Hindawi
Scientific Programming
Volume 2022, Article ID 7532076, 12 pages
https://doi.org/10.1155/2022/7532076

Research Article

Improvement of Wolf Pack Algorithm and Its Application to Logistics Distribution Problems

Wei Zheng

Henan Polytechnic Institute, Henan, Nanyang 473000, China

Correspondence should be addressed to Wei Zheng; zhengwei@hnpi.edu.cn

Received 1 August 2022; Revised 16 August 2022; Accepted 26 August 2022; Published 13 September 2022

Academic Editor: Lianhui Li

In logistics distribution systems, the constrained optimisation of the cargo dispensing problem has been the focus of research in related fields. At present, many scholars try to solve the problem by introducing swarm intelligence algorithms, including genetic algorithm, particle swarm algorithm, bee swarm algorithm, fish swarm algorithm, etc. Each swarm intelligence algorithm has different characteristics, but they all have certain advantages for the optimisation of complex problems. In recent years, the Wolf Pack algorithm, an emerging swarm intelligence algorithm, has shown good global convergence and computational robustness in solving complex high-dimensional functions. Therefore, this article chooses to use the Wolf Pack algorithm to solve a multi-vehicle and multi-goods dispensing problem model. First, the principle and process of the Wolf Pack algorithm are introduced, and two improvements are proposed for the way of location update and the way of step update. Then, a mathematical model of the multi-vehicle and multi-goods dispensing problem is developed. Next, the mathematical model is solved using the proposed improved Wolf Pack algorithm. The experimental results show that the proposed improved Wolf Pack algorithm effectively solves the cargo dispatching problem. In addition, the proposed improved Wolf Pack algorithm can effectively reduce the number of vehicles to be dispatched compared with other swarm intelligence algorithms.

1. Introduction

With the progress of science and technology, after reducing the cost of raw materials and improving labour productivity, modern logistics has become a "new source of profit" for enterprises and has received widespread attention from the logistics industry and even from the business community [1, 2]. Especially in the field of commodity circulation, different types of large-scale modern logistics enterprises have emerged. With the rapid development of the logistics industry, the study of logistics distribution has also attracted more widespread attention [3, 4]. How to reduce costs and at the same time improve efficiency, as well as obtain more economic and social benefits, has become a topic of research for experts and scholars.

In modern logistics, the biggest cost of logistics and distribution is the transport cost, which in turn has a lot to do with the vehicle. Therefore, when carrying out distribution operations, the load capacity and volume of the vehicle should be fully considered [5, 6]. By selecting the right transport vehicle, the optimum utilisation of the transport vehicle (100% utilisation) is achieved as far as possible, which is an effective way to reduce distribution costs. However, in practical situations, the space utilisation and load capacity of the vehicle often cannot be maximised at the same time due to the type of goods, packaging methods, etc. These situations can waste transport power and cause an increase in transport costs. In logistics and distribution systems, the issue of cargo dispensing is the most fundamental item [7, 8]. Reasonable dispensing can improve distribution efficiency on the one hand and reduce distribution costs on the other. At present, in the actual cargo dispensing business, it is usually done based on manual estimation methods. There is no uniform planning based on experience alone, which wastes both manpower and material resources [9, 10]. Therefore, the rational allocation of goods is conducive to improving distribution efficiency, reducing distribution costs, and achieving higher vehicle utilisation. Therefore, due to its extensive application background and important theoretical value, the study of cargo dispensing has become an important research content in the logistics and distribution industry [11, 12].

The key to the cargo dispensing problem is how to maximise the vehicle's capacity and volume, thereby reducing distribution costs and improving distribution efficiency. It is a complex discrete multi-constrained combinatorial optimisation problem, which belongs to the NP problem like the traveler problem and workshop scheduling problem [13, 14]. And for solving NP problems, heuristic algorithms or intelligent optimisation algorithms are the most commonly used techniques. Solving cargo dispensing problems often requires the use of heuristic algorithms or intelligent optimisation algorithms to approximate the optimisation solution.

There have been a number of studies on heuristic algorithms for cargo dispensing and loading problems. For example, Tran et al. [15] constructed a three-dimensional multilayer loading layout optimisation model with the maximisation of combined vehicle load and volume utilisation as the optimisation objective and designed a heuristic algorithm that can quickly develop a reasonable loading solution. Chua et al. [16] combined the two problems of vehicle dispensing optimization and transport path optimization into one and used the classical Dijkstra's algorithm and the improved C–W saving algorithm to solve the optimisation problem for the full-load transportation case and the optimisation problem for the non-full-load transportation case, respectively. Experimental results show that the solution is effective. Du et al. [17] developed an intelligent cargo dispensing model with multidimensional constraints and proposed a hybrid algorithm based on heuristic ideas and fuzzy principles. The test results showed the effectiveness of this intelligent dispensing model. Chao et al. [18] established a cargo dispensing model with the optimisation objective of maximising revenue and minimising expenditure and used a heuristic algorithm to solve it. Finally, the effectiveness of this algorithm was verified by example. However, heuristic algorithms tend to rely too much on personal experience. When the problem is large, heuristics can become "combinatorially explosive." As a result, heuristics are not very efficient when solving large-scale problems and lack global optimisation capabilities, giving only approximate or locally optimal solutions to the problem.

Swarm intelligence algorithms are all inspired by the evolution of organisms in nature, foraging, clustering, and information exchange. At present, the main swarm intelligence optimisation algorithms are simulated annealing algorithms, genetic algorithms, particle swarm algorithms, ant colony algorithms, immune algorithms, etc. These intelligent optimisation algorithms are all easy to implement and have good robustness. When solving complex optimisation problems, the advantages of swarm intelligence algorithms are more obvious. For solving large-scale, multi-constrained complex problems, swarm intelligent optimisation algorithms are more widely used. For example, Jamrus et al. [19] used a genetic algorithm to investigate the optimal placement of containers and developed a corresponding single-vehicle transport dispensing model, while Miao et al. [20] designed a hybrid genetic algorithm and applied it to a multi-species cargo dispensing model. The experimental results showed that the above model could fully utilise the load and volume of the loading tools in a balanced manner. Sicilia et al. [21] used an ant colony algorithm to solve a bi-objective bulk cargo loading model and verified that the proposed model and algorithm were feasible through experiments.

