Applied Soft Computing Journal 95 (2020) 106561

Contents lists available at ScienceDirect

Applied Soft Computing Journal

journal homepage: www.elsevier.com/locate/asoc

Metaheuristics for solving the vehicle routing problem with the time windows and energy consumption in cold chain logistics

Mei-xian Song [a], Jun-qing Li [a,b,*], Yun-qi Han [b], Yu-yan Han [a], Li-li Liu [a], Qun Sun [a]

[a] School of Computer, Liaocheng University, Liaocheng 252059, China
[b] School of Information Science and Engineering, Shandong Normal University, Jinan 250014, China

ARTICLE INFO

Article history:
Received 21 May 2019
Received in revised form 20 March 2020
Accepted 15 July 2020
Available online 18 July 2020

Keywords:
Vehicle routing problem
Time windows
Energy consumption
Artificial fish swarm algorithm
Cold chain logistic

ABSTRACT

In this study, we consider a canonical vehicle routing problem (VRP) in the cold chain logistic system, where three special constraints are included, i.e., the dispatching time windows for each customer, different types of vehicles, and different energy consumptions and capacities for each vehicle. The objective is to minimize the total cost including the fixed cost and the energy consumptions. An improved artificial fish swarm (IAFS) algorithm is proposed, where a special encoding approach is designed to consider the problem feature with different type of vehicles. Then, improved preying and following heuristics are developed to perform the exploitation and exploration tasks. A novel customer satisfaction heuristic is embedded in the proposed algorithm, which makes the problem close to the reality. To further improve the performance of the algorithm, a right-shifting heuristic is designed to increase the customer satisfaction without increasing the energy consumption. An initialization heuristic based on the canonical Put Forward Insertion Heuristics (PFIH) is proposed to generate initial solutions with better performance. Finally, a set of realistic instances is generated to test the performance of the proposed algorithm, and after detailed experimental comparisons, the competitive performance of the proposed algorithm is verified.

1. Introduction

Since the vehicle routing problem (VRP) was first introduced by Dantzig and Ramser [1], it has attracted an increasing amount of attention. Solomon [2] conducted one of the earlier studies for solving the vehicle routing problem with time windows (VRPTW) and verified this problem was NP-hard. Tan et al. [3] solved the VRPTW by using a hybrid algorithm that combined the genetic (GA) algorithm and mountain climbing algorithm. Liu et al. [4] proposed a mixed integer programming model and an effective adaptive large neighborhood search heuristic for the VRPTW and synchronization services. Xia et al. [5] utilized the tabu search (TS) algorithm for solving the open vehicle routing problems with soft time windows and customer satisfaction. More recently, Naccache et al. [6] designed branch heuristics to develop a hybrid adaptive large neighborhood search to solve the multipick-up and delivery problem with time windows.

The heterogeneous vehicle routing problem with time windows (HVRPTW) is a new variation of the canonical VRP. Researchers tried to adopt effective strategies to solve the heterogeneous vehicle routing problem with time windows [7–9]. Koç and Bektaş et al. [10] developed a hybrid evolutionary algorithm (HEA) to solve this problem. Dondo et al. [11] and Adelzadeh et al. [12] focused on the heterogeneous fleet vehicle routing problem with time windows. At the same time, green issues have received increasing attention in the context of the HVRPTW [13,14]. Hiermann and Puchinger et al. [15] presented a hybrid heuristic which combines an adaptive large neighborhood search with an embedded local search. Li et al. [16] considered the energy issues and studied an emission-based heterogeneous fixed fleet routing problem (E-HFFVRP). There are extensions of the heterogeneous vehicle routing problem with time windows, for example the vehicle routing problem with mixed linehaul and backhaul customers (VRPMB) [17], heterogeneous fleet vehicle routing with overloads and time windows (HFVROTW) [18]. Jia and Yu et al. [19] combined a cooperative multiple task assignment problem with stochastic velocities and time windows for heterogeneous unmanned aerial vehicles and studied a two-stage stochastic programming model. Ghannadpour and Zarrabi [20] used the non-dominated sorting genetic algorithm II (NSGA II) to solve the multi-objective heterogeneous vehicle routing and scheduling problem, with energy minimizing. Wang et al. [21] developed a mathematical model, a ruin-recreate heuristic algorithm, and a threshold tabu search method to solve the heterogeneous multi-type fleet vehicle routing problem with time windows and an incompatible loading

* Corresponding author at: School of Information Science and Engineering, Shandong Normal University, Jinan 250014, China.
E-mail address: lijunqing@lcu-cs.com (J.-q. Li).

https://doi.org/10.1016/j.asoc.2020.106561
1568-4946/

constraint (HVRPTW-ILC), like refrigerated and non-refrigerated vehicles. Low-temperature transportation is the crucial link of cold chain logistics; however, it has high energy consumption. In ref [22] and [23], temperature remains low continuously during the processes of precooling, cold storage, and refrigerated transportation, more energy is needed and consumed leading to more carbon being released into the environment. Low-carbon logistics is for reducing energy consumption and greenhouse gas emissions, which is contradictory to the inherent properties of cold chain logistics. Recently, Li et al. [24] considered that cold chain logistics needs various costs and developed a cold chain logistics route optimization model including carbon emission cost. Their study aims to integrate the low carbon economy into the cold chain logistics. For the vehicle routing problem with simultaneous pickups and deliveries and time windows (VRP-SPDTW), Mingyong and Erbao investigated this type of problem by using an improved differential evolution algorithm (IDE) [25]. Marinakis et al. [26] tackled the VRPTW with a new variant of the particle swarm optimization (PSO) algorithm. Chen et al. [27] proposed a hybrid approach (HPSO) that combines PSO algorithm and simulated annealing (SA) for the solution of multi-compartment VRPTW arising from urban distribution. Vidal et al. [28] proposed a new hybrid genetic search metaheuristic to efficiently address several classes of multi-depot and periodic vehicle routing problems which used the evolutionary strategy with covariance matrix adaptation (CMA-ES) to converge towards good solutions used.

