

Article

Cargo Terminal Intelligent-Scheduling Strategies Based on Improved Bee Colony Algorithms

Haiquan Wang^{1,*}, Menghao Su², Xiaobin Xu³, Hans-Dietrich Haasis⁴, Ran Zhao¹, Shengjun Wen¹
and Yan Wang⁵

- ¹ Zhongyuan Petersburg Aviation College, Zhongyuan University of Technology, 41 Zhongyuan Road, Zhengzhou 450007, China; zhaoran@zut.edu.cn (R.Z.); wsj@zut.edu.cn (S.W.)
² Faculty of Information Engineering, Zhengzhou University of Industrial Technology, 16 Xueyuan Road, Xinzheng 451100, China; menghaosu@outlook.com
³ School of Automation, Hangzhou Dianzi University, 115 Wenyi Road, Hangzhou 310018, China; xuxiaobin1980@hdu.edu.cn
⁴ Maritime Business and Logistics, University of Bremen, Bibliothekstr. 1, 28359 Bremen, Germany; hdhaasis@gmx.de
⁵ Faculty of Electrical and Engineering, Zhongyuan University of Technology, 41 Zhongyuan Road, Zhengzhou 450007, China; ywang@zut.edu.cn
* Correspondence: wanghq@zut.edu.cn

Abstract: Due to the rapid increase in cargoes and postal transport volumes in smart transportation systems, an efficient automated multidimensional terminal with autonomous elevating transfer vehicles (ETVs) should be established, and an effective cooperative scheduling strategy for vehicles needs to be designed for improving cargo handling efficiency. In this paper, as one of the most effective artificial intelligence technologies, the artificial bee colony algorithm (ABC), which possesses a strong global optimization ability and fewer parameters, is firstly introduced to simultaneously manage the autonomous ETVs and assign the corresponding entrances and exits. Moreover, as ABC has the disadvantage of slow convergence rate, a novel full-dimensional search strategy with parallelization (PfdABC) and a random multidimensional search strategy (RmdABC) are incorporated in the framework of ABC to increase the convergence speed. After being evaluated on benchmark functions, it is applied to solve the combinatorial optimization problem with multiple tasks and multiple entrances and exits in the terminal. The simulations show that the proposed algorithms can achieve a much more desired performance than the traditional artificial bee colony algorithm in terms of balancing the exploitation and exploration abilities, especially when dealing with the cooperative control and scheduling problems.

Keywords: intelligent scheduling; artificial bee colony algorithm; vehicle cooperative control; parallelization; multidimensional search



Citation: Wang, H.; Su, M.; Xu, X.; Haasis, H.-D.; Zhao, R.; Wen, S.; Wang, Y. Cargo Terminal Intelligent-Scheduling Strategies Based on Improved Bee Colony Algorithms. *Appl. Sci.* **2023**, *13*, 8750. <https://doi.org/10.3390/app13158750>

Academic Editors: Daxin Tian, Kuo-Ching Ying, Fei Hui and Wei Shangguan

Received: 4 May 2023

Revised: 8 July 2023

Accepted: 18 July 2023

Published: 28 July 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Nowadays, transport activities around the world are developing rapidly, and the cooperative management of cargo terminals, which are the major gateway for cargo services, has become crucial. Thus, an effective and efficient scheduling and cooperative management strategy considering the route of ETVs and the assignment of entrances and exits should be designed to minimize the time cost for handling all inbound and outbound cargoes [1–3].

Actually, if the numbers of ETVs and tasks are small, the scheduling problem can be regarded as a zero-one integer linear programming problem and can be solved using the simple method, such as the Hungarian algorithm [4], the mixed-integer linear model [5], the enumeration technique [6], the priority rule-based procedure [7], and so on. But for large-scale problems that are difficult to be solved by traditional optimization algorithms, the evolutionary algorithms with high robustness, become more effective. Guo [8] and

Qiu [9] studied the inbound and outbound cargo scheduling problem with a single ETV and solved it with the genetic algorithm and the particle swarm optimization (PSO) algorithm, respectively. PSO was also applied to assign two ETVs to different cargo areas with an improved shared fitness strategy in ref. [10]. The works mentioned above focus on deciding the cargo transportation sequence by considering the picking sequence and ETV routing, but very few studies have discussed the problem of the assignment of entrances and exits if there are several gates in the freight station.

In our research, a swarm intelligent algorithm named the artificial bee colony algorithm (ABC), which possesses a strong global optimization ability and fewer parameters [11–14], is firstly proposed to solve the problem of assigning the entrances and exits as well as autonomous ETVs for several outbound and inbound tasks simultaneously. Actually, as an artificial intelligence algorithm, the ABC has been well adapted for various complex optimization and scheduling problems [15–21]; however, it often suffers from the problem of a slow convergence rate because of its single-dimensional random search strategy in the bee updating phases. To accelerate the convergence speed without reducing the accuracy, other metaheuristic algorithms were introduced and combined with the traditional ABC. Ustun and Toktas [22] combined the mutation and the crossover operators in differential evolution algorithm with the onlooker bee phases to improve the accuracy and speed up the convergence. Aiming at improving the optimization accuracy, combined with the learning characteristics of the Q-learning algorithm, the update dimension in each iteration could be dynamically adjusted in ref. [23]. Xu et al. [24] introduced a differential evolution strategy in the employed bee phase to accelerate its convergence and adopted the global best position to guide the updating processes in the onlooker bee phase, which could enhance the local search ability. The firework explosion search mechanism was introduced to explore the potential food sources of ABC in ref. [25]. A modified search operator was employed in ref. [26] to exploit useful information of the current best solution in the onlooker phase, and the ability of exploitation could be improved. Obviously, most of the improvements are based on the introduced search strategies or the combinations with other algorithms, and there are few systematic analyses and improvements from the perspective of operation mechanisms. Therefore, for balancing the abilities of exploration and exploitation, after modeling the actions of the ETV in a cargo terminal with multiple entrances and exits, improved ABC algorithms with paralleled full-dimensional search strategy and random multidimensional search strategy are proposed in this paper.