As an emerging swarm intelligence algorithm, the Wolf Pack algorithm exhibits good global convergence and computational robustness in the process of solving complex high-dimensional functions [22, 23]. During the hunting process, each wolf is classified into Alpha, Beta, or Omega wolves according to its different roles. Alpha wolves are always the strongest wolves in the pack and are responsible for directing the pack to capture prey without participating in the roaming, running, or siege process; Beta wolf is an elite unit of wolves, responsible for searching for prey; and Omega wolf is the attack force of wolves and is responsible for quickly closing in on the Alpha Wolf's direction when the Alpha Wolf initiates an attack command, in order to capture prey. Zhu et al. [24] proposed a Wolf Pack algorithm based on ant colony optimisation for solving the TSP problem. An ant colony algorithm was used to initialise the population in order to implement a heuristic crossover operator, while adaptive adjustment of the crossover probability and variation probability was employed. This algorithm achieves good optimisation results for smaller TSP problems. However, when the scale is larger, this algorithm needs further improvement. As an efficient intelligent optimisation algorithm, the Wolf Pack algorithm can provide a practical and effective solution to the cargo dispensing problem, thereby effectively reducing logistics costs (improving dispensing efficiency).

Therefore, this study aims to use the Wolf Pack algorithm to solve the cargo dispensing problem in logistics distribution, so that it can effectively improve the solution quality. Although the Wolf Pack algorithm has good global search performance and superiority-seeking ability, the number of iterations and convergence speed need to be further improved. Therefore, this article improves the traditional Wolf Pack algorithm. In addition, this article provides a brief introduction to the components and classification of the cargo dispensing problem in logistics distribution and establishes a multi-vehicle, multi-goods mathematical model. The improved Wolf Pack algorithm in this article is used to solve this model in order to reduce the number of vehicles used, which can effectively improve the solution quality. The final experimental results validate the performance and application value of the improved algorithm.

The main innovations and contributions of this article include:

(1) Through an in-depth study of the Wolf Pack algorithm, it was found that the location update method of Beta and Omega wolves has certain shortcomings, so the location update and step update of the Wolf Pack algorithm were improved respectively to achieve better optimisation search results.

(2) A mathematical model is described and developed for the multi-vehicle, multi-goods dispensing problem. The constraints of the model are deformed and transformed into a penalty function added to the fitness function. The model is solved using the proposed improved Wolf Pack algorithm to reduce the number of vehicles used, thus effectively improving the quality of the solution.

The rest of the article is organised as follows: In Section 2, the Wolf Pack algorithm is studied in detail, while Section 3 provides the improvements to the Wolf Pack algorithm. In Section 4, the multi-vehicle, multi-goods dispensing problem based on the improved Wolf Pack algorithm is studied in detail, while Section 5 provides experimental results and analysis. Finally, the article is concluded in Section 6.

2. Wolf Pack Algorithm

2.1. Principle of the Wolf Pack Algorithm. Each wolf is classified as an Alpha, Beta, or Omega wolf, depending on the role it plays in the hunting process [25, 26]. Alpha wolves are always the strongest wolves in the pack and are responsible for directing the pack to capture prey without participating in the roaming, running, or siege process; Beta wolf is an elite unit of wolves, responsible for searching for prey; and Omega wolf is the attack force of wolves and is responsible for the [27, 28]. The principle of the Wolf Pack algorithm is shown in Figure 1.

When a wolf perceives a greater concentration of prey than the current Alpha wolf, we consider that wolf to be more likely to capture the prey. That wolf will then replace the Alpha wolf and call the surrounding Omega wolves to approach the current location [29]. When the prey is caught, the earlier the wolf catches the prey, the more food it gets, which approach allows wolves capable of capturing prey to maintain sufficient stamina [30, 31]. Wolves with sufficient physical strength are more likely to catch prey in the later hunting process, thus ensuring the development of wolfs.

The total number of wolves is assumed to be N and the total number of variables is D. The state of an artificial wolf is represented as $X_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$, where$x_{\mathrm{id}}$ denotes the position of the ith artificial wolf in the dth dimension. The objective function is $Y = f(x)$, where Y denotes the concentration value of prey perceived by a wolf, that is the degree of adaptation.

2.1.1. Generation of Alpha Wolves. An Alpha wolf is an optimal value in the initial solution. An Alpha wolf is not fixed. During the iteration, the position of each wolf is updated continuously. If a better solution appears, the Alpha wolf is replaced by another wolf [32, 33].

2.1.2. The Wandering Process of Beta Wolves. Among N artificial wolves, the number of Beta wolves is $S_$num. The scale factor is denoted as a, the number of directions is h, and the wandering step is $\mathrm{step_a}$. The fitness value of the Beta wolves in the initial solution is Y_i. The Beta wolves then take a step forward in a direction ($p = 1,2, \ldots, h$) based on their current position. The position of the advancing Beta wolves in the dth dimension is updated.

$$x_{\mathrm{id}}^{p} = x_{\mathrm{id}} + \sin\left(2\pi \times \frac{p}{h}\right) \times \mathrm{step}_a^d. \tag{1}$$

After updating the Beta wolf's position, the adaptation value Y_i of the Beta wolf's current position is compared with the adaptation value Y_{lead} of the Alpha wolf. If $Y_i > Y_{\mathrm{lead}}$, this Beta wolf replaces the Alpha wolf, and the Omega wolf is called to run towards the current position; otherwise the wandering continues until the maximum number of times $T_{\max}$ is reached.

2.1.3. The Long-Range Raiding Process of Omega Wolves. Except Alpha wolf and Beta wolf, the remaining artificial wolves are Omega wolves. The number of Omega wolves is $_$num. When an Omega wolf receives a call from an Alpha wolf, it will run in the direction of the Alpha wolf. The step length of the long-range raid is step_b. The position of the Omega wolf in the dth dimension is updated at the $(k+1)$th iteration.

$$x_{\mathrm{id}}^{k+1} = x_{\mathrm{id}}^{k} + \mathrm{step}_b^d \times \frac{\left(g_d^k - x_{id}^k\right)}{\left|g_d^k - x_{\mathrm{id}}^k\right|}, \tag{2}$$

where g_d^k is the position in the d-th dimension of the Alpha wolf at the $(k+1)$th iteration.

When the distance between the Omega wolf and the Alpha wolf is less than d_{near}, it enters into a siege process of its prey.

$$d_{\mathrm{near}} = \frac{1}{D \times \omega} \times \sum_{d=1}^{D} \left|\max_d - \min_d\right|, \tag{3}$$

where ω is the decision factor and $[\max_d - \min_d]$ is the range of values for the dth dimensional variable.

2.1.4. The Siege Process of Prey. Because Alpha wolves are closest to the prey, the position of Alpha wolves is considered to be the position of the prey G_d^k. The siege step of the wolves in the prey siege process is step_c. Location updates are calculated as follows:

$$x_{\mathrm{id}}^{k+1} = x_{\mathrm{id}}^{k} + \lambda \times \mathrm{step}_c^d \times \left|G_d^k - x_{id}^k\right|, \tag{4}$$

where λ is a random number between [1].