The origin of cold chain logistics can be traced back to the invention of the refrigerator in the 19th century. With the increasing demand for freezing and refrigerating food, cold chain logistics have been given an increasing amount of attention. Some studies combined the cold chain delivery problem with the VRPTW, where the penalty cost was the main factor to reflect the time sensitivity of the cold chain [29,30]. Tarantilies et al. [31] proposed an improved algorithm for this type of optimization problem. Zhang et al. [32] utilized the TS algorithm to optimize cold chain logistics distribution systems. Hsu et al. [33] considered the equipment energy consumption in the process of perishable food delivery as well as the impact of time window limits. Furthermore, Hsu and Chen [34] studied the optimization of vehicle size and distribution scheduling of multi-temperature food preparation according to the requirements of different temperature regions. However, there are fewer studies in the literature considering energy consumption and cold chain vehicles as well as the time window constraints for VRPs. Therefore, in this study, we consider the heterogeneous vehicle routing problem with time window constraints and energy consumption in a cold chain logistic system. Here, we consider not only the impact of delivery energy consumption but also many other constraints, such as the time windows and different types of vehicles, and solved the problem by using an improved artificial fish swarm (IAFS) algorithm. The main contributions are as follows: (1) a novel encoding approach containing two-dimensional vectors is developed; (2) to apply the canonical AFS algorithm for solving the discrete optimization problem, improved preying, following, and swarming behaviors are proposed; (3) a right shift heuristic is designed to improve the performance; and (4) an improved PFIH-based initialization method is utilized.

During recent years, many types of meta-heuristics have been developed for solving realistic optimization problems, in which artificial bee colony (ABC) and artificial fish swarm (AFS) algorithm are the two typical optimization algorithms inspired from the nature behaviors. Many researchers have applied meta-heuristics algorithm to solve many types of optimization problems, such as the parallel batching distributed flow shop problem [35], flexible job-shop scheduling problem [36], lot-streaming hybrid flowshop with variable sub-lots [37], dynamic economic dispatch problems with valve-point effects [38], realistic scheduling problems with transportation constraints [39], vehicle routing problems with time window and synchronized visit constraints [40], capacitated VRP [41], environmental VRP [42]. The AFS has been recently developed and applied for solving realistic optimization problems, such as optimal chiller loading problems [43], knapsack problems [44,45], mechanical parameter inversion problems [46], multiple flaws identification problems [47], the multiobjective fuzzy disassembly line balancing problem [48], and the attribute reduction algorithm [49]. Costa et al. [50] improved the AFS by using the hyperbolic augmented Lagrangian method. The other meta-heuristics for solving the VRP problem, including ant colony optimization [51–53], genetic local search algorithm [54], variable neighborhood search algorithm [55], knowledge-guided local search [56], tabu search algorithm [57,58], large multiple neighborhood search algorithm [59], iterated local search algorithm (ILS) [60,61], multi-swarm PSO method [62], large neighborhood search heuristic [63,64], and the genetic algorithm with exact dynamic programming [65]. From the literature about the VRP problems, we can see that most of literatures consider local search heuristics to solve it. However, the AFS algorithm has been verified to be an efficient algorithm with exploration and exploitation abilities, and there are less literature applying AFS algorithm for the VRP problems. Therefore, in this study, we aim to develop an improved AFS algorithm to solve the HVRPTW problems in the cold chain system.

The remainder of this paper is organized as follows. Section 2 briefly describes the problem. In Section 3, the canonical AFS algorithm is described. In Section 4, the detailed implementation of the improved AFS algorithm is reported. Then, Section 5 gives the experimental comparisons based on extended instances based on the SOLOMON instances. Finally, Section 6 summarizes the final conclusion.

2. Problem description

2.1. Description of the problem

In the canonical VRP, all vehicles are of the same type. That is, all vehicles have the same energy consumption, capacities, and devices. However, in a realistic logistic system, there are generally many types of vehicles with different capacities, energy consumption indexes and special devices. Meanwhile, the customer satisfaction normally affects the dispatch efficiency. For example, vehicles arrive within the time window of the customer will achieve a higher satisfaction. In this paper, the evaluation method of customer satisfaction is added to reflect the urgency of customer time window more accurately. Different from the general satisfaction calculation, not only through the calculation of the number of customers who do not serve in the time window, but also through an improved way to express the late or early time of the vehicle in the way of satisfaction. Any advance or delay will lead to a decline in satisfaction.

In addition, we adopt a rating method to determine customer satisfaction, which is different from other studies, where the number of customers outside their time windows [66] or the number of timeouts added to the target value cannot reflect the urgency of the customer point in the short window [5]. There are many advantages in this approach. First of all, the smaller the time window of the customer, the lower the tolerance to the customer. Secondly, according to the time interval of advance or delay, a proportion is calculated and added to the objective value. These advantages contribute to rendering our problems more practical.

Thus, in this study, we consider VRPs with the following features and consumption considerations:

(1) Vehicles with different energy consumption indexes are considered;
(2) VRPs with time constraints and the customer satisfaction are considered;
(3) Multiobjective VRPs are optimized, including minimization of the energy consumption and the negative comment values related to the customer satisfaction;
(4) Each route must start and end at the depot;
(5) Each customer has only one vehicle to serve;
(6) The sum of customer demands cannot exceed the maximum capacity of the type of vehicle assigned to the route;
(7) The vehicle type must be consistent with the type required by the customer;
(8) The service time of the vehicle should meet the customer's demand