The rest of paper is organized as follows: Section 2 introduces how to establish the scheduling model for a cargo terminal. Then, two improved strategies based on the ABC are proposed in Section 3. In Section 4, the improved algorithms are applied to solve the ETV scheduling problem considering multiple entrances and exits. Finally, the above work is summarized.

2. The Scheduling Model of Freight Station

The airport freight station consists of a container storage area, a bulk cargo storage area, and an unhandled cargo area. As the core of the whole station, the container storage area is a three-dimensional warehouse used for handling the containerized cargo which is unloaded from aircraft on the air side or inbounded from the bulk cargo storage area on the land side. The shelves in the warehouse consist of two rows with 16 entrances and exits, and each row has eight layers and 60 columns.

ETV is employed for handling cargoes between different entrances and exits. During the pickup and delivery operations, ETVs experience three stages, which are acceleration, constant speed, and deceleration, in the horizontal and vertical directions. Thus, the time needed to finish a whole delivery process is determined by the maximum value between the horizontal travelling time T_x and the vertical lifting time T_y , which are defined as Equations (1) and (2).

$$\begin{cases} T_x = 2 * \sqrt{\frac{w * u}{a_x}}, & w * u \leq D_x; \\ T_x = T_{xi} + \frac{w * u - D_x}{V_x}, & w * u > D_x. \end{cases} \quad (1)$$

$$\begin{cases} T_y = 2 * \sqrt{\frac{h * e}{a_y}}, h * e \leq D_y; \\ T_y = T_{yj} + \frac{h * e - D_y}{V_y}, h * e > D_y. \end{cases} \quad (2)$$

Here, e and u are the differences of layers and columns between the initial position and the destination, h and w are the height and width of storage location, and a_x, a_y, V_x , and V_y are the accelerations and velocities in horizontal and vertical directions, respectively.

Equations (3) and (4) describe the time costs T_{xi} and T_{yj} , and the travelled distances D_x and D_y , when ETV accelerates to the maximum speed and immediately decelerates to 0 in different directions

$$T_{xi} = 2 * V_x / a_x, T_{yj} = 2 * V_y / a_y \quad (3)$$

$$D_x = \frac{1}{4} * a_x * T_{xi}^2, D_y = \frac{1}{4} * a_y * T_{yj}^2 \quad (4)$$

Based on the motion analysis for ETVs, the time cost T_i needed to execute the i th task, including picking up, releasing, as well as moving the assigned cargoes, is expressed as Equation (5),

$$T_i = T_{i_0} + T_{i_1} + 2\zeta \quad (5)$$

Here, T_{i_0} is the time needed to move from its current position to the cargo's position, T_{i_1} is the time used to move to the destination after an ETV receives the cargo, and ζ is the time used to load or unload the cargo.

Thus, if there are n independent inbound and outbound tasks in the cargo terminal, in order to improve its operational efficiency, the sequence of inbound and outbound tasks considering the actions of the ETV should be scheduled, and the total time cost T defined as Equation (6) should be minimized.

$$\text{Minimize } T = \sum_{i=1}^n T_i = \sum_{i=1}^n (T_{i_0} + T_{i_1}) + 2n\zeta \quad (6)$$

In order to solve the minimization problem mentioned above, an effective optimization algorithm should be introduced.

3. Methodologies

3.1. ABC Algorithm

As a typical swarm intelligence algorithm, ABC simulates the foraging behaviors of natural bees, where a food source represents a solution and its fitness is measured in terms of nectar amount.

The algorithm begins with a randomly distributed initial population generation and evaluation [11]. Then, repeated search cycles are executed to update the optimal solution. During the cycles, as shown in Equation (7), employed bee probabilistically produces neighbor food source x'_{ik} around the current optimal solution x_{ik} .

$$x'_{ik} = x_{ik} + \text{rand}(-1, 1) * (x_{ik} - x_{jk}) \quad (7)$$

Here, i and j represent the numbers of the solutions, $i, j \in \{1, 2, \dots, N\}$, $i \neq j$. k is the dimension of the solution, which is selected randomly, $k \in \{1, 2, \dots, D\}$.

Subsequently, the m th onlooker bee randomly chooses to exploit or not around corresponding employed bee with the probability P_m defined as Equation (8).

$$P_m = \frac{\text{fitness}_m}{\sum_{m=1}^{SN} \text{fitness}_m} \quad (8)$$

If the current solution to be exploited cannot be improved for several iterations, it will be abandoned, and a randomly produced scout bee will replace it.

3.2. The Improved ABC Algorithms

Obviously, in the ABC algorithm, a random single dimensional search is executed, which means only one dimension will be randomly selected and updated according to Equation (7) in either the employed bee phase or onlooker bee phase. In this case, the updated dimension may be different in each iteration and the optimal dimension obtained in the previous iterations is likely to be omitted in the following iterations. Thus, the search toward the possible optimal solution is unable to be continued, and the optimization accuracy as well as the convergence speed will be affected. To improve the probability of obtaining the optimal solution without any influence on the convergence speed, several improvements are introduced in this section.