There is a relationship between the step sizes of the three different stages.

$$\mathrm{step}_a^d = \frac{\mathrm{step}_b^d}{2} = step_c^d \times 2 = \frac{\left|\max_d - \min_d\right|}{S}. \tag{5}$$

During the siege process of a wolf pack, the position is updated if the adaptation of the current position is greater than the adaptation of the original position; otherwise, the position remains unchanged.

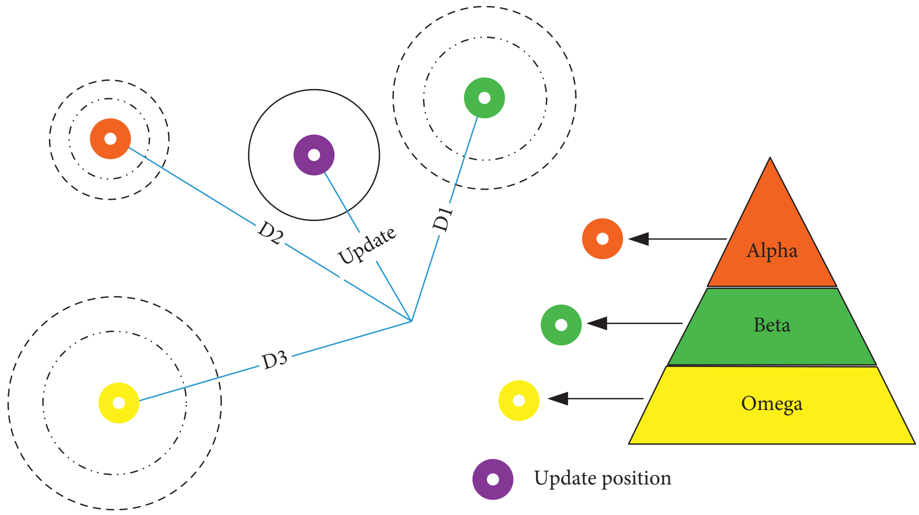

Figure 1: Principle of the Wolf Pack algorithm.

2.1.5. Renewal of Wolf Pack. In order to maintain the quality of the wolf population while preserving the diversity of the pack, the least adapted wolves are selected for culling and new artificial wolves are randomly generated [34–37]. The number of eliminated artificial wolves is the same as the number of newborn artificial wolves. The number of retirements needs to be determined by human experience.

2.2. Flow Chart of the Wolf Pack Algorithm. Figure 2 shows the flow chart of the Wolf Pack algorithm.

The detailed steps of the Wolf Pack algorithm are as follows:

Step 1: Initialise the number of artificial wolves N, the position X_i, the maximum number of generations to be selected $k_{\max}$, the scale factor a, the maximum number of wanderings $T_{\max}$, the distance determination factor ω, and the step size factor S.

Step 2: Follow equation (1) for position update. If the Beta wolf's fitness value Y_i is greater than the Alpha wolf's fitness value Y_{lead}, replace Alpha wolf and jump to Step 3; otherwise, the Beta wolf continues to wander until the maximum number of wanderings $T_{\max}$ is reached.

Step 3: Follow equation (2) for position update. If the fitness value of an Omega wolf Y_i is greater than the fitness value of an Alpha wolf Y_{lead}, replace the Alpha wolf; otherwise, the Omega wolf continues to long-range raid until the distance to the prey is less than the judged distance d_{near}.

Step 4: Begin the wolf pack siege process and carry out the position update according to equation (5), together with the Alpha wolf update.

Step 5: Update Wolf Pack.

Step 6: Determine whether the maximum number of iterations is reached or the optimisation accuracy is achieved. If yes, the output of the optimal solution is carried out; otherwise jump to Step 2.

3. Improvements to the Wolf Pack Algorithm

After an in-depth study of the Wolf Pack algorithm, it was found that position update methods have certain shortcomings for Beta wolf and Omega wolf. Therefore, in this article, the Wolf Pack algorithm is improved in order to achieve better merit-seeking results.

3.1. Location Update. The Beta wolf's position during the wandering phase [38] was updated as follows:

$$x_{\text{id}}^{p} = x_{\text{id}} + \sin\left(2\pi \times \frac{p}{h}\right) \times \text{step}_a^d, \tag{6}$$

where h is the number of directions in Beta wolf's wandering process.

In most cases, the value of h is 4. The calculation shows that there are only two values for updating the position of the Beta wolf.

$$\begin{aligned} x_{\text{id}}^{1} &= x_{\text{id}} + \text{step}_a^d, \\ x_{\text{id}}^{3} &= x_{\text{id}} - \text{step}_a^d, \\ x_{\text{id}}^{2} &= x_{\text{id}}^{4} = x_{\text{id}}. \end{aligned} \tag{7}$$

That is, when Beta wolves wander in each of the four directions, they are only able to obtain two values that are different from their original position. This increases the computational effort and weakens the ability of Beta wolves to wander [39].

Therefore, the updated formula for the wandering phase has been improved in this article.

$$x_{\text{id}}^{p} = x_{\text{id}} + \cos\left(\pi \times \frac{p}{h}\right) \times \text{step}_a^d. \tag{8}$$

When h takes the value of 4, the Beta wolf's position is updated with a value of 3.

