768]$	0	Multi-modal

Tables 4–6 illustrate the optimization results with four different algorithms in different dimensions, where the best results among the four indices are highlighted in bold font. As can be seen from the tables, all algorithms could solve the nonlinear problems within limited time, the average running time Aver-R and the shortest running time Best-R increase from 60 dimensions to 100 dimensions because more dimensions need to be updated for all algorithms.

For functions *f₃(x)*, *f₅(x)*, *f₆(x)*, *f₇(x)* and *f₈(x)*, the values of all four indices in each dimension corresponding to IMABC are smallest, which means IMABC possesses the highest optimization accuracy, the fastest convergence speed as well as the best stability when solves the above functions. Another improved algorithm RmdABC gets smaller fitness values and Mean ± std dev values, but it takes longer time compared with the one of ABC. On the other hand, it improves the optimization efficiency of fdABC at the expense of a worse solution. In other words, RmdABC balances the exploration and exploitation abilities of fdABC and ABC.

In addition, IMABC obtains the lowest values of Mean ± std dev and Best-F, but it takes longer running time to obtain optimal solution than ABC algorithm for *f₂(x)*, which means that the stability and global search ability of IMABC are the best, while its convergence performance is better than fdABC and RmdABC because it updates less dimensions. Besides that, as the results of optimizing functions *f₁(x)*, *f₄(x)* and *f₉(x)*, fdABC obtains the best fitness value as well as best stability among all algorithms as it executes full-dimensional search and IMABC takes much less time to get the optimal solution.

The curves of fitness values for functions *f₃(x)* under different dimensions are depicted as Figures 4–6, and the same conclusions as mentioned above could be obtained.

Table 4. The performance of IMABC, RmdABC, fdABC and ABC with 60 dimensions.

		Aver-R(s)	Mean ± std dev	Best-F	Best-R(s)
$f_1(x)$	ABC	111.849	2.320e+5 ± 4.515e+4	1.709e+5	111.4955
	fdABC	253.316	2.929e-252 ± 0	1.416e-252	251.577
	RmdABC	188.172	1.654e-2 ± 1.433e-2	1.49e-4	179.199
	IMABC	29.213	3.469e-159 ± 5.567e-159	1.6314e-160	28.702
$f_2(x)$	ABC	121.258	3.024e+41 ± 6.658e+41	1.593e+36	120.909
	fdABC	676.282	8.680e-255 ± 0	7.047e-258	672.294
	RmdABC	401.083	2.091e-42 ± 4.015e-42	6.660e-46	395.373
	IMABC	229.23419	0 ± 0	0	383.842
$f_3(x)$	ABC	116.400	5490.448 ± 4756.626	4884.468	115.536
	fdABC	284.881	2.33e-3 ± 1.35e-3	1.27e-4	279.176
	RmdABC	212.479	1.1359 ± 0.955	0.323	208.025
	IMABC	79.286	0.0018 ± 0.0020	4.004e-05	78.222
$f_4(x)$	ABC	122.561	0.0294 ± 0.0017	0.0266	121.327
	fdABC	266.4800	6.271e-14 ± 3.432e-15	5.773e-14	272.640
	RmdABC	203.594	4.250e-06 ± 3.376e-06	1.808e-07	200.635
	IMABC	58.415	6.306e-14 ± 3.837e-15	5.773e-14	55.179
$f_5(x)$	ABC	121.356	198.642 ± 12.773	179.002	121.051
	fdABC	250.787	0 ± 0	0	245.425
	RmdABC	190.324	4.145e-06 ± 5.252e-06	2.271e-07	189.681
	IMABC	52.091	0 ± 0	0	51.353288
$f_6(x)$	ABC	120.309	0.232 ± 0.029	0.177	119.949
	fdABC	224.032	0 ± 0	0	218.470
	RmdABC	178.819	1.262e-08 ± 1.059e-08	4.285e-10	178.200
	IMABC	29.427	0 ± 0	0	28.468

Continued on next page

$f_7(x)$	ABC	123.764	4.449e-09 ± 3.849e-09	5.566e-10	122.544
	fdABC	255.694	0 ± 0	0	254.511
	RmdABC	192.567	4.210e-08 ± 3.580e-08	4.677e-10	190.664
	IMABC	84.560	0 ± 0	0	82.977
$f_8(x)$	ABC	127.611	1.354e-05 ± 1.623e-05	2.079e-07	126.884
	fdABC	256.347	0 ± 0	0	254.650
	RmdABC	194.270	8.781e-11 ± 2.043e-10	6.019e-15	192.641
	IMABC	90.816	0 ± 0	0	88.339
$f_9(x)$	ABC	125.083	1.694 ± 0.0995	1.518	123.144
	fdABC	221.099	6.484e-14 ± 2.901e-15	5.773e-14	220.187
	RmdABC	195.0828	1.738 ± 3.070	1.224e-05	188.5410
	IMABC	35.2991	0.058 ± 0.183	5.773e-14	34.9160

Table 5. The performance of IMABC, RmdABC, fdABC and ABC with 80 dimensions.