3.2.1. Paralleled Full-Dimensional ABC Algorithm

Different from randomly updating only one dimension in traditional ABC algorithm, the full-dimensional search strategy (fdABC) is introduced in this section, where all dimensions of the solution are traversed with Equation (7) and the optimal dimension is kept for further exploration, therefore the search could be extended and the possibility of obtaining the optimal solution will be improved, but the cost of time will increase inevitably. In order to improve its efficiency, a master-slave parallel mode [27] is applied to the most time-consuming stages, such as the calculation of initial fitness values and the updating procedure in the employed bee phase, in which the population is divided into different parts and the repeated calculations are executed in multi-core processor. Thus the corresponding tasks could be finished in parallel, meanwhile the process of the ABC algorithm will not be affected. The following are the main steps of PfdABC algorithm:

Step 1: Initialization. The parameters of ABC, including the food source, the population, and the maximum number of iterations are initialized, and the initial population is divided into different parts. Then, each of them is evaluated in different CPU cores.

Step 2: Employed bee phase. A full-dimensional search is performed in this phase. All the employed bees are equally distributed into different CPU cores, and the neighborhood searches using Equation (7) are executed in all dimensions, where k varies from zero to the number of dimensions. If the fitness value of the updated solution is better than the previous one, it will be preserved for further searches.

Step 3: Onlooker bee phase. Onlooker bee selects a food source with Equation (8) and executes a full-dimensional search around the selected solution.

Step 4: Scout bee phase. If the number of iteration reaches the threshold and there is no better solution, a new solution will be generated randomly.

Step 5: Record the global optimal solution obtained so far and jump to step 2 for further exploration until reaching the maximum iteration number.

3.2.2. Random Multidimensional Artificial Bee Colony Algorithm

The PfdABC algorithm mentioned above travels all the dimensions in parallel, which could improve the optimization accuracy and efficiency. In this part, another algorithm named random multidimensional artificial bee colony (RmdABC) algorithm is proposed to balance the abilities of exploration and exploitation. The key improvement of the strategy is to randomly select several dimensions from $\{1, 2, \dots, D\}$, and execute corresponding updating cycles with Equation (7) in the related dimensions. The number of updating cycles for one solution is equal to the number of selected dimensions, that is, it could be one to D . As RmdABC randomly traverses any number of dimensions for each solution, fewer dimensions are updated compared to PfdABC, and its time complexity could be greatly improved. On the other hand, RmdABC covers more dimensions compared to ABC, and the likelihood of obtaining the optimal solution could be enhanced. The pseudo-code of RmdABC is shown as Algorithm 1.

Algorithm 1: Pseudo-code of RmdABC.

```

01:    //Initialization,
      set the maximum number of iterations, the swarm size  $N$ , the number of dimension  $D$ 
02:    //Employed bee phase
      for  $i = 1$  to  $FoodNumber$ 
03:         $flag = 0$ ;
04:         $Random\_D = randi(D)$ ; //Generate a random sequence
05:        //Random multi-dimensional greedy search strategy
        for  $j = 1$  to  $Random\_D$ 
06:            produce candidate solution with Equation (7), evaluate its fitness value;
07:            if  $fitness(Sol_i) < fitness(Food_i)$  then  $Food_i = Sol_i$  and  $flag = 1$ ;
08:        end if
09:    end for
10:    if  $flag = 1$  then  $trial = 0$ ; else  $trial + 1$ ;
11:    end if
12: end for
13: //Onlooker bee phase
      According to Equation (8), calculate the probability  $prob_i$  and determine if the onlooker
      bee chooses to exploit or not around the  $i$ th employed bee
14: for  $i = 1$  to  $FoodNumber$ 
15:      $flag = 0$ ;
16:     if  $rand < prob_i$ 
17:         produce the candidate solution with Equation (7) and evaluate its fitness value;
18:         if  $fitness(Sol_i) < fitness(Food_i)$  then  $Food_i = Sol_i$  and  $flag = 1$ ;
19:     end if
20:     if  $flag = 1$  then  $trial = 0$ ; else  $trial + 1$ ;
21: end if
22: end for
23: //Scout bee phase
      if  $trial > Limit$ 
24:      $trial = 0$ ;
25:     Randomly generate a solution;
26: end if
27: end for

```

As stated above, more dimensions are explored in fdABC compared to ABC, but more time is needed. With the master-slave parallel strategy, calculations could be executed in different CPU cores, and the time for optimization could be reduced; through random dimensional selection, less dimensions will be explored, thus the efficiency could be improved. Therefore, the two proposed strategies could effectively balance the abilities of exploration and exploitation.

4. Implementation and Experimental Results

To evaluate the performance of the proposed algorithms, we consider two cases, which are the benchmark functions and the task-set scheduling problem in an air cargo terminal.

Corresponding experiments are executed using MATLAB on a computer with Inter(R) Core (TM) i7-8750h CPU @2.20Ghz, 16 GB of memory. The parameters of the PSO and ABC algorithms are set as follows: The swarm size of the corresponding algorithms is set to 200; the maximum number of local searches is 100; the maximum number of iterations is equal to 1500; cognitive and social components of PSO are both set to 1.8; and the inertia weight, which determines how the previous velocity of the particle influences the velocity in the next iteration, is 0.6, as recommended in [28].

4.1. Verification with Benchmark Functions

CEC benchmark functions as shown in Table 1 are introduced, and ABC, fdABC, PfdABC, as well as RmdABC are applied to solve them with MATLAB. The algorithms are

executed twenty times, and the simulation statistical results, including average runtime, shortest runtime, average optimal results, best optimization solutions, and their variances, are concluded in Tables 2–4, corresponding to different dimensions.

Table 1. Benchmark functions.

The Name of Function	Function Expression	Search Space	Minimum Value
1. Bent Cigar Function	$f_1(x) = x_1^2 + 10^6 \sum_{i=2}^D x_i^2$	$[-100, 100]^D$	0
2. Sum of Different Power Function	$f_2(x) = \sum_{i=1}^D x_i ^{i+1}$	$[-100, 100]^D$	0
3. Rosenbrock's Function	$f_3(x) = \sum_{i=1}^{D-1} (100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2)$	$[-100, 100]^D$	0
4. Rastrigin's Function	$f_4(x) = \sum_{i=1}^n [x_i^2 - 10 \cos(2\pi x_i) + 10]$	$[-500, 500]^D$	0
5. Step's Function	$f_5(x) = \sum_{i=1}^n (x_i + 0.5)^2$	$[-100, 100]^D$	0

Table 2. The performance of PfdABC, RmdABC, fdABC, and ABC on 60 dimensions.