2.2. Problem formulation

The HVRPTW is defined in a complete undirected network graph $G = (V, A)$, where V is a vertex set including one depot (vertex 0) and a set V' of n customers. $A = \{(i, j) | i, j \in V\}$ is an edge set, and represents the path between each customer. Each $i \in V$ is a customer with a nonnegative demand d_i, and service cost or time is $st_i (st_0 = 0)$. Let G_1, G_2, K_1, and K_2 represent the sets of the two types of goods and the two types of vehicles, respectively. Goods in G_1 (regular goods) and G_2 (refrigerated goods) must be delivered by vehicles in K_1 and K_2, respectively. Each vertex v_i $(i = 1, 2, \ldots, n)$ is associated with a non-negative demand set $\{q_{i,1}, q_{i,2}\}$ for two types of goods. We set an allowable time window $[E_{\text{ti}}, L_{\text{ti}}]$ and a preferred time window $[EE_{\text{ti}}, LL_{\text{ti}}]$. Customers cannot be visited after their allowable time window. In this study, we assume that the speed of the vehicles is constant, and the traveling energy consumption for vehicles is used to replace the traveling time cost. The problem is to minimize the total cost and to maximize the sum of all customers' satisfaction to increase service quality. The detailed list of parameters is as follows:

$k \in \{K_1 \cup K_2\}$	The types of vehicle
$g \in \{G_1 \cup G_2\}$	The types of good
d_i	Demand for customer i
t_i $(t_0 = 0)$	Time the vehicle arrives at customer i
st_i	Service time for customer i
c_{ij}	Vehicle travel time from i to j
q_k	Maximum loading capacity of type k vehicles
e_k	Energy consumption index of vehicle k
f_k	Fixed cost of type k vehicles
$[E_{ti}, L_{ti}]$	Preferred time window for customer i
$[EE_{ti}, LL_{ti}]$	Allowable time window for customer i
sv_i	Satisfaction for customer i

2.3. Problem example

Fig. 1 displays a solution example for an HVRPTW with 8 customers. There are two types of vehicles, where K_1 is regular vehicle type and K_2 is refrigerated vehicle type. The requirements of cargo compatibility and related parameters are listed in Tables 1 and 2, respectively.

3. Canonical AFS algorithm

The AFS algorithm is a population-based optimization algorithm that mimics the shoal of fishes foraging activities, which contains the following main steps: (1) a set of initial artificial fishes are generated to construct the initial population solutions; (2) each of the artificial fish will go through the following four behaviors, i.e., preying, swarming, following and random behavior, to get a better position with higher concentrations of food; and (3) a bulletin board solution is used to record the best solution found so far.

Table 1
Example customer parameters.

Customer	Vehicle type		$[E_{ti}, L_{ti}]$	$[EE_{ti}, LL_{ti}]$	st_i	d_i
	K_1	K_2				
1	0	1	[0,35]	[0,40]	17	2
2	1	0	[5,25]	[0,30]	15	4
3	0	1	[60,80]	[55,85]	17	1
4	0	1	[30,50]	[25,55]	17	4
5	1	0	[10,20]	[5,25]	15	1
6	1	0	[75,90]	[70,95]	15	2
7	1	0	[40,60]	[35,65]	15	3
8	1	0	[30,50]	[25,55]	15	2

Table 2
Example vehicle parameters.

Vehicle type	K_1	K_2
Maximum load q_k	15	8
Enable fixed costs f_k	30	40

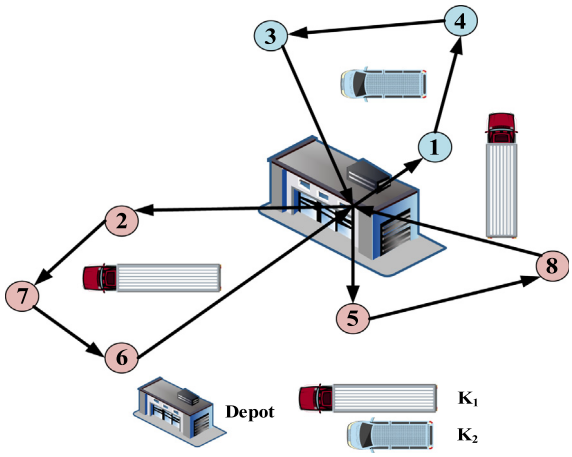


Fig. 1. Example VRP route chart.

3.1. Population initialization

Suppose m artificial fishes are randomly generated to represent the initialization of feasible solutions. Then, the state of the artificial fish swarm can be represented by a vector $X = (x^1, x^2, \ldots, x^m)$, where x^i is the current fish status. The food concentration of the current position of the artificial fish is expressed as $Y_i = f(x^i)$, and the distance between i and j is represented by $d_{ij} = \|x^i - x^j\|$. The step is the moving step length of the artificial fish. δ is the crowding factor in the current neighborhood, $\delta = \alpha\eta_{\max}, \alpha \in (0, 1]$, α is the extreme value close to the level and $\eta_{\max}$ is the maximum number of artificial fish expected to gather in this neighborhood. Visual is the visual range of artificial fish.

$$x_v = x + Visual \cdot Rand() \tag{1}$$

$$x^{inext} = x + \frac{x^v - x}{\|v\|} \cdot Step \cdot Rand() \tag{2}$$

where the function $Rand()$ is used to generate 0–1 random numbers.

3.2. Preying behavior

In the canonical AFS algorithm, the preying behavior is used to generate a neighboring solution in the visual scope of the selected artificial fish. Suppose x^i is a current state artificial fish and x^j is a

random artificial fish in the visual scope defined by the formula (3). If the randomly selected solution is better than the current one, then x^i will be replaced by x^j directly; otherwise, x^i will be replaced by another randomly selected solution in its visual range. The detailed implementation of the preying behavior is given as in (3).

$$x^{inext} = \begin{cases} x^i + \frac{x^j - x^i}{\|x^j - x^i\|} \cdot Step \cdot Rand\,() & if\ Y_j > Y_i \\ x^i + Visual \cdot Rand\,() & else \end{cases} \tag{3}$$

3.3. Swarming behavior

In the canonical AFS algorithm, the swarming behavior is used to generate a neighboring solution in the visual scope of the selected artificial fish. There are two situations in the swarming behavior: (1) to move towards the center of neighboring partners and (2) to avoid overcrowding. Suppose X_i is the current solution, n_f is the number of solutions in the current neighborhood, and x^c is the center position of current visual scope. Then, we use $Y_c/n_f > \delta Y_i$ to indicate that the center has more promising food and is not crowded. Under this condition, the solution will move further towards the center position. Then, the resulting neighboring solution will be used to replace the bulletin board solution and the current solution. The detailed formulation of the swarming behavior is given as follows.