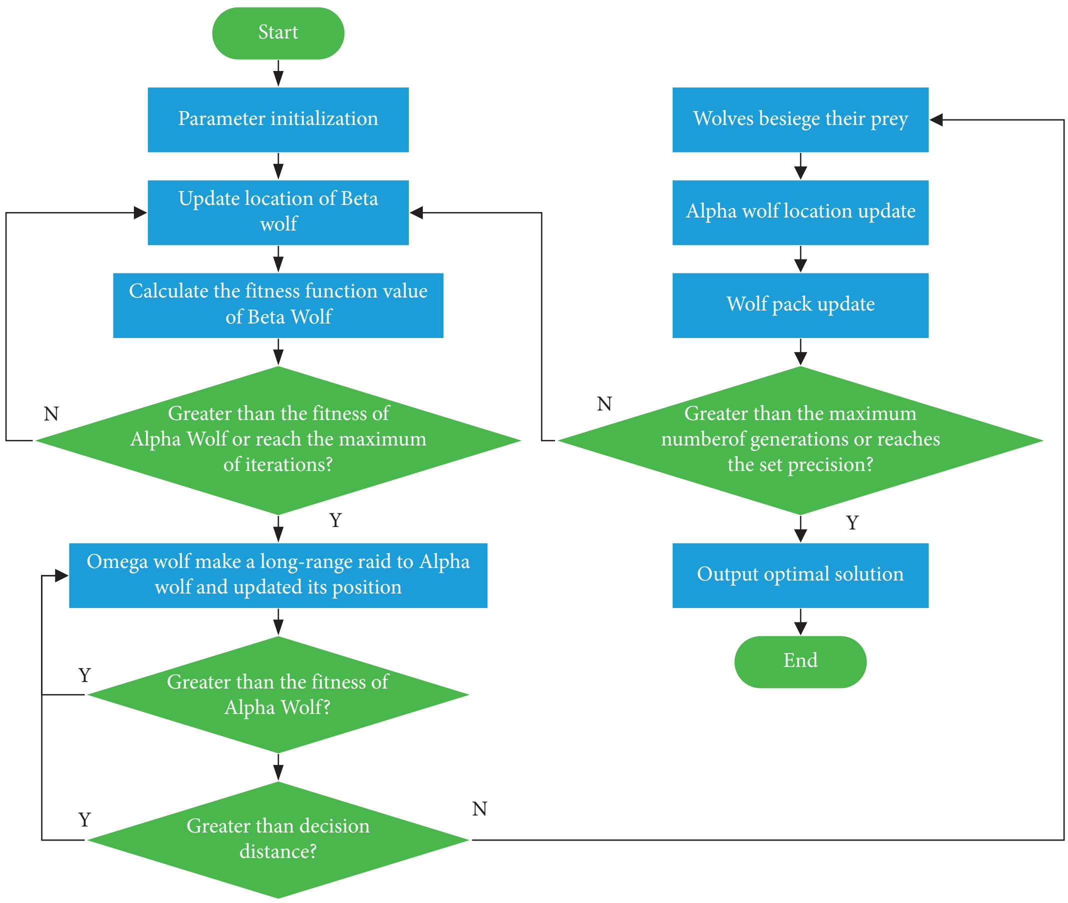

Figure 2: Flow of the Wolf Pack algorithm.

$$x_{id}^{1} = x_{id} + \frac{\sqrt{2}}{2}\text{step}_a^d,$$
$$x_{id}^{3} = x_{id} - \frac{\sqrt{2}}{2}\text{step}_a^d, \tag{9}$$
$$x_{id}^{4} = x_{id} - \text{step}_a^d.$$

The improvement ensures that when Beta wolves wander in more than one direction, at most one value can be the same as the original position value.

3.2. Step Size Update. The step size of the long-range raiding and the step size of the siege are always fixed. However, as the distance between the wolves and their prey decreases (increasing number of iterations), both step values should decrease adaptively.

Therefore, this article uses an adaptive approach to improve the update process of the step size.

$$x_{id}^{k+1} = x_{id}^{k} + \frac{(1-k)}{k_{\max}} \times \text{step}_b^d \times \frac{\left(g_d^k - x_{id}^k\right)}{\left|g_d^k - x_{id}^k\right|},$$
$$x_{id}^{k+1} = x_{id}^{k} + \lambda \times \frac{(1-k)}{k_{\max}} \times \text{step}_c^d \times \left|G_d^k - x_{id}^k\right|, \tag{10}$$

where k is the current number of iterations and $k_{\max}$ is the maximum number of iterations.

4. Multi-Vehicle, Multi-Goods Dispensing Problem Based on the Improved Wolf Pack Algorithm

4.1. Description of Cargo Dispensing Problems. As a fundamental part of logistics and distribution, the quality of the cargo dispensing has an impact not only on the efficiency of distribution, but also on the operational efficiency of the entire logistics centre. Therefore, a reasonable cargo dispensing can reduce logistics costs, on the one hand, and improve the efficiency of distribution on the other. The dispensing problem is a complex discrete multi-constrained combinatorial optimisation problem, which belongs to the same NP problem as the travel merchant problem (TSP) and the workshop scheduling problem. The main components of the cargo dispensing problem are goods, vehicles, constraints, objective functions, etc. [40]. The volume and weight of the cargo are the basis for the decision of vehicle allocation. When the volume or weight of the cargo exceeds the volume or capacity of the vehicle, multiple transport vehicles are required to distribute it. A general description of the cargo dispensing problem is shown in Figure 3.

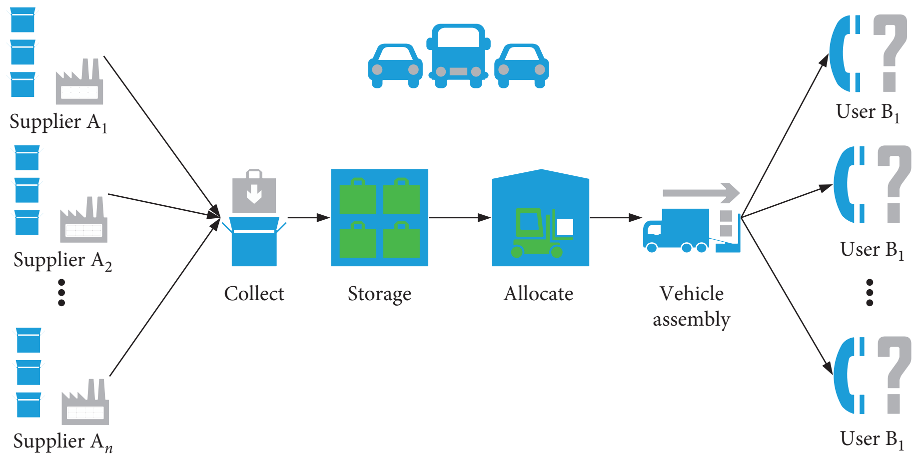

Figure 3: General description of the cargo dispensing problem.

4.2. Problem Description and Mathematical Model. Real-life cargo dispensing problems are very complex, and to facilitate modelling and solution, some cases are simplified: (1) the goods are all shipped to the same distribution centre; (2) the goods are intermixable and not incompatible; (3) there is no priority of goods.

Problem description: In a distribution centre, there are m vehicles of different types. Each vehicle has a maximum weight and a maximum volume of w_k and v_k $(k = 1,2, \ldots, m)$, respectively. There are n different goods to be dispensed, with weights and volumes of w_i and v_i $(i = 1,2, \ldots, n)$, respectively. Under the condition that the goods are relatively limited, the number of vehicles to be used is required to reach a minimum value.

The mathematical model for the multi-vehicle, multi-goods dispensing problem presented in this article is shown as follows:

$$\min f = \sum_{k=1}^{m} y_k, \sum_{i=1}^{n} x_{\mathrm{ki}} w_i \le y_k W_k, \sum_{i=1}^{n} x_{ki} v_i \le y_k V_k, \sum_{k=1}^{m} x_{ki} = 1, \tag{11}$$

$$y_k = \begin{cases} 1, \text{ The} k \text{thvehicleisselected}, \\ 0, \text{ The} k \text{thvehicleisnotselected}, \end{cases} \tag{12}$$

$$x_{\mathrm{ki}} = \begin{cases} 1, \text{ The} i \text{thshipmentisloadedintothe} k \text{thvehicle}, \\ 0, \text{ The} i \text{thshipmentisnotloadedintothe} k \text{thvehicle}. \end{cases} \tag{13}$$

The objective function is (11). It can be seen that the objective function is a minimum optimisation problem, that is, the number of vehicles used is the least. The constraint indicates that the total weight of the cargo on the kth *vehicle does* not exceed the capacity of the vehicle. (12) and (13) represent the constraints on the values taken by the variables y_k and x_{ki}.