Function Name	Algorithm	Average Runtime/s	Average Optimal Result/s	Best Optimal Result/s	Shortest Runtime/s	Variance of Optimal Result/s
1. Bent Cigar Function	ABC	111.849	2.320×10^5	1.709×10^5	111.4955	2.039×10^9
	fdABC	253.316	2.929×10^{-252}	1.416×10^{-252}	251.577	0
	RmdABC	188.172	1.654×10^{-2}	1.49×10^{-4}	179.199	2.28×10^{-4}
	PfdABC	256.408	2.995×10^{-16}	1.028×10^{-16}	179.199	1.055×10^{-32}
2. Sum of Different Power Function	ABC	121.258	3.0245×10^{41}	1.593×10^{36}	120.91	4.433×10^{83}
	fdABC	676.282	8.680×10^{-255}	7.047×10^{-258}	672.29	0
	RmdABC	401.083	2.091×10^{-42}	6.660×10^{-46}	395.373	1.791×10^{-83}
	PfdABC	490.647	4.164×10^{-89}	1.902×10^{-90}	488.905	4.053×10^{-177}
3. Rosenbrock's Function	ABC	116.400	5490.448	4884.468	115.536	2.083×10^5
	fdABC	284.881	2.33×10^{-3}	1.27×10^{-4}	279.176	2.021×10^{-6}
	RmdABC	212.479	1.1359	0.323	208.025	1.013
	PfdABC	288.203	0.140	0.0072	286.3592	0.0245
4. Rastrigin's Function	ABC	121.356	198.642	179.002	121.051	163.148
	fdABC	250.787	0	0	245.425	0
	RmdABC	190.324	4.145×10^{-6}	2.271×10^{-7}	189.681	2.759×10^{-11}
	PfdABC	279.681	2.956×10^{-13}	2.274×10^{-13}	272.530	3.447×10^{-27}
5. Step's Function	ABC	120.309	0.232	0.177	119.949	0.843×10^{-4}
	fdABC	224.032	0	0	218.470	0
	RmdABC	178.819	1.262×10^{-8}	4.285×10^{-10}	178.200	1.121×10^{-16}
	PfdABC	266.942	3.298×10^{-22}	1.464×10^{-22}	255.218	1.416×10^{-44}

Table 3. The performance of PfdABC, RmdABC, fdABC, and ABC on 80 dimensions.

Function Name	Algorithm	Average Runtime/s	Average Optimal Result/s	Best Optimal Result/s	Shortest Runtime/s	Var Optimal Result/s
1. Bent Cigar Function	ABC	186.356	7.443×10^7	6.612×10^7	185.785	8.560×10^{13}
	fdABC	349.896	9.500×10^{-25}	1.073×10^{-251}	341.500	0
	RmdABC	254.843	1.830	0.187	242.967	8.657926
	PfdABC	336.140	3.739×10^{-11}	2.729×10^{-11}	330.117	6.778×10^{-23}
2. Sum of Different Power Function	ABC	161.169	7.666×10^{82}	9.065×10^{73}	157.306	5.875×10^{116}
	fdABC	1092.133	5.377×10^{-250}	9.849×10^{-253}	1083.415	0
	RmdABC	641.927	1.236×10^{-18}	4.733×10^{-22}	629.479	1.193259×10^{-35}
	PfdABC	767.770	9.062×10^{-72}	2.578×10^{-72}	761.159	6.124×10^{-143}
3. Rosenbrock's Function	ABC	153.868	65,508.192	53,885.588	152.849	7.237×10^7
	fdABC	408.311	0.00517	0.000291	393.921	3.638×10^{-5}
	RmdABC	297.845	2.626	1.1049	287.474	1.758
	PfdABC	388.867	0.264	0.041	380.135	0.064

Table 3. Cont.

Function Name	Algorithm	Average Runtime/s	Average Optimal Result/s	Best Optimal Result/s	Shortest Runtime/s	Var Optimal Result/s
4. Rastrigin's Function	ABC	155.747	759.215	687.486	153.785	2578.573
	fdABC	351.982	0	0	346.306	0
	RmdABC	275.060	0.0184	0.0056	270.149	9.127×10^{-5}
	PfdABC	358.622	4.206×10^{-13}	2.274×10^{-13}	353.856	5.888×10^{-27}
5. Step's Function	ABC	151.843	8.142	6.349	151.444	0.965
	fdABC	300.048	0	0	297.340	0
	RmdABC	537.590	6.049×10^{-7}	2.1807×10^{-8}	373.611	3.606×10^{-13}
	PfdABC	333.573	3.940×10^{-17}	2.531×10^{-17}	323.896	9.741×10^{-35}

Table 4. The performance of PfdABC, RmdABC, fdABC and ABC on 100 dimensions.