$$x^{inext} = \begin{cases} x^i + \frac{x^c - x^i}{\|x^c - x^i\|} \cdot Step \cdot Rand\,() & if\ \left(\frac{Y_c}{n_f} > \delta Y_i\right) \\ prey\,() & else \end{cases} \tag{4}$$

3.4. Following behavior

The following behavior is a kind of operator to learn from the optimal or better neighboring solutions. Given a current solution x^i and the optimal partner X_{best} among all neighboring solutions in the current neighborhood, Y_{best} is the best fitness value in the current neighborhood. The condition $Y_{best}/n_f > \delta Y_i$ indicates that the neighboring center is not crowded with more promising food, and then the current solution will move further towards the center position, which is calculated as follows.

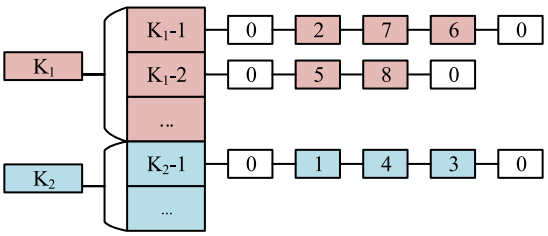
$$x^{inext} = \begin{cases} x^i + \frac{x^{best} - x^i}{\|x^{best} - x^i\|} \cdot Step \cdot Rand\,() & if\ \left(\frac{Y_{best}}{n_f} > \delta Y_i\right) \\ prey\,() & else \end{cases} \tag{5}$$

3.5. Random behavior

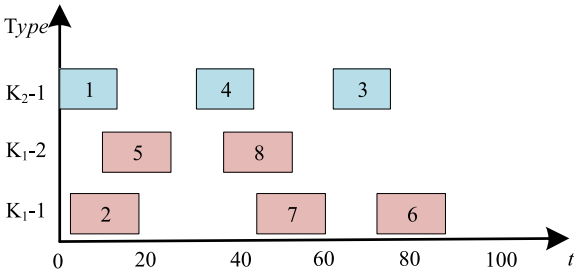
The random behavior is to select a state inside visual scope randomly and move towards the direction of this state. Therefore, the main step of the random behavior is to find a solution among the neighboring solutions of the current one and replace it with the selected neighboring solution if the condition is satisfied.

4. The proposed improved AFS algorithm

In this section, we propose an improved AFS algorithm (IAFS) to solve the HVRPTW problems in the cold chain logistics. It should be noted that, the canonical AFS algorithm is developed for solving the continuous optimization algorithms. In order to design a discrete version of the AFS algorithm, all the components in the proposed algorithm are discretized, including the encoding and decoding methods, the initialization approach and the improved behaviors for the canonical AFS algorithm.



(a) Coding representation



(b) Gantt chart for the example solution

Fig. 2. Coding and decoding method.

4.1. Encoding method

In the proposed algorithm, two-dimensional array encoding method is adopted to consider the problem features and the objectives. The first dimension of the two-dimensional array represents each vehicle, while the second dimension reports the assigned customers and processing sequence for each customer.

Given the example in Tables 1 and 2, there are eight customers, two types of vehicles and 25 vehicles for each type. The first type of vehicle is the regular vehicle without any special devices, and the second type is the cold chain or special vehicles. Fig. 2(a) gives an encoding example, where K_1 and K_2 represent the two types of vehicles, respectively. For each vehicle, we assign an array of serial numbers containing the assigned customers. Given eight customers, five customers {2, 7, 6, 5, 8} are assigned to the first type of vehicle, while the remaining three customers {1, 4, 3} are processed on the second type of vehicles. Further, the three customers {2, 7, 6} are assigned to the same vehicle belonging to $K_1 - 1$, and the customers {5, 8} are processed on $K_1 - 2$ with the assigned sequence.

The decoding of a solution is very simple, where all customers are assigned to the corresponding vehicle one by one. Meanwhile, the distance between two pair of customers, the earliest and latest time for each customer, the customer satisfied level, the service duration for each customer, the vehicle capacity, and the maximum trip time for each vehicle should be considered during the decoding process. For the coding representation in Fig. 2(a) and the customer requirements and vehicle information in Tables 1 and 2, the decoding Gantt chart is shown in Fig. 2(b), where each customer is serviced according to their processing sequences on the assigned vehicles. It should be noted that, in the decoding process, one of the important task is to compute the customer satisfaction values and add it to the problem fitness. The calculation of the customer satisfaction level is given in Section 4.2.

4.2. Customer satisfaction

During recent years, the customer satisfaction has gained more and more research focuses because the fact that the customer satisfaction often represents the service performances. The common

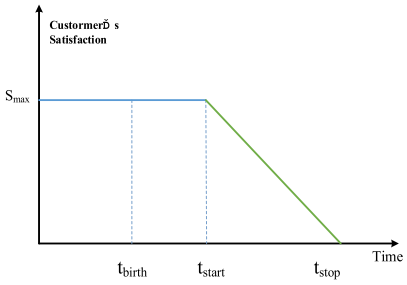


Fig. 3a. soft time window.