		Aver-R(s)	Mean ± std dev	Best-F	Best-R(s)
$f_1(x)$	ABC	186.356	7.443e+7 ± 9.252e+6	6.612e+7	185.785
	fdABC	349.896	9.500e-252 ± 0	1.073e-251	341.500
	RmdABC	254.843	1.830 ± 2.791	0.187	242.967
	IMABC	37.905	2.141e-150 ± 2.160e-150	1.580e-151	37.526
$f_2(x)$	ABC	161.169	7.666e+82 ± 2.424e+83	9.065e+73	157.306
	fdABC	1092.133	5.377e-250 ± 0	9.849e-253	1083.415
	RmdABC	641.927	1.236e-18 ± 3.277e-18	4.733e-22	629.479
	IMABC	385.096	0 ± 0	0	383.842
$f_3(x)$	ABC	153.868	65508.192 ± 5912.818	53885.588	152.849
	fdABC	408.311	0.00517 ± 0.00572	0.000291	393.921
	RmdABC	297.845	2.626 ± 1.258	1.1049	287.474
	IMABC	111.143	0.00295 ± 0.00379	2.251e-05	100.158
$f_4(x)$	ABC	155.879	0.152 ± 0.00873	0.140	155.490
	fdABC	381.999	8.757e-14 ± 5.605e-15	7.905e-14	368.461
	RmdABC	279.508	2.649e-05 ± 1.28648e-05	9.587e-06	275.084
	IMABC	79.558	8.900e-14 ± 3.669e-15	8.615e-14	77.818
$f_5(x)$	ABC	155.747	759.215 ± 50.780	687.486	153.785
	fdABC	351.982	0 ± 0	0	346.306
	RmdABC	275.060	0.0184 ± 0.0096	0.0056	270.149
	IMABC	74.003	0 ± 0	0	72.661
$f_6(x)$	ABC	151.843	8.142 ± 0.982	6.349	151.444
	fdABC	300.048	0 ± 0	0	297.340
	RmdABC	537.590	6.049e-07 ± 6.005e-07	2.1807e-08	373.611
	IMABC	56.487	0 ± 0	0	53.586
$f_7(x)$	ABC	162.425	6.674e-08 ± 8.736e-08	1.244e-09	161.965
	fdABC	344.317	0 ± 0	0	341.703
	RmdABC	266.554	3.471e-07 ± 5.419e-07	1.397e-08	260.925
	IMABC	112.593	0 ± 0	0	111.564

Continued on next page

$f_8(x)$	ABC	165.271	2.556e-05 ± 2.445e-05	9.431e-08	164.439
	fdABC	344.205	0 ± 0	0	343.570
	RmdABC	260.079	2.892e-10 ± 3.535e-10	1.326e-14	258.259
	IMABC	121.254	0 ± 0	0	119.521
$f_9(x)$	ABC	162.977	3.614 ± 0.087	3.503	161.545
	fdABC	306.928	8.971e-14 ± 3.745e-15	8.615e-14	301.461
	RmdABC	273.895	2.0297e-04 ± 1.3504e-04	9.416e-05	261.859
	IMABC	56.5127	0.086 ± 0.272	7.905e-14	54.828

Table 6. The performance of IMABC, RmdABC, fdABC and ABC with 100 dimensions.