Function Name	Algorithm	Average Runtime/s	Average Optimal Result/s	Best Optimal Result/s	Shortest Runtime/s	Var Optimal Result/s
1. Bent Cigar Function	ABC	223.196	3.787×10^8	3.416×10^8	222.774	1.099×10^{15}
	fdABC	425.692	2.044×10^{-251}	1.091×10^{-251}	420.020	0
	RmdABC	303.243	12.037	0.405	300.169	199.478
	PfdABC	399.895	4.392×10^{-8}	3.463×10^{-8}	387.853	6.669×10^{-17}
2. Sum of Different Power Function	ABC	199.808	2.279×10^{120}	7.215×10^{111}	197.343	2.222×10^{241}
	fdABC	1664.371	1.671×10^{-245}	6.863×10^{-248}	1619.398	0
	RmdABC	920.823	3.470×10^{-6}	4.151×10^{-9}	911.149	1.069×10^{-10}
	PfdABC	1120.629	8.680×10^{-57}	3.9115×10^{-58}	1115.067	5.118×10^{-113}
3. Rosenbrock's Function	ABC	189.997	5.919×10^3	4.346×10^5	186.093	9.399×10^9
	fdABC	538.976	0.00884	0.001	504.637	1.04×10^{-4}
	RmdABC	351.792	6.042937	1.430	337.071	18.386
	PfdABC	456.292	1.021	0.201	449.671	1.568
4. Rastrigin's Function	ABC	193.990	152.642	193.287	193.2873	65.193
	fdABC	458.470157	0	0	452.714	0
	RmdABC	574.398	7.97×10^{-4}	8.599×10^{-5}	573.802	7.859×10^{-7}
	PfdABC	454.618	1.136×10^{-10}	8.356×10^{-11}	450.936	2.190×10^{-22}
5. Step's Function	ABC	189.754	78.395	63.6545	187.726	97.986519
	fdABC	637.867	0	0	622.008	0
	RmdABC	289.625	1.379×10^{-7}	1.838×10^{-8}	287.601	1.87107×10^{-14}
	PfdABC	398.953	5.632×10^{-14}	2.467×10^{-14}	391.252	3.979×10^{-28}

The results show that the average runtime and shortest runtime on all benchmark functions corresponding to PfdABC, fdABC, and RmdABC increase, but the quality of the optimal solution as well as the local-searching ability have been improved significantly compared to ABC, meanwhile the optimal variances on different dimensions show that the stabilities have been greatly improved with the three proposed strategies.

Figure 1 graphically shows the fitness values for solving Expanded Schaffer's Function with different dimensions. Obviously, fdABC, PfdABC, and RmdABC algorithms possess faster convergence speeds and better accuracies compared with ABC.

Based on the results above, the introduction of the random multidimensional strategy and parallel full-dimensional strategy greatly improves the performance of traditional ABC. fdABC possesses the best optimization result, but it spends too much time on optimizing. The optimization solutions corresponding to PfdABC and RmdABC are not as good as fdABC, but they can reduce the optimization time effectively. The time cost of RmdABC is lower than fdABC and PfdABC, as RmdABC randomly selects part of dimensions in each iteration. Obviously, the proposed PfdABC and RmdABC algorithms could balance the abilities of exploration and exploitation compared to ABC and fdABC, and the corresponding algorithms must be selected during application on the basis of the practical problems.