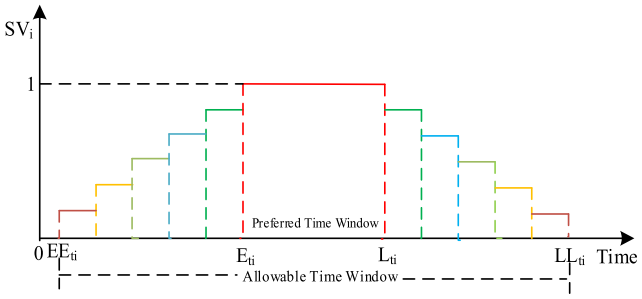


Fig. 3b. Service-level function of time windows.

way to compute the customer satisfaction values is to add it to the objective fitness directly. For example, in Ref [67], between the allowable time and prefer time, the customer satisfaction values are considered as a linear function. Gupta et al. [68] considered each customer has a desirable time window under which a service occurring over that interval would generate a high satisfaction (customer satisfaction is maximum) as opposed to a degraded (penalized) customer satisfaction if service takes place earlier or later than expected. Barkaoui et al. [69] proposed a dynamic vehicle routing problem with time windows and improved a satisfaction by several visits. In Ref [70], Guerriero et al. considered that customer satisfaction depends on the instant of time in which a drone reaches the customer point. More specifically, if the drone arrives at the events location before the time window, the customer satisfaction assumes the maximum value. It decreases linearly along with the increase of difference between the arrival time of drone and desired times and it becomes 0 when the drone arrives later the stop instant of event, as in Fig. 3a. But the initial and final events are not taken into account when evaluating the average satisfaction perceived by the customer and this assumption is unrealistic. One of the most important factors for customer satisfaction is to deliver products in the period of which customers incline to receive their demands. In the realistic applications, the customers generally cannot give an exact value for each time points. So, if a customer is served at its desired time, the grade of its satisfaction is 1; otherwise, the grade of satisfaction gradually decreases along with the increase of difference between the arrival time of vehicle and desired times. For example, in the dispatching logistics, the customer usually gives several values for the service satisfaction, which can be considered as a piecewise function. Fig. 3b gives an example of this type of piecewise function. The closer to the preferred time window the larger the satisfaction and in the other allowable time window, satisfaction decreases to 0 in stages. This is an accepted rule in real life.

There are three situations when the service vehicle arrives at the customer i at time t_i.

- When $E_{ti} \leq t_i < L_{ti}$, the customer satisfaction $sv_i = 1$;
- When $EE_{ti} \leq t_i < E_{ti}$, $sv_i = \left(1 - \frac{t_i - EE_{ti}}{E_{ti} - EE_{ti}}\right)$;
- When $L_{ti} \leq t_i < LL_{ti}$, $sv_i = \left(1 - \frac{LL_i - t_i}{LL_{ti} - L_{ti}}\right)$.

Assuming that there is a customer i, with the time window: $[E_{ti}, L_{ti}] = [50, 70]$, $[EE_{ti}, LL_{ti}] = [45, 75]$, $t_i = 48$, the tolerance time is 5. Because $EE_{ti} \leq t_i < E_{ti}$, the start time is earlier than the strict time window, but the customer can accept the vehicle for service and calculate bad review: $sv_i = \left(1 - \frac{48-45}{50-45}\right) = 0.4$.

4.3. Initialization method

Considering the initialization heuristic, Bräysy [71] gave a brief introduction about the classical route construction methods. There were three classical methods, i.e., Solomon [2], Potvin and Rousseau [72], and Ioannou et al. [73]. Solomon described several heuristics for the VRPTW, where the canonical put forward insertion heuristics (PFIH) was also one of the best-known route construction heuristics. The first heuristic begins with a solution where each customer is served by a unique vehicle, and then by merging two vehicles to decrease the number of vehicles, and therefore to save the cost and decrease the objective fitness. Then, the most efficient heuristic is to apply the insertion heuristic to construct a solution. By using this heuristics, a seed customer should firstly be chosen for each route, and then the unscheduled customers will be inserted into the current route until violation of the constraints. Two conditions are chosen for the seed customers, i.e., either the geographically farthest in relation to the depot or the lowest allowed starting time for service. To decide which unscheduled customer is to insert into the route, two typical conditions are considered, i.e., to select customers whose insertion costs minimize a measure of total route distance and time, or accounts for the urgency of servicing a customer. Potvin and Rousseau implemented a parallel version of Solomon's insertion heuristic. At first, a certain number of vehicles were initialized in a parallel way, that is, to dispatch n_r vehicles at once. The seed customer for each vehicle is also the same as in the Solomon, and the generalized regret measure was applied during the insertion of the unscheduled customer for each vehicle. Ioannou et al. embed the greedy look-ahead solution approach of Atkinson [74] into the customer selection and insertion criteria, which aimed to minimize the impact of insertion the newly selected customer. In consideration of the impact, both the impact on the customer under construction and the time window of the newly inserted customer are considered simultaneously.

In this study, we apply a simple and efficient initialization heuristics as follows: (1) firstly generate three solutions by using the Solomon PFIH heuristic, the parallel version insertion heuristic, and the greedy insertion heuristic; (2) after generating the three solutions, compute the fitness of these solutions, and sort each heuristic in an ascending order according to their fitness values. The result of sorting set is $HC = \{h_1, h_2, h_3\}$ with the fitness values $FC = \{f_1, f_2, f_3\}$; (3) calculate the applying times for each heuristics as follows:$AT_i = \frac{f_i}{\sum_{j=1}^{3} f_j}$; and (4) apply the heuristic h_i to generate AT_i solutions until the whole population has reached P_{size}.

4.4. Improved preying heuristic

In the canonical AFS algorithm, the preying behavior is used to generate the neighboring solution for each selected solution. However, the preying behavior in the AFS algorithm is just for the continuous optimization problem; to make it adaptable for solving the discrete optimization problem, we develop an improved preying behavior, which is given as follows.

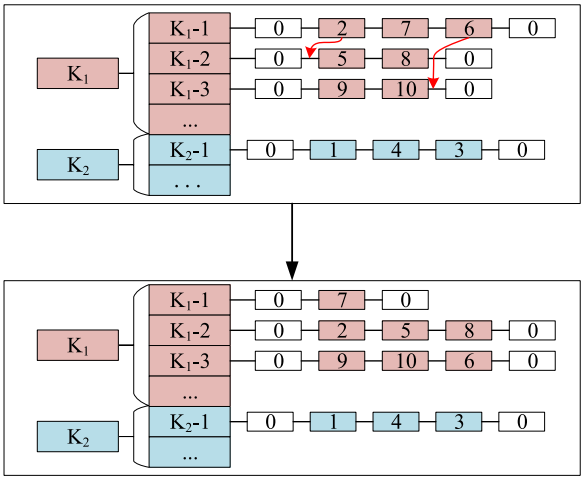


Fig. 4. Mutation process.