		Aver-R(s)	Mean ± std dev	Best-F	Best-R(s)
$f_1(x)$	ABC	223.196	3.787e+8 ± 3.315e+7	3.416e+8	222.774
	fdABC	425.692	2.044e-251 ± 0	1.091e-251	420.020
	RmdABC	303.243	12.037 ± 13.399	0.405	300.169
	IMABC	47.675	2.788e-144 ± 1.955e-144	6.548e-145	47.180
$f_2(x)$	ABC	199.808	2.279e+120 ± 4.714e+120	7.215e+111	197.343
	fdABC	1664.371	1.671e-245 ± 0	6.863e-248	1619.398
	RmdABC	920.823	3.470e-06 ± 9.807e-06	4.151e-09	911.149
	IMABC	583.836	0 ± 0	0	579.155
$f_3(x)$	ABC	189.997	5.919e+3 ± 4.890e+3	4.346e+5	186.093
	fdABC	538.976	0.00884 ± 0.00966	0.001	504.637
	RmdABC	351.792	6.042937 ± 4.06785	1.430	337.071
	IMABC	149.570532	0.00414 ± 0.00539	7.879e-05	128.478
$f_4(x)$	ABC	192.425	0.585 ± 0.057	0.476	191.377
	fdABC	733.001	1.1493e-13 ± 3.910e-15	1.110e-13	721.068
	RmdABC	376.785	0.006227 ± 0.00221	0.003694	372.387
	IMABC	100.566	1.167e-13 ± 6.311e-15	1.110e-13	98.099
$f_5(x)$	ABC	193.990	152.642 ± 8.074	193.287	193.2873
	fdABC	458.470157	0 ± 0	0	452.713643
	RmdABC	574.398	7.97e-4 ± 8.87e-4	8.599e-05	573.802
	IMABC	93.270	0 ± 0	0	92.877
$f_6(x)$	ABC	189.754	78.395 ± 9.899	63.6545	187.726
	fdABC	637.867	0 ± 0	0	622.008
	RmdABC	289.625	1.379e-07 ± 1.368e-07	1.838e-08	287.601
	IMABC	49.458	0 ± 0	0	48.919
$f_7(x)$	ABC	202.594	2.755e-08 ± 2.477e-08	6.526e-11	202.314
	fdABC	441.625	0 ± 0	0	433.375
	RmdABC	338.374	1.000e-06 ± 1.085e-06	8.739e-10	328.849
	IMABC	152.650	0 ± 0	0	148.707

Continued on next page

$f_8(x)$	ABC	205.3121	4.432e-05 ± 3.308e-05	2.5522e-06	204.430
	fdABC	444.093	0 ± 0	0	440.009
	RmdABC	332.326	2.141e-08 ± 4.487e-08	2.021e-11	330.175
	IMABC	152.809	0 ± 0	0	150.659
$f_9(x)$	ABC	203.890	5.293 ± 0.181	4.981	203.639
	fdABC	406.710	1.1564e-13 ± 6.050e-15	1.039e-13	402.956
	RmdABC	342.476	4.945e-04 ± 3.758e-04	2.069e-4	338.625
	IMABC	77.486	0.012 ± 0.036	1.039e-13	72.511

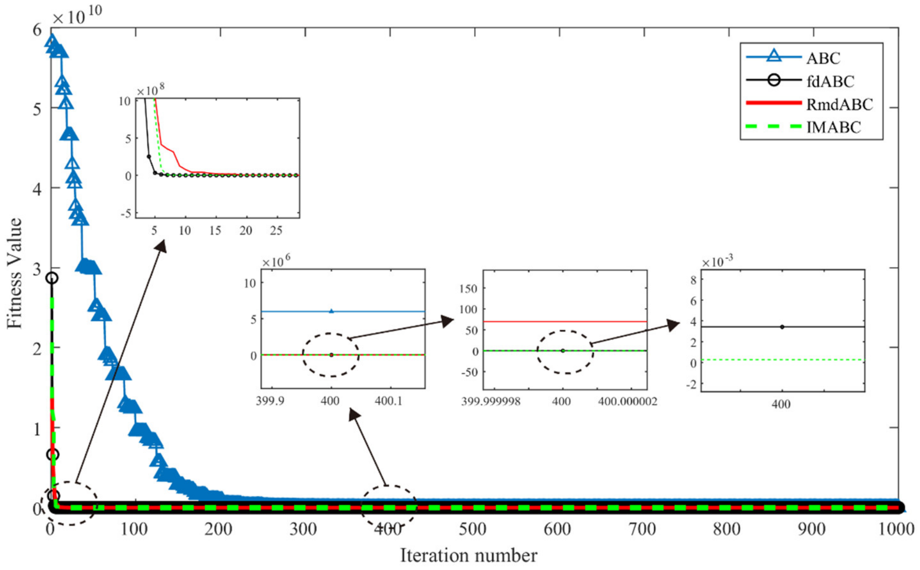

Figure 4. The fitness values of *f₃(x)* corresponding to different algorithms with 60 dimensions.