4.3. Fitness Function. Fitness function is the optimisation index of wolf pack algorithm and the important basis of survival of the fittest in the siege process. In the mathematical model of multi-vehicle and multi-goods allocation, the objective function cannot be directly used as the fitness function due to more constraints. Therefore, this article uses the penalty function to construct the fitness function.

$$\begin{aligned} \min F &= f + c\sum_{k}^{m}\left(\left[\max\left(0, g_k\right)\right]^2 + \left[\max\left(0, h_k\right)\right]^2\right), \\ g_k &= \sum_{i=1}^{n} x_{\mathrm{ki}} w_i - y_k W_k, \\ h_k &= \sum_{i=1}^{n} x_{\mathrm{ki}} v_i - y_k V_k, \end{aligned} \tag{14}$$

where c is the penalty factor.

The optimisation process of the Wolf Pack algorithm is a maximum value problem, so a further transformation has to be performed to obtain the optimisation fitness function of the Wolf Pack algorithm.

$$\mathrm{maxFit}(F) = -F. \tag{15}$$

5. Experimental Results and Analysis

5.1. Experimental Environment and Setup. In order to verify the effectiveness of a multi-vehicle, multi-cargo dispensing model based on the improved Wolf Pack algorithm, simulation tests were carried out using MATLAB. The hardware and software environments associated with the experiments are shown in Table 1. In addition, all experiments involved in this study were carried out in the same hardware and software environment. The parameter settings for the improved Wolf Pack algorithm during the experiments are shown in Table 2.

5.2. Typical Test Function Verification. Five typical continuous complexity functions are selected to verify the effectiveness and feasibility of the proposed improved wolf pack algorithm and to compare it with the traditional wolf pack algorithm.

Table 1: Experiment-related software and hardware environment and parameter settings.

Specification	Parameters
Processor	Intel Core i7
Memory	8 GB
Hard disk	500 GB
Operating systems	Windows 10 Professional
Programming software	MATLAB R2012a

Table 2: Parameter settings for the improved Wolf Pack algorithm.

Parameters	Numerical values
Wolf population	50
Maximum number of iterations	20
Scale factor for Beta wolves	4
Maximum number of wanderings for Beta wolves	20
Number of directions for Beta wolves	4
Distance determination factor	500
Step size factor	1000
Updating the scale factor	6

$$
\begin{aligned}
F_1 &= x_1^2 + x_2^2, -5 \le x_i \le 5, i = 1, 2,\\
F_2 &= 100\left(x_1^2 - x_2\right)^2 + \left(1 - x_1\right)^2, -2.048 \le x_i \le 2.048, i = 1, 2,\\
F_3 &= \left[1 + \left(x_1 + x_2 + 1\right)^2\left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2\right)\right]\times\\
&\quad \left[30 + \left(2x_1 - 3x_2\right)^2\left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2\right)\right], -2 \le x_i \le 2, i = 1, 2,\\
F_4 &= \left(4 - 2.1x_1^2 + \frac{1}{3}x_1^4\right)x_1^2 + x_1x_2 + \left(-4 + 4x_2^2\right)x_2^2, -3 \le x_i \le 3, i = 1, 2\\
F_5 &= 10\cos\left(2\pi x_1\right) + 10\cos\left(2\pi x_2\right) - x_1^2 - x_2^2 - 20, -5.12 \le x_i \le 5.12, i = 1, 2.
\end{aligned}
\tag{16}
$$

In order to illustrate the convergence and optimisation seeking ability of the improved Wolf Pack algorithm in this article, a comparison was made with the genetic algorithm and the traditional Wolf Pack algorithm on the above five test functions. Figure 4 shows the optimisation process curves of the three algorithms on the test function F_1. Figure 5 shows the optimisation curves of the three algorithms on the test function F_2

It can be seen that the genetic algorithm can only converge to a local optimum. Both the traditional Wolf Pack algorithm and the improved Wolf Pack algorithm can converge to the global optimum, while the improved Wolf Pack algorithm converges faster, indicating that the improved Wolf Pack algorithm in this article can greatly improve the efficiency of the search for the optimum. In order to further illustrate the optimisation finding ability of the improved Wolf Pack algorithm in this article, 100 calculations were carried out for each of the five test functions. Then, four metrics, the best value, the worst value, the average value, and the average deviation value (the deviation of the average value from the theoretical optimum value), were used to evaluate the results. A comparison of the test function optimisation results is shown in Table 3.

The best, worst, and average values show that the solution quality of the improved Wolf Pack algorithm is significantly better than that of the genetic algorithm and the traditional Wolf Pack algorithm. The highest accuracy was obtained by the improved Wolf Pack algorithm. In terms of the average deviation value, the improved wolf pack algorithm has the smallest value, that is the smallest difference from the theoretical optimum.

The global search capability of the three algorithms was then judged by the number of convergences. A comparison of the convergence results on the five test functions is shown in Table 4. The maximum number of evolutionary generations was set to 1000. 100 tests were performed for each algorithm. The convergence accuracy was 10^{-3}. These algorithms were considered to have reached the convergence condition when the difference between the optimised value and the theoretical optimum was less than 10^{-3}.

It can be seen that the improved Wolf Pack algorithm converges 100 times on all five test functions, indicating that it is able to achieve global convergence with 100% probability, that is it is more stable. The genetic algorithm has the worst convergence, with only 4 out of 100 experimental tests, indicating that it is prone to fall into the local extreme value trap. Compared to the traditional wolf pack algorithm, the improved wolf pack algorithm takes 40% less time to reach convergence, indicating that it can converge to the global optimum more quickly. Overall, the number of convergence iterations and convergence time of the improved wolf pack algorithm are the lowest.

5.3. Experimental Cases. Suppose a distribution centre has 20 delivery vehicles with a load and volume of 6 t and 10 m^3, respectively. There are 42 different types of goods to be dispensed. The goods now need to be rationally dispensed

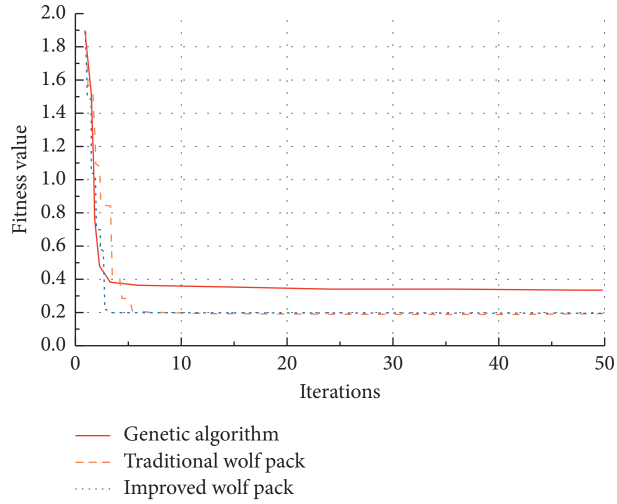

Figure 4: Optimisation curve for the F_1 function.