In the improved preying behavior, we adopt two strategies for the current artificial fish x^i:

(1) To generate a neighboring solution, one vehicle type is randomly selected, one vehicle is randomly selected for the selected vehicle type, and then a certain number of customers are randomly selected from the selected vehicle. After that, the selected customers are deleted from the selected vehicle, and then inserted into other vehicles belonging to the same vehicle type. Considering the time complexity of this type of local search method, it is obvious that the time complexity to insert a customer to the current solution is $O(n^2)$. Suppose the number of customers to be reinserted is $(1/r) \times n$, where r is the selected rate, and then the time complexity of this approach is $O(n^3)$.

(2) To generate a neighboring solution, similar to the first method, a random vehicle is randomly selected. Then, a random number of customers are randomly selected from the selected vehicle. After that, these selected customers are deleted from the selected vehicle, and then inserted into other vehicles belonging to the same vehicle type. Fig. 4 shows an example of this type of approach. In Fig. 4, two customers are randomly selected for the first type of vehicle, and an attempt is made to insert them into other vehicles of the first type.

It should be noted that, in the proposed algorithm, the two preying heuristics are selected in a random way.

4.5. Improved following heuristic

In the canonical AFS algorithm, the following heuristic is to make the current solution learn from the neighboring solutions. In the canonical following heuristics, if the visual scope of x^i is not crowded and the center has the best objective function value in the visual scope, x^i goes to the center. However, the canonical AFS algorithm is for continuous optimization problems. To make the AFS algorithm adaptive to solve the discrete optimization problem, we propose a discrete following behavior.

The main following behavior is implemented by two types of crossover operators, which are shown as follows:

(1) Reference crossover. The main idea of the reference crossover is to delete the customers occurred on the given routes of the other parent solution, and then to insert them into other best positions. The detailed steps are as follows: firstly, randomly select two parent solutions p_1 and p_2; then, a certain number of vehicles from p_2 are chosen, and the customers on these vehicles are stored into a set D; next, all the customers in D will be deleted from p_1, and these deleted customers will then be inserted into optimal positions by using the PFIH heuristic. Fig. 5(a) shows the procedure where p_1 and p_2 represent the parent solutions. According to this figure, some customers is selected from p_2 by a certain proportion, there are ten customers and we set $1/m(m = 2)$ customers are chosen, that is 5 customers. Then the customers on this route are removed from the routes of p_1 and reinserted into p_1 in the best insertion position. This procedure is continued until one feasible offspring solution is produced. The time complexity of the reference crossover is O(n).

(2) Simple crossover. The detailed steps of this type of crossover are as follows: firstly, randomly select two parent solutions p_1 and p_2; then the first type of vehicles of the offspring solution learn from one of the parent solution, and the other type of vehicles learn from the other one. Fig. 5(b) gives the crossover steps, where the two parent solutions are presented. Then, the special vehicle is taken from p_2 and the regular vehicle is taken form p_1, and thus the neighboring solution obtained. It should be noted that the crossover operator can generate a feasible solution without any repair process.

In the proposed algorithm, the two types of crossover operators are chosen in a random way, then the current artificial fish x^i moves towards the center (best point) of the fish swarm inside the visual scope with a probability p_c or to a random neighboring solution with a probability $1 - p_c$. The main advantages of the improved following behavior are as follows: (1) with a probability to learn from the best solution in the visual range, a solution can hold the ability to escape from the local best; and (2) to learn from a random neighboring solution, the proposed algorithm can enhance its exploration abilities. The time complexity of the reference crossover is O(n).

4.6. Right-shifting heuristic

In the considered problem, two types of time window are set for each customer, i.e., the allowable time window and the prefer time window. It should be noted that, in the prefer time window, the customer will be fully satisfied with the service, while in the allowable time window, the customer will be partially satisfied.

From the decoding procedure, we can see that each customer will be scheduled as early as possible, that is, to start their service times without considering the penalty to the customer satisfaction. For example, when the assigned vehicle arrives at the customer before the starting time of the allowable time window, and then the vehicle will wait for the opening of the allowable time window. As long as the starting time of the allowable time window begins, the service will start. This decoding method is the commonly used approach, which can generate a solution with minimum travel trip time. However, at that time, the customer satisfaction is lower because the service time is not in the prefer time window.

In the proposed algorithm, we develop a right-shifting heuristic, which can shift the customer service time considering the waiting time of the vehicle. The detailed steps of the right-shifting heuristic are given as follows.

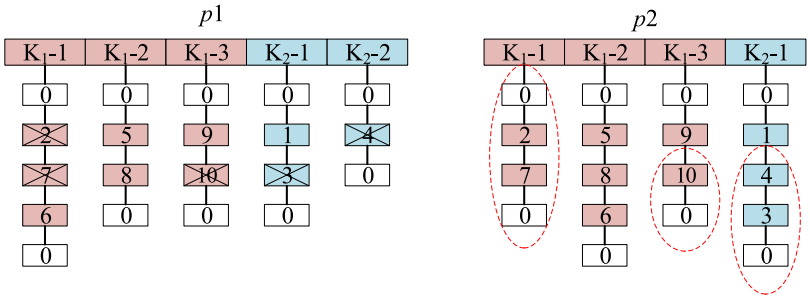
Step 1. For a given solution, check all the vehicles, and for each vehicle, perform the following steps.

Step 2. For each vehicle k, from right to left, find the first customer i that makes the vehicle waiting for the opening of its allowable time window.