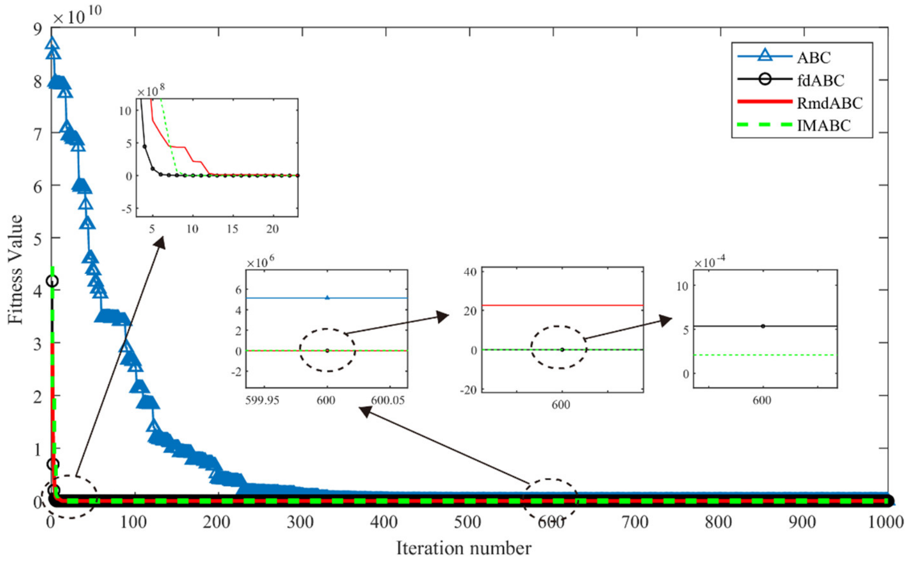

Figure 5. The fitness values of *f₃(x)* corresponding to different algorithms with 80 dimensions.

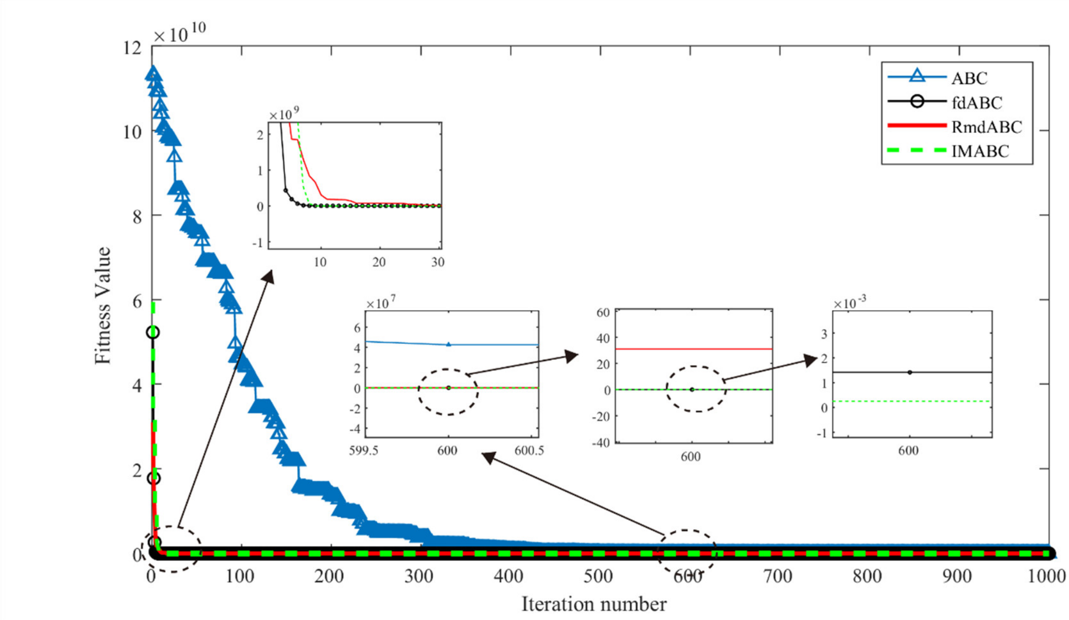

Figure 6. The fitness values of *f*₃*(x)* corresponding to different algorithms with 100 dimensions.

From the analysis above, it can be seen that the proposed IMABC algorithm is able to produce better solutions with higher stability compared with ABC, and costs shorter computational time than other improved algorithms. RmdABC which is deduced from fdABC could reduce its time cost, but the quality of solution becomes poor because it updates less dimensions. Therefore, the proposed IMABC and RmdABC algorithms can balance exploration and exploitation abilities of ABC and fdABC.

4. Task set schedule with improved ABC algorithms

The performance of proposed algorithms has been evaluated by solving complex mathematical problems above, and they will be applied to solve the scheduling problem in the container storage area of Luoyang Beijiao Airport in this section.

There are 16 entrances and exits in the container storage area of the airport station, the entrance coordinates are R_1 (1-1-5), R_2 (1-1-15), R_3 (2-1-20), R_4 (1-1-25), R_5 (1-1-30), R_6 (1-1-35), R_7 (1-1-40), R_8 (1-1-50) and R_9 (1-1-60), and the exit coordinates are C_1 (1-1-8), C_2 (1-1-18), C_3 (1-1-28), C_4 (1-1-38), C_5 (2-1-48), C_6 (1-1-53) and C_7 (1-1-58), where R_3, C_5 are I/O ports at the landside, and the rest are I/O ports at the airside. The first value in the bracket represents the row number of the shelf, the second value indicates the number of layer and the third value is the number of column. Table 7 depicts the assigned positions and current positions of 60 tasks needed to be scheduled, where the first 30 ones are input tasks, and the last 30 ones are output tasks.