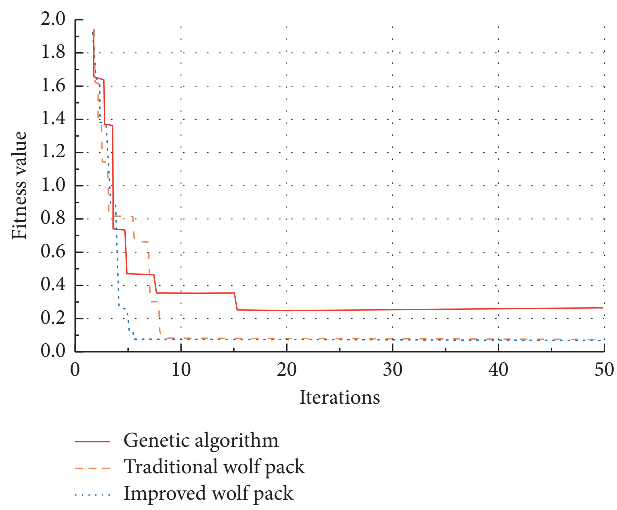

Figure 5: Optimisation curve for the F_2 function.

Table 3: Comparison of test function optimisation results.

Functions	Algorithms	Best value	Worst value	Average	Average deviation value
	Genetic algorithm	2.15E-05	1.0173	0.1302	0.1302
F_1	Traditional Wolf Pack	4.55E-11	1.17E-05	1.65E-07	1.65E-07
	Improved Wolf Pack	4.55E-11	4.55E-11	4.55E-11	4.55E-11
	Genetic algorithm	8.63E-04	2.8217	0.64209	0.64209
F_2	Traditional Wolf Pack	1.08E-10	5.76E-02	2.63E-03	2.63E-03
	Improved Wolf Pack	1.01E-10	1.94E-03	2.30E-04	2.30E-04
	Genetic algorithm	3.0094	96.9906	19.8777	16.8777
F_3	Traditional Wolf Pack	3	3.0002	3 + 7.79E-06	7.79E-06
	Improved Wolf Pack	3	3 + 5.63E-09	3 + 1.47E-09	1.47E-09
	Genetic algorithm	−1.03068005	−0.122028	−0.754928	0.2767
F_4	Traditional Wolf Pack	−1.03162779	−0.999872	−1.0276737	3.95E-03
	Improved Wolf Pack	−1.03162779	−1.03153801	−1.03161098	1.70E-05
	Genetic algorithm	−1.0174	−12.6704	−4.9979	4.9979
F_5	Traditional Wolf Pack	−9.46E-09	−1.2372	−0.1319	0.1319
	Improved Wolf Pack	−9.46E-09	−9.46E-09	−9.46E-09	9.46E-09

Table 4: Comparison of convergence results of test functions.

Functions	Algorithms	Maximum number of iterations	Minimum number of iterations	Average number of iterations	Average convergence time/s	Number of convergences
F_1	Genetic algorithm	1000	84	977	9.157	4
	Traditional Wolf Pack	539	11	46	0.539	100
	Improved Wolf Pack	60	7	26	0.304	100
F_2	Genetic algorithm	1000	2	990	9.433	11
	Traditional Wolf Pack	1000	2	442	5.387	61
	Improved Wolf Pack	947	12	219	2.681	100
F_3	Genetic algorithm	1000	1000	1000	9.582	0
	Traditional Wolf Pack	600	23	115	1.419	100
	Improved Wolf Pack	162	21	66	0.823	100
F_4	Genetic algorithm	1000	1	970	9.905	3
	Traditional Wolf Pack	1000	16	256	3.224	78
	Improved Wolf Pack	326	1	52	0.654	100
F_5	Genetic algorithm	1000	1000	1000	9.462	0
	Traditional Wolf Pack	1000	43	293	3.560	89
	Improved Wolf Pack	355	17	104	1.282	100

Table 5: Number, weight, and volume of goods.

No.	Weight/t	Volume/m^3	No.	Weight/t	Volume/m^3	No.	Weight/t	Volume/m^3
1	1.221	1.05	15	1.040	2.60	29	1.102	2.46
2	1.156	1.98	16	0.805	1.23	30	2.041	2.20
3	0.700	2.00	17	1.220	0.65	31	1.900	2.80
4	1.243	3.14	18	1.000	2.40	32	2.400	3.20
3	1.600	2.86	19	1.782	0.87	33	1.029	3.00
6	1.612	2.17	20	1.100	1.54	34	3.000	1.20
7	2.300	4.80	21	1.030	5.60	35	1.840	1.20
8	1.930	5.20	22	0.730	4.40	36	1.796	3.89
9	1.850	2.30	23	1.030	1.80	37	2.650	1.01
10	1.900	3.80	24	2.430	3.80	38	1.975	123
11	1.120	2.00	25	1.520	4.00	39	0.800	1.00
12	1.431	4.02	26	1.890	5.46	40	1.100	3.20
13	0.600	2.78	27	1.320	3.54	41	1.200	0.80
14	0.306	3.22	28	1.150	1.60	42	2.000	1.10

and the minimum number of vehicles to be used are required. The numbers, weights, and volumes of all goods are shown in Table 5.

The experiment was carried out for 40 calculations. Both the conventional Wolf Pack algorithm and the modified Wolf Pack algorithm yielded an optimal solution of 12 after 40 calculations, which indicates that a minimum of 12 vehicles are required. An optimal dispensing solution is shown in Table 6.

When using the improved particle swarm algorithm (IPSO) to solve the same multi-vehicle, multi-goods dispensing problem, the best value is 13 in 35 out of 40 operations and the worst value is 16. When using the improved genetic algorithm (IGA) to solve the same vehicle dispensing problem, the best value is 13 in 36 out of 40 operations and the worst value is 17. The results of the different algorithms are compared in Table 7 and Figure 6. It can be seen that the improved Wolf Pack algorithm has better optimisation

Table 6: The solution of dispensing.