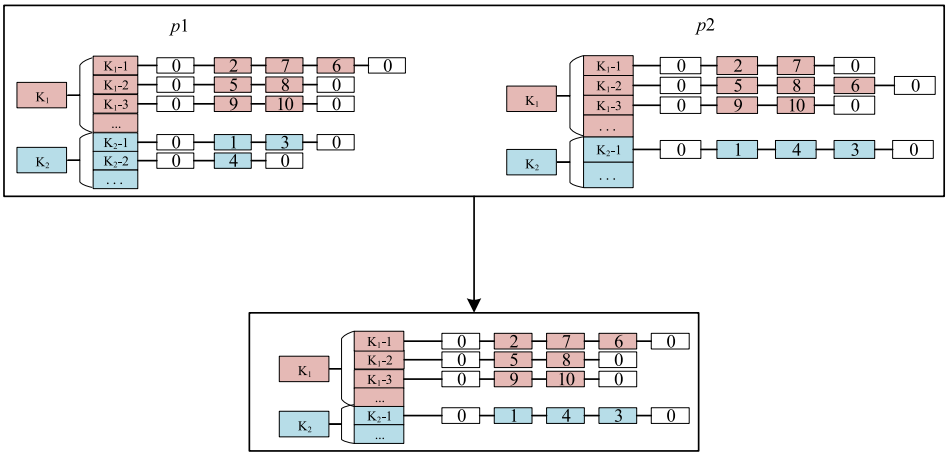
Step 3. For vehicle k, find customer j satisfying the following conditions: (1) customer i precedes j; (2) the starting service time of j is between the earliest allowable time and the earliest prefer time, that is, the starting time of j can be shifted right to increase the customer satisfaction.

Step 4. Repeat steps 2 and 3 for each vehicle.

For example, assume the current customers i with the time window [40, 50, 60, 70], where the prefer time window is [50,



(a) Reference crossover.



(b) Simple crossover

Fig. 5. Improved following heuristic.

60], and the allowable time window is [40,70]. The following customer is j with the time window [100, 110, 130, 140]. Assume that the service vehicle arrives at i at time point 45 and j at the time of 95. Therefore, the vehicle needs to wait for the allowable time window of j for 5 time units. Then, we can right shift the starting service time of i to right for 15 time units without affecting the starting time of j and following customers.

4.7. Framework of the proposed algorithm

The detailed steps of the proposed algorithm are given as follows.

Step 1. Set the system parameters.

Step 2. Initialize a population of solutions by using the initialization method discussed in Section 4.3, and evaluate each solution in the population and apply the right-shifting heuristic discussed in Section 4.6.

Step 3. While the stop criteria not satisfied, perform steps 4 to 5.

Step 4. For each solution in the current population, perform the local search procedures as follows:

Step 4.1 Perform the preying heuristic discussed in Section 4.4.

Step 4.2 Replace the best solution found so far if the newly-generated solution is better than the former.

Step 5. For each solution in the population, perform the following global search:

Step 5.1 Apply the following heuristic discussed in Section 4.5.

Step 5.2 Replace the best solution found so far if the newly-generated solution is better than the former.

5. Numerical analysis

5.1. Experimental instances

In the canonical SOLOMON example, all vehicles are of the same type, i.e., there is no difference among the vehicles. However, in the realistic logistic system, such as the cold food transportation system, some types of food should be transported by the vehicles with retriggers to keep them within a certain range of temperatures. To better consider the actual constraint, based on the canonical SOLOMON instances, a set of extended instances considering the cold vehicles are randomly generated. The extended instances also include 55 instances named "cc101" to "crc207". The main differences between the canonical SOLOMON instances and the extended instances are as follows: (1) all vehicles are categorized into two types, i.e., the regular vehicles and the special or refrigerated vehicles; (2) all customers are also classified into two types, i.e., the regular customers who require only the regular vehicles to transport regular products, and the special customers with requirements of the refrigerated vehicles; (3) all vehicles have different energy consumption rate and different capacities, and a vehicle with a larger capacity has a higher probability with a larger energy consumption rate.

5.2. Experimental parameter

The parameters of the experiment include the following. (1) R_a: the neighbor radius, which is the visual scope of a fish. For example, $R_a = 0.2$ represents that the 'visual scope' of a fish is 20% of the fish swarm. The larger R_a is, the larger the neighborhood range is. If $R_a = 1$, it means that the entire population is of a

Table 3
Comparisons of the different preying heuristics.