For this constrained optimization problem as shown in Eqs (1) and (2), the solution is the sequence of inbound and outbound tasks with dual ETVs, which corresponds to the minimum total time cost as Eq (1). In other words, it is a discrete optimization problem, integer encoding scheme mentioned in [27] is introduced and random numbers between -10 and 10 are assigned to each dimension of the optimized solutions, after sorting them in ascending order based on their values, the corresponding scheduling scheme as well as the optimal solution could be obtained. Furthermore, the constraint conditions as Eq (2) should be checked for each obtained scheduling scheme in the iterative optimization procedure. If the constraints corresponding to the generated solution are satisfied, the solution will be saved and used for further exploration, otherwise a new solution needs to be introduced.

Comparative studies among four algorithms, including ABC, fdABC, RmdABC and IMABC, are executed for the above scheduling problem. The swarm size of all algorithms is set to 200 with 60 dimensions, the maximum local search time is 50, the stopping criterion is set to 1000 generations. Initial populations are generated through uniformly random sampling from the search space. Each algorithm is independently tested 20 times. The experiments are performed with an Intel Core i7-8750H CPU and 8GB of RAM.

Table 7. Assigned or current positions of inbound or outbound tasks.

	Assigned position of inbound task		Assigned position of inbound task		Current position of outbound task		Current position of outbound task
1	I(1-5-10)	16	I(1-8-44)	31	O(1-3-10)	46	O(2-8-10)
2	I(2-3-14)	17	I(2-8-32)	32	O(1-5-55)	47	O(1-3-32)
3	I(1-3-23)	18	I(2-3-54)	33	O(1-5-25)	48	O(1-4-50)
4	I(1-5-26)	19	I(1-3-40)	34	O(2-4-8)	49	O(2-3-38)
5	I(1-5-30)	20	I(1-4-60)	35	O(2-2-18)	50	O(2-1-58)
6	I(1-2-18)	21	I(1-3-20)	36	O(2-1-16)	51	O(1-5-24)
7	I(1-5-24)	22	I(2-2-43)	37	O(2-3-51)	52	O(1-4-30)
8	I(1-4-40)	23	I(2-4-50)	38	O(1-5-6)	53	O(2-6-40)
9	I(1-5-40)	24	I(1-6-10)	39	O(2-5-3)	54	O(2-4-35)
10	I(1-5-35)	25	I(2-7-20)	40	O(1-6-12)	55	O(2-8-51)
11	I(2-5-23)	26	I(1-6-15)	41	O(2-6-13)	56	O(2-2-30)
12	I(1-7-43)	27	I(2-8-30)	42	O(2-7-49)	57	O(1-2-60)
13	I(1-3-48)	28	I(2-2-45)	43	O(1-7-57)	58	O(1-3-26)
14	I(1-8-50)	29	I(1-7-58)	44	O(1-5-25)	59	O(1-6-35)
15	I(1-6-21)	30	I(1-4-9)	45	O(2-6-18)	60	O(2-8-45)

Table 8 presents the optimization results (fitness value which is the scheduling time corresponding to the optimal solution) in the first ten trials. It can be seen that all proposed algorithms are able to produce high-quality solutions for scheduling problem, the performance of IMABC is better than RmdABC and ABC.

Table 8. The scheduling results with different algorithms.

	1	2	3	4	5	6	7	8	9	10
ABC	2783.30	2784.07	2793.17	2793.17	2801.13	2801.13	2789.08	2793.07	2793.17	2783.30
fdABC	2632.56	2632.56	2631.32	2631.08	2632.56	2632.56	2632.56	2632.56	2632.56	2632.56
RmdABC	2669.17	2664.82	2675.17	2654.29	2667.66	2660.38	2643.60	2646.47	2653.82	2653.82
IMABC	2634.83	2634.20	2641.65	2639.56	2625.64	2630.45	2627.47	2624.59	2634.13	2629.54

Table 9 lists four important indices, they are the average time needed to execute the sequence corresponding to optimal solution in 20 trials (Avg), the fitness value of the best solution (Min), the fitness value of the worst solution (Max) and the corresponding running time (CPU time). It is clear that the last two proposed algorithms possess better performance as the first three indices decrease by 6% at most compared with ABC, but the time needed to obtain the optimal solution is longer. It means RmdABC and IMABC can keep the balance between the optimization accuracy and convergence speed compared with ABC. Moreover, because fdABC covers more dimensions than other mentioned algorithms, the fitness values are better than RmdABC, but the running time is 1.97 and 2.35 times of RmdABC and IMABC.

Table 9. The scheduling results with different algorithms.