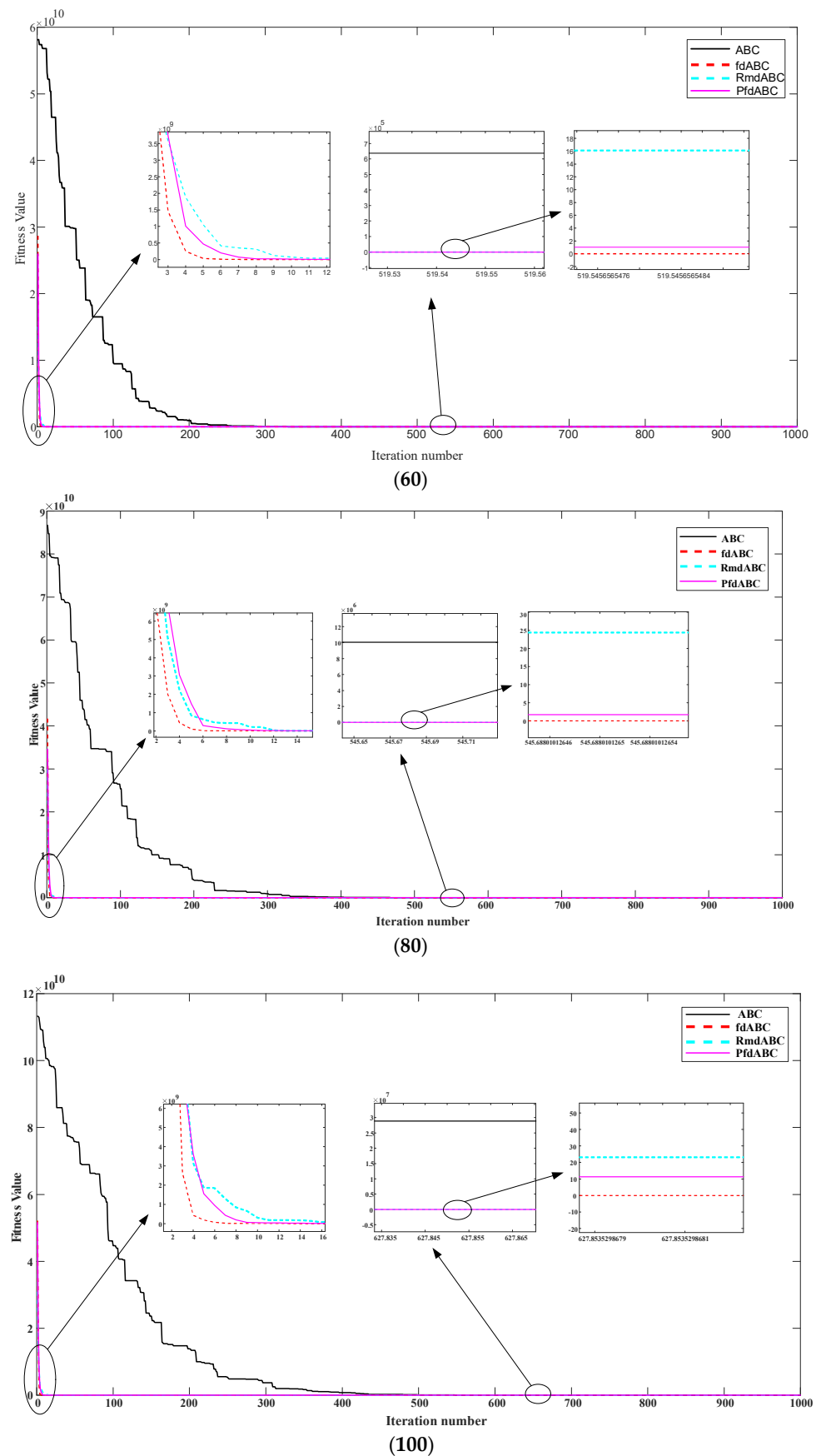


Figure 1. The fitness values with different strategies for Expanded Schaffer's Function.

4.2. Scheduling Problem

The cargo terminal mentioned in this section is the northern freight area in Xinzheng international airport, and there are totally 16 entrances and exits in total in the container storage area. The coordinates of the nine entrances are $R_1(1-1-5)$, $R_2(2-1-15)$, $R_3(1-1-20)$, $R_4(1-1-25)$, $R_5(1-1-30)$, $R_6(1-1-35)$, $R_7(1-1-40)$, $R_8(1-1-50)$, and $R_9(1-1-60)$, and the coordinates of exits are $C_1(1-1-8)$, $C_2(1-1-18)$, $C_3(1-1-28)$, $C_4(1-1-38)$, $C_5(1-1-48)$, $C_6(2-1-53)$, and $C_7(1-1-58)$ respectively, where the first value in the bracket represents the row number of shelf, the second value indicates the number of layers, and the third value is the number of columns. Sixty tasks are waiting to be scheduled, where the first thirty tasks are inbound and the later thirty tasks are outbound. The assigned positions are listed as Table 5.

Table 5. Task sets to be scheduled.

Inbound Tasks		Inbound Tasks		Outbound Tasks		Outbound Tasks	
1	I(1-5-34)	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(1-3-58)	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(1-2-55)	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(2-8-10)

For solving the optimal scheduled sequence, a sort mapping coding (SMC) strategy is introduced to establish the mapping relationship between the solution and optimization algorithm. It assigns a random number to each dimension of the solution, then sorts the numbers in ascending order and assigns index values based on their sequence. The resulting sequence represented by the index values is the scheduling scheme. The values obtained with SMC could be updated based on Equation (7) with the help of ABC, where the number of dimensions to be updated is determined by different algorithms: ABC algorithm only randomly selects one dimension, all dimensions can be updated in PfdABC, and RmdABC randomly selects certain dimensions.

For the above scheduling problem, comparative studies among five algorithms, including PSO, ABC, PfdABC, and RmdABC, are executed. The scheduling program ran 20 times, and Tables 6–9 show the optimization results, including the convergence iterations and task execution time corresponding to the optimal solution, the resulting sequences, and the allocation plans of entrances and exits. Obviously, all swarm intelligent algorithms could solve the complicated cargo scheduling problem in certain iterations.

Table 6. (a) The scheduling results with PSO. (b) Entrances and exits allocation scheme with PSO.

(a)										
	1	2	3	4	5	6	7	8	9	10
Iteration	889	654	805	841	912	735	889	903	836	749
Result/s	6841.9	6811.9	6828.4	6779.6	6854.9	6819.5	6841.9	6866.1	6826.1	6811.9
	11	12	13	14	15	16	17	18	19	20
Iteration	889	654	901	912	912	905	697	903	827	749
Result/s	6841.9	6811.9	6828.4	6854.9	6854.9	6819.5	6757.4	6866.1	6826.1	6828.4

Table 6. *Cont.*

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

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

Table 7. (a) The scheduling results with ABC. (b) Entrances and exits allocation scheme with ABC.

(a)										
	1	2	3	4	5	6	7	8	9	10
Iteration	1098	872	725	915	1003	1000	697	605	835	898
Result/s	6852.9	6772.4	6806.41	6697.15	6892.79	6905.64	6748.78	6865.9	6787.76	6803
	11	12	13	14	15	16	17	18	19	20
Iteration	732	1079	509	840	602	675	786	821	753	835
Result/s	6839.3	6824.9	6877.64	6796.28	6787.27	6864.15	6808.01	6871.29	6839.52	6817.27

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

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

Table 8. (a) The scheduling results with PfdABC. (b) Entrances and exits allocation scheme with PfdABC.

(a)										
	1	2	3	4	5	6	7	8	9	10
Iteration	488	230	223	501	179	346	267	464	202	595
Result/s	6621.11	6623.11	6613.61	6609.98	6617.36	6617.24	6617.36	6611.73	6617.12	6615.49
	11	12	13	14	15	16	17	18	19	20
Iteration	213	464	228	82	82	536	153	524	422	388
Result/s	6606.35	6617.48	6630.49	6615.37	6611.86	6617.36	6617.48	6619.24	6615.37	6611.74

Table 8. *Cont.*

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

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

Table 9. (a) The scheduling results with RmdABC. (b) Entrances and exits allocation scheme with RmdABC.