Vehicle no.	Goods no.	Total weight/t	Total volume/m^3	Load utilisation (%)	Volume utilisation (%)
1	22, 23, 30, 35	5.641	9.6	94.02	96
2	1, 4, 7, 41	5.964	9.79	99.4	97.9
5	9, 16, 18, 27	4.975	9.47	82.92	94.7
7	20, 36, 40, 42	5.996	9.73	99.93	97.3
8	6, 14, 28, 33	4.097	9.99	68.28	99.9
9	2, 10, 19, 29	5.94	9.11	99	91.1
12	5, 17, 26	4.71	8.97	78.5	89.7
13	11, 24, 25	5.07	9.8	84.5	98
14	3, 12, 34	5.131	7.22	85.52	72.2
16	8, 32	4.33	8.4	72.17	84
19	21, 37, 38	5.655	7.84	94.25	78.4
20	13, 15, 31, 39	4.34	9.18	72.33	91.8

Table 7: Optimisation results.

Algorithms	Best value	Worst value	Probability of best value (%)
IPSO [41]	13	16	87.5
IGA [42]	13	17	90
Wolf Pack algorithm	13	15	100
Improved Wolf Pack algorithm	12	12	100

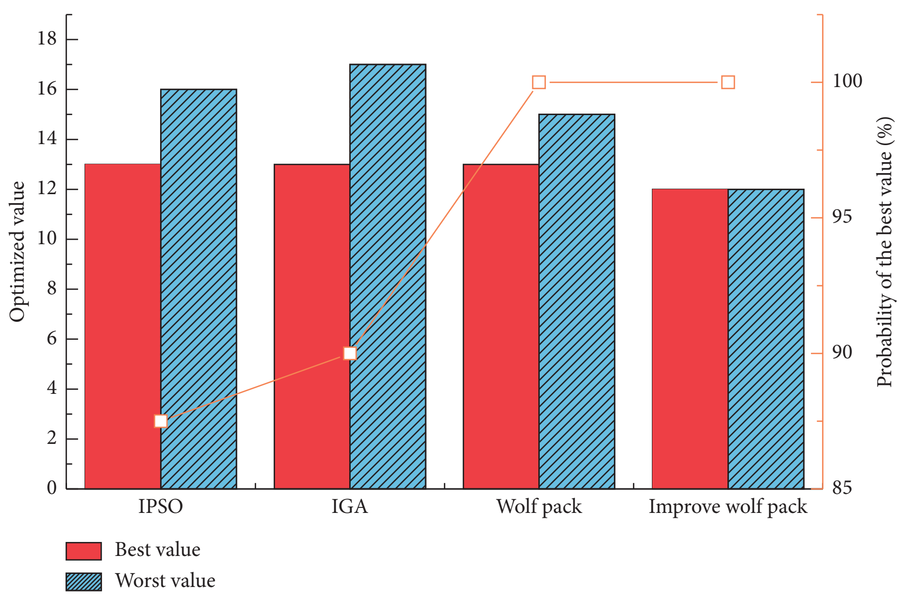

Figure 6: Comparison results of the optimising performance.

capability and better stability in solving multi-vehicle and multi-goods dispensing problems.

6. Conclusion

In this article, a mathematical model of a multi-vehicle and multi-goods dispensing problem is established and constraints are transformed into penalty functions. Simulation results of five typical test functions show that the proposed improved wolf pack algorithm achieves better performance in terms of accuracy and convergence time. Compared with the traditional Wolf Pack algorithm, the time required for convergence of the improved Wolf Pack algorithm is reduced by 40%. The simulation results of the logistics case show that the improved Wolf Pack algorithm has stronger optimising performance and better stability in solving the multi-vehicle and multi-goods dispensing problem, thus effectively reducing the number of vehicles. As the number of vehicles is often limited in actual logistics distribution, subsequent research will consider more realistic constraints to further improve the dispensing model.

Data Availability

The data that support the findings of this study are available from the author upon request.

Conflicts of Interest

The author declares that there are no conflicts of interest.

References

[1] S. Winkelhaus and E. H. Grosse, “Logistics 4.0: a systematic review towards a new logistics system,” *International Journal of Production Research*, vol. 58, no. 1, pp. 18–43, 2020.

[2] S. H. Chung, “Applications of smart technologies in logistics and transport: a review,” *Transportation Research Part E: Logistics and Transportation Review*, vol. 153, Article ID 102455, 2021.

[3] F. Sgarbossa, E. H. Grosse, W. P. Neumann, D. Battini, and C. H. Glock, “Human factors in production and logistics systems of the future,” *Annual Reviews in Control*, vol. 49, pp. 295–305, 2020.

[4] V. Yavas and Y. D. O. Ozen, “Logistics centers in the new industrial era: a proposed framework for logistics center 4.0,” *Transportation Research Part E: Logistics and Transportation Review*, vol. 135, Article ID 101864, 2020.

[5] T. M. Choi, “Risk analysis in logistics systems: a research agenda during and after the COVID-19 pandemic,” *Transportation Research Part E: Logistics and Transportation Review*, vol. 145, Article ID 102190, 2021.

[6] C. S. Tang and L. P. Veelenturf, “The strategic role of logistics in the industry 4.0 era,” *Transportation Research Part E: Logistics and Transportation Review*, vol. 129, pp. 1–11, 2019.

[7] L. Koh, A. Dolgui, and J. Sarkis, “Blockchain in transport and logistics – paradigms and transitions,” *International Journal of Production Research*, vol. 58, no. 7, pp. 2054–2062, 2020.

[8] T. S. Clair and J. Guzman, “Contribution volatility and public pension reform,” *Journal of Pension Economics and Finance*, vol. 17, no. 4, pp. 513–533, 2018.

[9] X. Qin, Z. Liu, and L. Tian, “The strategic analysis of logistics service sharing in an e-commerce platform,” *Omega*, vol. 92, Article ID 102153, 2020.

[10] J. M. Moshref and M. Winkenbach, “Applications and Research avenues for drone-based models in logistics: a classification and review,” *Expert Systems with Applications*, vol. 177, Article ID 114854, 2021.

[11] R. Beysenbaev and Y. Dus, “Proposals for improving the logistics performance index,” *The Asian Journal of Shipping and Logistics*, vol. 36, no. 1, pp. 34–42, 2020.

[12] M. Pournader, Y. Shi, S. Seuring, and S. L. Koh, “Blockchain applications in supply chains, transport and logistics: a systematic review of the literature,” *International Journal of Production Research*, vol. 58, no. 7, pp. 2063–2081, 2020.

[13] I. J. Orji, S. Kusi-Sarpong, S. Huang, and D. Vazquez-Brust, “Evaluating the factors that influence blockchain adoption in the freight logistics industry,” *Transportation Research Part E: Logistics and Transportation Review*, vol. 141, Article ID 102025, 2020.

[14] I. M. Ar, I. Erol, I. Peker, A. I. Ozdemir, T. D. Medeni, and I. T. Medeni, “Evaluating the feasibility of blockchain in logistics operations: a decision framework,” *Expert Systems with Applications*, vol. 158, Article ID 113543, 2020.