	Min(s)	Max(s)	Avg(s)	CPU time (s)
ABC	2783.299	2,801.132	2791.459	22,016.33881
fdABC	2631.078	2632.564	2632.291	114,872.57987
RmdABC	2643.596	2675.170	2658.920	58,699.41842
IMABC	2624.589	2641.653	2632.203	48,795.48948

For the two proposed algorithms, the fitness value obtained by IMABC was a 1% reduction with respect to RmdABC and the searching time with IMABC is 83% of RmABC, obviously the optimization ability of IMABC is better than RmABC.

The conclusion also can be obtained from Figure 7 which depicts the average fitness values of the optimization problem, the curve slope of fitness values corresponding to the three improved algorithms is relatively steeper as compared with ABC, which proves their convergence. And the fitness values corresponding to IMABC converges at about 100 iterations, thus its ability of convergence could be proved.

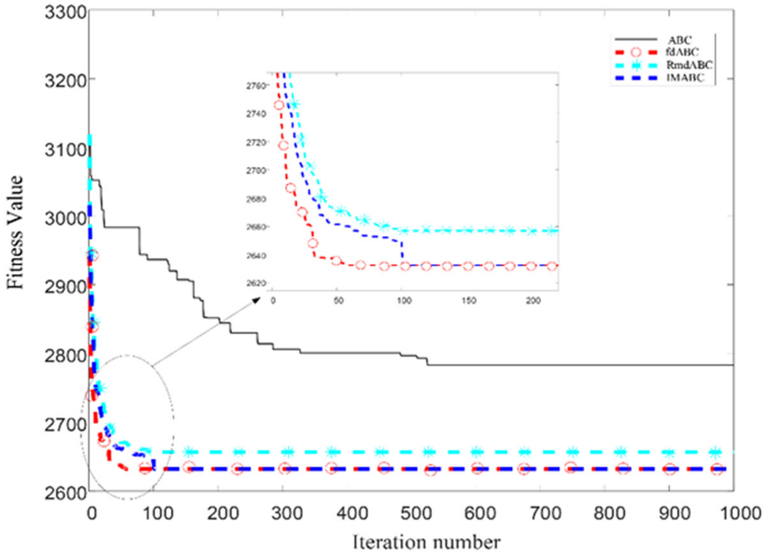

Figure 7. The average fitness values with different algorithms.

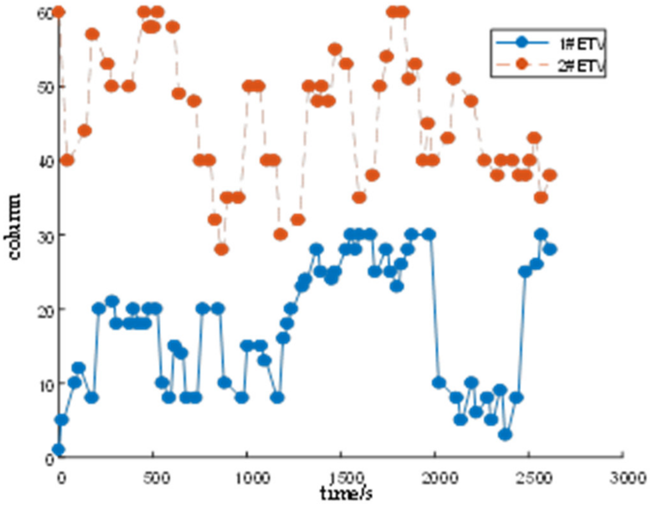

Figure 8. The trajectories of dual ETVs.

Table 10. Entrance and exit allocation scheme optimized by IMABC.

	Inbound tasks		Outbound tasks
1	1, 24, 30	1	31, 34, 38, 39, 40, 41, 46, 60
2	2, 26	2	35, 36, 45
3	6, 11, 15, 21, 25	3	33, 44, 47, 51, 52, 56, 58
4	3, 4, 7	4	49, 53, 54, 59
5	5, 17, 27	5	42, 48, 55
6	10	6	32, 37, 43
7	8, 9, 12, 16, 19, 22, 28	7	50, 57
8	13, 14, 18, 23		
9	20, 29		

Figure 8 and Table 10 show the corresponding trajectories of two ETVs and the sequence of task set optimized with IMABC algorithm respectively. It is clear that the dual ETVs could execute the scheduling tasks successfully without conflicting with each other.

From the above results, the following conclusions could be obtained:

1) IMABC are valid for solving the complex scheduling problem and they can improve the searching ability of traditional ABC algorithm.

2) fdABC possesses the excellent exploration ability, IMABC could keep the balance between the optimization accuracy and convergence speed compared with ABC and fdABC under this scheduling background.

3) For IMABC, with the help of the current optimal dimensions, the convergence speed could be improved compared with fdABC and RmdABC.