(a)										
	1	2	3	4	5	6	7	8	9	10
Iteration	209	209	213	209	223	546	580	209	331	423
Result/s	6617.12	6617.12	6611.86	6617.12	6611.86	6611.86	6615.61	6617.12	6617.24	6617.36
	11	12	13	14	15	16	17	18	19	20
Iteration	539	191	209	223	529	580	546	223	209	223
Result/s	6621.11	6611.86	6617.12	6611.86	6615.61	6615.61	6615.61	6611.86	6617.12	6611.86

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

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

The statistical information of the above results are described in the box plot as Figure 2. The numbers 1, 2, 3, and 4 in horizontal coordinate represent PSO, ABC, PfdABC, and RmdABC, respectively. The plot shows that PfdABC and RmdABC are better as the upper limits are lower than the lower limits of PSO and ABC, PfdABC possesses higher optimization accuracy because of its lower low limit, and RmdABC is more stable as the distance between the lower and upper limits is shorter.

In addition, several important indices, including the average time required to execute the resulting sequence (Avg), the shortest executed time (Min), the longest inbound and outbound time (Max), and the average convergence generations (Ite) are calculated based on the results of 20 independent runs and shown in Table 10. It can be easily found that the

minimum, maximum, and average inbound and outbound time corresponding to PfdABC and RmdABC decreased by 4% at most compared to conventional ABC and PSO, and there are relatively small gaps among the values of Avgs. On the other hand, by examining the values of Ite, it can be seen that PfdABC and RmdABC require less generations to locate the optimal solutions and that it improves by almost 60% compared to ABC and PSO, that is, the advantage of fast convergence speed could be proved.

Table 10. The scheduling results.

	Min (s)	Max (s)	Avg (s)	Ite
PSO	6757.4	6854.9	6828.6	828
ABC	6697.14	6905.64	6822.92	814
PfdABC	6606.35	6630.49	6616.34	329
RmdABC	6611.86	6621.11	6615.19	332

Therefore, PfdABC and RmdABC can obtain faster convergence speed and smaller mean values than those of ABC and PSO in the scheduling problem, and they are demonstrated to be useful tools for solving the air cargo terminal scheduling problem.

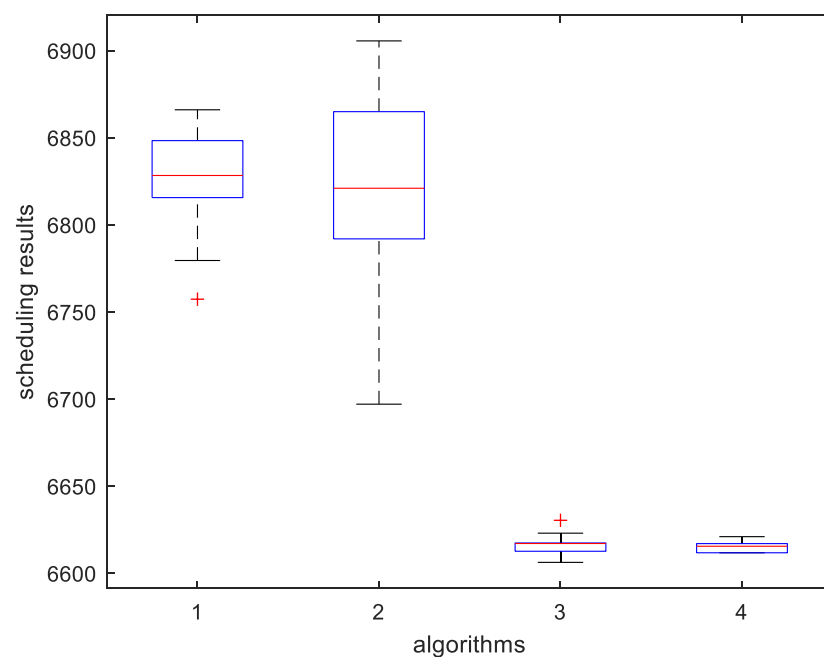


Figure 2. Box plot of corresponding algorithms.

5. Conclusions

To improve the efficiency of the cargo terminal in a smart transport system, the intelligent ABC algorithm was introduced to schedule the task set in this paper. Moreover, for increasing the diversity of the population while accelerating the convergence, improved algorithms, including PfdABC and RmdABC, are proposed to enhance the optimization performance. The experimental results show that the ABC, PfdABC, and RmdABC algorithms can solve the task-sets scheduling problem and that the proposed RmdABC and PfdABC algorithms could improve the exploration and exploitation performance efficiently.

The proposed algorithms are efficient in solving complex optimization problems, but their ability of solving dynamic scheduling problems should be verified, and the adaptation of control parameters of ABC for different scheduling requirements should be addressed.

Author Contributions: Conceptualization, H.W. and M.S.; funding acquisition, S.W. and H.W.; investigation and methodology, X.X. and H.W.; supervision, Y.W.; writing—original draft, H.W., Y.W., H.-D.H. and R.Z.; writing—review and editing, H.W., H.-D.H. and S.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Training Program for Young Teachers in Universities of Henan Province (No. 2020GGJS137) and the Key Technologies R&D Program of Henan (Nos. 222102210016, 232102211020 and 222102210019).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data can be obtained from the corresponding author upon reasonable request.