[15] H. Tran, N. Krommenacker, and P. Charpentier, “The Internet of Things for logistics: perspectives, application review, and challenges,” *IETE Technical Review*, vol. 39, no. 1, pp. 93–121, 2022.

[16] C. Chua, M. Danyluk, D. Cowen, and L. Khalili, “Introduction: turbulent circulation: building a critical engagement with logistics,” *Environment and Planning D: Society and Space*, vol. 36, no. 4, pp. 617–629, 2018.

[17] Y. Du, F. Chen, X. Fan, L. Zhang, and H. Liang, “Research on cargo-loading optimization based on genetic and fuzzy integration,” *Journal of Intelligent and Fuzzy Systems*, vol. 40, no. 4, pp. 8493–8500, 2021.

[18] C. Chao, H. Mei, and H. Yanhui, “Study of railway freight vehicle body’s dynamic model based on goods loading technical standards,” *Procedia Engineering*, vol. 29, pp. 3572–3577, 2012.

[19] T. Jamrus and C. F. Chien, “Extended priority-based hybrid genetic algorithm for the less-than-container loading problem,” *Computers & Industrial Engineering*, vol. 96, pp. 227–236, 2016.

[20] L. Miao, Q. Ruan, K. Woghiren, and Q. Ruo, “A hybrid genetic algorithm for the vehicle routing problem with three-dimensional loading constraints,” *RAIRO - Operations Research*, vol. 46, no. 1, pp. 63–82, 2012.

[21] J. A. Sicilia, B. Royo, E. Larrode, and A. Fraile, “A decision support system for a long-distance routing problem based on the ant colony optimization metaheuristic,” *Procedia-Social and Behavioral Sciences*, vol. 111, pp. 1035–1044, 2014.

[22] L. Xu, B. Wang, X. Du, and Y. Hong, “Prediction method of mine gas emission based on complex neural work optimized by Wolf pack algorithm,” *Systems Science & Control Engineering*, vol. 6, no. 3, pp. 85–91, 2018.

[23] Y. Chen, Z. Jia, X. Ai, D. Yang, and J. Yu, “A modified two-part wolf pack search algorithm for the multiple traveling salesmen problem,” *Applied Soft Computing*, vol. 61, pp. 714–725, 2017.

[24] Y. Zhu, L. Zhu, Y. Long, and J. Luo, “The Wolf pack assignment rule based on ant colony algorithm and the path planning of scout ants in complex raster diagra,” *Journal of Physics: Conference Series*, vol. 1732, no. 1, Article ID 012084, 2021.

[25] F. Yang, “A hybrid recommendation algorithm-based intelligent business recommendation system,” *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 21, no. 6, pp. 1317–1322, 2018.

[26] Y. Chen, D. Yang, and J. Yu, “Multi-UAV task assignment with parameter and time-sensitive uncertainties using modified two-Part Wolf pack search algorithm,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 54, no. 6, pp. 2853–2872, 2018.

[27] X. Chen, F. Cheng, C. Liu, L. Cheng, and Y. Mao, “An improved Wolf pack algorithm for optimization problems: design and evaluation,” *PLoS One*, vol. 16, no. 8, Article ID e0254239, 2021.

[28] Y. Lu, Y. Ma, J. Wang, and L. Han, “Task assignment of UAV swarm based on wolf pack algorithm,” *Applied Sciences*, vol. 10, no. 23, p. 8335, 2020.

[29] L. Zhen, Y. Liu, W. Dongsheng, and Z. Wei, “Parameter estimation of software reliability model and prediction based on hybrid wolf pack algorithm and particle swarm optimization,” *IEEE Access*, vol. 8, pp. 29354–29369, 2020.

[30] S. Xu, L. Li, Z. Zhou, Y. Mao, and J. Huang, “A task allocation strategy of the UAV swarm based on multi-discrete wolf pack algorithm,” *Applied Sciences*, vol. 12, no. 3, p. 1331, 2022.

[31] S. Xian, T. Li, and Y. Cheng, “A novel fuzzy time series forecasting model based on the hybrid wolf pack algorithm and ordered weighted averaging aggregation operator,” *International Journal of Fuzzy Systems*, vol. 22, no. 6, pp. 1832–1850, 2020.

[32] L. Y. Cang and X. P. Dong, "Improved Wolf Pack algorithm for optimum design of truss structures," *Executive Manager*, vol. 6, no. 8, p. 1411, 2020.
[33] Z. Wang, S. Yu, L. Y. Chen, and Y. Li, "Robust design for the lower extremity exoskeleton under a stochastic terrain by mimicking wolf pack behaviors," *IEEE Access*, vol. 6, pp. 30714–30725, 2018.
[34] H. S. Wu, J. J. Xue, R. B. Xiao, and J. Q. Hu, "Uncertain bilevel knapsack problem based on an improved binary wolf pack algorithm," *Frontiers of Information Technology & Electronic Engineering*, vol. 21, no. 9, pp. 1356–1368, 2020.
[35] Y. J. Gao, F. M. Zhang, Y. Zhao, and C. Li, "A novel quantum-inspired binary wolf pack algorithm for difficult knapsack problem," *International Journal of Wireless and Mobile Computing*, vol. 16, no. 3, pp. 222–232, 2019.
[36] J. Hu, H. Wu, R. Zhan, and N. Li, "Hybrid integer-coded Wolf Pack Algorithm for multiple-type flatcars loading problem," *Journal of Rail Transport Planning & Management*, vol. 16, Article ID 100201, 2020.
[37] M. Papin, M. Aznar, E. Germain, F. Guerold, and J. Pichenot, "Using acoustic indices to estimate wolf pack size," *Ecological Indicators*, vol. 103, pp. 202–211, 2019.
[38] X. Feng, K. Hu, and X. Lou, "Infrared and visible image fusion based on the total variational model and adaptive wolf pack algorithm," *IEEE Access*, vol. 8, pp. 2348–2361, 2020.
[39] R. Menassel, B. Nini, and T. Mekhaznia, "An improved fractal image compression using wolf pack algorithm," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 30, no. 3, pp. 429–439, 2018.
[40] R. Pinto, A. Lagorio, and R. Golini, "The location and sizing of urban freight loading/unloading lay-by areas," *International Journal of Production Research*, vol. 57, no. 1, pp. 83–99, 2019.
[41] H. Saleh, H. Nashaat, W. Saber, and H. M. Harb, "IPSO task scheduling algorithm for large scale data in cloud computing environment," *IEEE Access*, vol. 7, pp. 5412–5420, 2019.
[42] P. Shahrokh and F. Safi-Esfahani, "QoS-based web service composition applying an improved genetic algorithm (IGA) method," *International Journal of Enterprise Information Systems*, vol. 12, no. 3, pp. 60–77, 2016.