5. Conclusions

In this paper, in order to improve the efficiency in the container storage area of airport cargo terminal, a study on scheduling of cargoes sequence and the action of ETV with ABC algorithms was performed. The dual ETVs scheduling model was established which considered the assignment of I/O ports as well as the trajectory of two ETVs with the constraint of avoiding collisions, and improved IMABC algorithm was proposed to solve this scheduling problem more effective. The computational experiments were carried out and the results proved the proposed algorithm could effectively avoid conflicts and generate optimal scheduling sequences.

ABC and corresponding improved algorithms have been proved to be effective in solving the scheduling problem, but the time cost for obtaining the optimal solution is high because of the iterative calculations. Improving its efficiency with appropriate methods, such as parallelization, can make the algorithm more useful especially in the scheduling problems. In addition, how to choose appropriate control parameters for different optimization issues is a problem in ABC and improved algorithms as with other metaheuristic algorithms. This problem has not been investigated sufficiently in the literature. Therefore, designing a general principle for tuning the control parameters of ABC can be addressed as a searching subject in future studies.

Acknowledgments

This work was supported by International joint research center for logistics management and engineering in Zhongyuan University of Technology, as well as the High-end foreign expert program of Ministry of Science and Technology, grant number G2021026006L; the Training Program for Young Teachers in Universities of Henan Province, grant number 2020GGJS137; the NSFC-Zhejiang Joint Fund for the Integration of Industrialization and Informatization, grant number U1709215; the Zhejiang Province Key R & D projects, grant number No. 2019C03104; the Key Scientific Research Projects of Henan Province, grant number 22A413011; Henan Province Science and Technology R & D projects, grant number 202102210135, 212102310547 and 212102210080; National Nature Science Foundation of China, grant number U1813201, 72101033 and 71831001; Beijing Natural Science Foundation Project grant number KZ202210037046; Canal Plan-Youth Top-notch Talent Project of Beijing Tongzhou District, grant number YHQN2017014.

Conflict of interest

The authors declare there are no conflicts of interest.

References

1. D. W. Alexander, R. Merkert, Challenges to domestic air freight in Australia: evaluating air traffic markets with gravity modelling, *J. Air Transp. Manage.*, **61** (2017), 41–55. https://doi.org/10.1016/j.jairtraman.2016.11.008
2. C. H. Guo, Research on application of scheduling optimization of ETV based on improved genetic algorithm, *Logist. Sci-Tech*, **38** (2015), 61–69. https://doi.org/10.13714/j.cnki.1002-3100.2015.10.019
3. J. D. Qiu, Z. Y. Jiang, M. N. Tang, Research and application of NLAPSO algorithm to ETV scheduling optimization in airport cargo terminal, *J. Lanzhou Jiaotong Univ.*, **34** (2015), 65–70. https://doi.org/10.3969/j.issn.1001-4373.2015.01.013