Acknowledgments: The authors would like to thank Ping Liu, Hongjun Li and Jianhua Wei for their valuable discussions and feedback.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Tang, L.C.; Ng, T.S.; Lam, S.W. Improving air cargo service through efficient order release. In Proceedings of the 7th International Conference on Service Systems and Service Management, Tokyo, Japan, 28–30 June 2010.
2. Wu, P.J.; Yang, C.K. Sustainable development in aviation logistics: Successful drivers and business strategies. *Bus. Strategy Environ.* **2021**, *30*, 3763–3771. [\[CrossRef\]](#)
3. Gong, Q.; Wang, K. International trade drivers and freight network analysis: The case of the Chinese air cargo sector. *J. Transp. Geogr.* **2018**, *71*, 253–262. [\[CrossRef\]](#)
4. Kuhn, H.W. The Hungarian method for the assignment problem. *Nav. Res. Logist. Q.* **1955**, *2*, 83–97. [\[CrossRef\]](#)
5. Hu, W.; Mao, J.; Wei, K. Energy-efficient Dispatching Solution in an Automated Air Cargo Terminal. In Proceedings of the IEEE International Conference on Automation Science and Engineering, Madison, WI, USA, 17–20 August 2013; pp. 144–149.
6. Hamzaoui, M.A.; Sari, Z. Optimal dimensions minimizing expected travel time of a single machine flow rack AS/RS. *Mechatronics* **2015**, *31*, 158–168. [\[CrossRef\]](#)
7. Boysen, N.; Fedtke, S.; Weidinger, F. Optimizing automated sorting in warehouses: The minimum order spread sequencing problem. *Eur. J. Oper. Res.* **2019**, *270*, 386–400. [\[CrossRef\]](#)
8. Guo, C.H. Research on application of scheduling optimization of ETV based on improved genetic algorithm. *Logist. Sci.-Tech.* **2015**, *38*, 61–69.
9. Qiu, J.D.; Jiang, Z.Y.; Tang, M.N. Research and application of NLPSO algorithm to ETV scheduling optimization in airport cargo terminal. *J. Lanzhou Jiaotong Univ.* **2019**, *34*, 65–70.
10. Ding, F.; Song, X.J. Application of shared fitness particle swarm in double ETV system. *Comput. Meas. Control.* **2018**, *26*, 228–247.
11. Radhwan, A.A.; Saleh, R.A. Artificial bee colony algorithm with directed scout. *Soft Comput.* **2021**, *25*, 13567–13593.
12. Dokeroglu, T.; Sevinc, E.; Cosar, A. Artificial bee colony optimization for the quadratic assignment problem. *Appl. Soft Comput.* **2019**, *76*, 595–606. [\[CrossRef\]](#)
13. Awadallah, M.A.; Betar, M.A.; Bolaji, A.L.; Doush, I.A.; Hammouri, A.I.; Mafarja, M. Island artificial bee colony for global optimization. *Soft Comput.* **2020**, *24*, 13461–13487. [\[CrossRef\]](#)
14. Roeva, O.; Petkova, D.; Castillo, O. Joint set-up of parameters in genetic algorithms and the artificial bee colony algorithm: An approach for cultivation process modelling. *Soft Comput.* **2021**, *25*, 2015–2038. [\[CrossRef\]](#)
15. Forouzandeh, S.; Berahmand, K.; Nasiri, E.; Rostami, M. A Hotel Recommender System for Tourists Using the Artificial Bee Colony Algorithm and Fuzzy TOPSIS Model: A Case Study of Trip Advisor. *Int. J. Inf. Technol. Decis. Mak.* **2021**, *20*, 399–429. [\[CrossRef\]](#)
16. Li, Y.C.; Liu, L.B.; Wang, J.; Zhao, J. Improved artificial bee algorithm for reliability-based optimization of truss structures. *Open Civ. Eng. J.* **2018**, *11*, 235–243.
17. Luo, K.P. A hybrid binary artificial bee colony algorithm for the satellite photograph scheduling problem. *Eng. Optim.* **2020**, *52*, 1421–1440. [\[CrossRef\]](#)
18. Alazzawi, A.; Rais, H.M.; Basri, S. Hybrid artificial bee colony strategy for t-way test set generation with constraints support. In Proceedings of the IEEE Student Conference on Research and Development, Bandar Seri Iskandar, Malaysia, 15–17 October 2019.
19. Weidinger, F. Picker routing in rectangular mixed shelves warehouses. *Comput. Oper. Res.* **2018**, *95*, 139–150. [\[CrossRef\]](#)
20. Zhou, J.J.; Yao, X.F. A hybrid artificial bee colony algorithm for optimal selection of QoS-based cloud manufacturing service composition. *Int. J. Adv. Manuf. Technol.* **2017**, *88*, 3371–3387. [\[CrossRef\]](#)
21. Chen, G.; Sun, P.; Zhang, J. Repair strategy of military communication network based on discrete artificial bee colony algorithm. *IEEE Access* **2020**, *8*, 73051–73060. [\[CrossRef\]](#)

22. Ustun, D.; Toktas, A.; Erkan, U.; Akdagli, A. Modified artificial bee colony algorithm with differential evolution to enhance precision and convergence performance. *Expert Syst. Appl.* **2022**, *198*, 116930. [\[CrossRef\]](#)
23. Long, X.J.; Zhang, J.T.; Zhou, K.; Jin, T.G. Dynamic Self-Learning Artificial Bee Colony Optimization Algorithm for Flexible Job-Shop Scheduling Problem with Job Insertion. *Processes* **2022**, *10*, 571. [\[CrossRef\]](#)
24. Xu, F.Y.; Li, H.L.; Pun, C.M.; Hu, H.D. A new global best guided artificial bee colony algorithm with application in robot path planning. *Appl. Soft Comput.* **2020**, *88*, 106037. [\[CrossRef\]](#)
25. Chen, X.; Wei, X.; Yang, G.X.; Du, W.L. Fireworks explosion based artificial bee colony for numerical optimization. *Knowl.-Based Syst.* **2020**, *188*, 105002. [\[CrossRef\]](#)
26. Brajevic, I. A shuffle-based artificial bee colony algorithm for solving integer programming and minimax problems. *Mathematics* **2021**, *9*, 1211. [\[CrossRef\]](#)
27. Wang, H.Q.; Wei, J.H.; Wen, S.J.; Hou, Y.L. Research on parallel optimization of artificial bee colony algorithm. In Proceedings of the International Conference on Advanced Mechatronic system, Auckland, New Zealand, 9–12 July 2018.
28. Karaboga, D.; Akay, B. A comparative study of Artificial Bee Colony algorithm. *Appl. Math. Comput.* **2009**, *214*, 108–132.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.