4. B. Lei, Study on two-ETV task scheduling of airport cargo terminal based on expert system, *Logist. Sci-Tech*, **38** (2015), 13–16. https://doi.org/10.13714/j.cnki.1002-3100.2015.03.004
5. F. Ding, X. J. Song, Application of shared fitness particle swarm in double ETV system, *Comput. Meas. Control*, **26** (2018), 228–247. https://doi.org/10.16526/j.cnki.11-4762/tp.2018.11.050
6. H. Q. Wang, J. H. Wei, S. J. Wen, H. N. Yu, X. G. Zhang, Improved artificial bee colony algorithm and its application in classification, *J. Rob. Mechatron.*, **30** (2018), 921–926. https://doi.org/10.20965/jrm.2018.p0921
7. L. Z. Cui, G. H. Li, Y. L. Luo, F. Chen, Z. Ming, N. Lu, et al., An enhanced artificial bee colony algorithm with dual-population framework, *Swarm Evol. Comput.*, **43** (2018), 184–206. https://doi.org/10.1016/j.swevo.2018.05.002
8. L. Z. Cui, G. H. Li, Z. X. Zhu, Q. Z. Lin, Z. K. Wen, N. Lu, et al., A novel artificial bee colony algorithm with an adaptive population size for numerical function optimization, *Inf. Sci.*, **414** (2017), 53–67. https://doi.org/10.1016/j.ins.2017.05.044
9. D. Karaboga, B. Basturk, Artificial bee colony optimization algorithm for solving constrained optimization problems, in *Foundations of Fuzzy Logic and Soft Computing*, (2007), 789–798. https://doi.org/10.1007/978-3-540-72950-1_77
10. Y. C. Li, J. Wang, L. B. Liu, J. Zhao, Improved artificial bee algorithm for reliability-based optimization of truss structures, *Open Civ. Eng. J.*, **11** (2017), 235–243. https://doi.org/10.2174/1874149501711010235
11. K. P. Luo, A hybrid binary artificial bee colony algorithm for the satellite photograph scheduling problem, *Eng. Optim.*, **52** (2019), 1421–1440. https://doi.org/10.1080/0305215X.2019.1657113
12. A. K. Alazzawi, H. Rais, S. Basri, Y. A. Alsariera, PhABC: A hybrid artificial bee colony strategy for t-way test set generation with constraints support, in *2019 IEEE Student Conference on Research and Development*, (2019), 106–111. https://doi.org/10.1109/scored.2019.8896324
13. F. Weidinger, Picker routing in rectangular mixed shelves warehouses, *Comput. Oper. Res.*, **95** (2018), 139–150. https://doi.org/10.1016/j.cor.2018.03.012
14. J. J. Zhou, X. F. Yao, A hybrid artificial bee colony algorithm for optimal selection of QoS-based cloud manufacturing service composition, *Int. J. Adv. Manuf. Technol.*, **88** (2017), 3371–3387. https://doi.org/10.1007/s00170-016-9034-1
15. G. Chen, P. Sun, J. Zhang, Repair strategy of military communication network based on discrete artificial bee colony algorithm, *IEEE Access*, **8** (2020), 73051–73060. https://doi.org/10.1109/ACCESS.2020.2987860
16. M. Ghanem, A. Jantan, A novel hybrid artificial bee colony with monarch butterfly optimization for global optimization problems, in *First EAI International Conference on Computer Science and Engineering*, (2017), 27–38. http://dx.doi.org/10.4108/eai.27-2-2017.152257
17. X. Chen, X. Wei, G. X. Yang, W. L. Du, Fireworks explosion based artificial bee colony for numerical optimization, *Knowledge-Based Syst.*, **188** (2020), 105002. https://doi.org/10.1016/j.knosys.2019.105002
18. P. J. Gaidhane, M. J. Nigam, A hybrid grey wolf optimizer and artificial bee colony algorithm for enhancing the performance of complex systems, *J. Comput. Sci.*, **27** (2018), 284–302. https://doi.org/10.1016/j.jocs.2018.06.008
19. Z. P. Liang, K. F. Hu, Q. X. Zhu, Z. X. Zhu, An enhanced artificial bee colony algorithm with adaptive differential operators, *Appl. Soft Comput.*, **58** (2017), 480–494. https://doi.org/10.1016/j.asoc.2017.05.005

20. F. Y. Xu, H. L. Li, C. M. Pun, H. D. Hu, Y. J. Li, Y. R. Song, et al., A new global best guided artificial bee colony algorithm with application in robot path planning, *Appl. Soft Comput.*, **88** (2020), 106037. https://doi.org/10.1016/j.asoc.2019.106037
21. X. Y. Song, M. Zhao, Q. F. Yan, S. G. Xing, A high-efficiency adaptive artificial bee colony algorithm using two strategies for continuous optimization, *Swarm Evol. Comput.*, **50** (2019), 100549. https://doi.org/10.1016/j.swevo.2019.06.006
22. W. F. Gao, Z. F. Wei, Y. T. Luo, J. Cao, Artificial bee colony algorithm based on parzen window method, *Appl. Soft Comput.*, **74** (2019), 679–692. https://doi.org/10.1016/j.asoc.2018.10.024
23. H. Wang, W. J. Wang, S. Y. Xiao, Z. H. Cui, M. Y. Xu, X. Y. Zhou, Improving artificial bee colony algorithm using a new neighborhood selection mechanism, *Inf. Sci.*, **527** (2020), 227–240. https://doi.org/10.1016/j.ins.2020.03.064
24. S. Q. Zhang, J. F. Teng, J. H. Gu, Artificial bee algorithm based on multi-dimensional greedy search, *Comput. Eng.*, **40** (2014), 189–193. https://doi.org/10.3969/j.issn.1000-3428.2014.11.037
25. W. L. Xiang, X. L. Meng, Y. Z. Li, R. C. He, M. Q. An, An improved artificial bee colony algorithm based on the gravity model, *Inf. Sci.*, **429** (2018), 49–71. https://doi.org/10.1016/j.ins.2017.11.007
26. H. Q. Wang, M. H. Su, R. Zhao, X. B. Xu, H. D. Haasis, J. H. Wei, et al., Improved multi-dimensional bee colony algorithm for airport freight station scheduling, preprint, arXiv:2207.11651.
27. H. Q. Wang, J. H. Wei, M. H. Su, Z. Dong, S. S. Zhang, Task set scheduling of airport freight station based on parallel artificial bee colony algorithm, in *Bio-inspired Computing: Theories and Applications*, (2019), 484–492. https://doi.org/10.1007/978-981-15-3425-6_37
28. J. C. Bansal, A. Gopal, A. K. Nagar, Stability analysis of artificial bee colony optimization algorithm, *Swarm Evol. Comput.*, **41** (2018), 9–19. https://doi.org/10.1016/j.swevo.2018.01.003