Ins	Best	Preying heuristics		dev	
		I	II	I	II
cc101	1108.61	1218.40	1108.61	9.90	**0.00**
cc102	903.88	903.88	912.03	**0.00**	0.90
cc103	797.28	845.39	797.28	6.03	**0.00**
cc104	612.61	612.61	642.69	**0.00**	4.91
cc105	950.81	1025.63	950.81	7.87	**0.00**
cc106	899.83	918.41	899.83	2.07	**0.00**
cc107	657.87	657.87	675.93	**0.00**	2.74
cc108	578.08	578.08	601.55	**0.00**	4.06
cc109	627.84	627.84	638.37	**0.00**	1.68
cc201	1434.43	1692.15	1434.43	17.97	**0.00**
cc202	856.38	929.49	856.38	8.54	**0.00**
cc203	484.93	550.40	484.93	13.50	**0.00**
cc204	421.59	437.55	421.59	3.79	**0.00**
cc205	896.92	980.03	896.92	9.27	**0.00**
cc206	712.02	712.02	727.03	**0.00**	2.11
cc207	489.35	511.55	489.35	4.54	**0.00**
cc208	608.90	608.90	623.66	**0.00**	2.42
cr101	1308.52	1409.74	1308.52	7.74	**0.00**
cr102	936.20	961.36	936.20	2.69	**0.00**
cr103	672.29	672.29	679.96	**0.00**	1.14
cr104	443.74	443.74	458.86	**0.00**	3.41
cr105	761.73	780.46	761.73	2.46	**0.00**
cr106	679.61	712.91	679.61	4.90	**0.00**
cr107	527.96	528.32	527.96	0.07	**0.00**
cr108	386.36	386.36	407.72	**0.00**	5.53
cr109	632.15	650.89	632.15	2.96	**0.00**
cr110	461.32	461.32	490.59	**0.00**	6.34
cr111	524.39	548.04	524.39	4.51	**0.00**
cr112	430.51	445.71	430.51	3.53	**0.00**
cr201	729.25	729.25	730.49	**0.00**	0.17
cr202	616.36	616.36	619.40	**0.00**	0.49
cr203	513.92	516.80	513.92	0.56	**0.00**
cr204	349.54	349.54	364.91	**0.00**	4.40
cr205	537.00	539.35	537.00	0.44	**0.00**
cr206	417.10	484.01	417.10	16.04	**0.00**
cr207	354.99	370.24	354.99	4.30	**0.00**
cr208	305.75	305.75	318.70	**0.00**	4.24
cr209	441.42	441.42	443.69	**0.00**	0.51
cr210	443.43	443.43	477.20	**0.00**	7.62
cr211	334.93	403.67	334.93	20.52	**0.00**
crc101	935.94	958.06	935.94	2.36	**0.00**
crc102	743.09	789.86	743.09	6.29	**0.00**
crc103	591.71	595.75	591.71	0.68	**0.00**
crc104	498.54	498.54	499.25	**0.00**	0.14
crc105	865.33	865.33	867.52	**0.00**	0.25
crc106	738.30	738.30	740.15	**0.00**	0.25
crc107	580.27	580.27	592.45	**0.00**	2.10
crc108	534.35	562.95	534.35	5.35	**0.00**
crc201	954.72	954.72	1014.66	**0.00**	6.28
crc202	753.63	759.05	753.63	0.72	**0.00**
crc203	523.31	523.31	534.56	**0.00**	2.15
crc204	429.09	434.25	429.09	1.20	**0.00**
crc205	786.05	821.42	786.05	4.50	**0.00**
crc206	648.87	687.31	648.87	5.92	**0.00**
crc207	523.59	523.59	551.04	**0.00**	5.24
Mean	651.43	675.49	658.67	3.24	**1.33**

Table 4
Comparisons of the different following heuristics.

Ins	Best	Following heuristics		dev	
		I	II	I	II
cc101	1108.61	1194.65	1108.61	7.76	**0.00**
cc102	892.41	892.41	912.03	**0.00**	2.20
cc103	797.28	825.26	797.28	3.51	**0.00**
cc104	642.69	655.49	642.69	1.99	**0.00**
cc105	950.81	1068.03	950.81	12.33	**0.00**
cc106	880.98	880.98	899.83	**0.00**	2.14
cc107	675.93	707.95	675.93	4.74	**0.00**
cc108	557.31	557.31	601.55	**0.00**	7.94
cc109	614.73	614.73	638.37	**0.00**	3.84
cc201	1434.43	1963.60	1434.43	36.89	**0.00**
cc202	839.80	839.80	856.38	**0.00**	1.97
cc203	484.93	493.85	484.93	1.84	**0.00**
cc204	421.59	424.22	421.59	0.62	**0.00**
cc205	896.92	904.98	896.92	0.90	**0.00**
cc206	680.71	680.71	787.03	**0.00**	15.62
cc207	489.35	532.17	489.35	8.75	**0.00**
cc208	618.57	618.57	623.66	**0.00**	0.82
cr101	1308.52	1474.47	1308.52	12.68	**0.00**
cr102	936.20	1073.21	936.20	14.63	**0.00**
cr103	679.96	720.39	679.96	5.95	**0.00**
cr104	458.86	476.33	458.86	3.81	**0.00**
cr105	761.73	796.87	761.73	4.61	**0.00**
cr106	679.61	760.58	679.61	11.91	**0.00**
cr107	527.96	546.45	527.96	3.50	**0.00**
cr108	399.62	399.62	407.72	**0.00**	2.03
cr109	632.15	679.44	632.15	7.48	**0.00**
cr110	490.59	506.18	490.59	3.18	**0.00**
cr111	524.39	599.50	524.39	14.32	**0.00**
cr112	430.51	479.47	430.51	11.37	**0.00**
cr201	800.43	800.43	851.49	**0.00**	6.38
cr202	619.40	665.09	619.40	7.38	**0.00**
cr203	506.70	506.70	513.92	**0.00**	1.43
cr204	359.79	359.79	364.91	**0.00**	1.42
cr205	525.75	525.75	537.00	**0.00**	2.14
cr206	417.10	453.91	417.10	8.83	**0.00**
cr207	354.99	385.41	354.99	8.57	**0.00**
cr208	316.39	316.39	318.70	**0.00**	0.73
cr209	448.00	448.00	483.69	**0.00**	7.97
cr210	477.20	510.93	477.20	7.07	**0.00**
cr211	334.93	379.28	334.93	13.24	**0.00**
crc101	935.94	1002.11	935.94	7.07	**0.00**
crc102	743.09	775.66	743.09	4.38	**0.00**
crc103	591.71	640.84	591.71	8.30	**0.00**
crc104	499.25	510.67	499.25	2.29	**0.00**
crc105	867.52	883.13	867.52	1.80	**0.00**
crc106	740.15	773.62	740.15	4.52	**0.00**
crc107	592.45	597.70	592.45	0.89	**0.00**
crc108	534.35	543.00	534.35	1.62	**0.00**
crc201	979.74	979.74	1014.66	**0.00**	3.56
crc202	658.16	658.16	753.63	**0.00**	14.51
crc203	534.56	562.40	534.56	5.21	**0.00**
crc204	419.61	419.61	429.09	**0.00**	2.26
crc205	786.05	1000.38	786.05	27.27	**0.00**
crc206	567.19	567.19	648.87	**0.00**	14.40
crc207	538.22	538.22	551.04	**0.00**	2.38
Mean	651.86	691.24	662.61	5.02	**1.72**

single neighbor scope, which is similar to the GA algorithm. (2)C_r: the crowding parameter which tells whether the neighboring space is crowded. (3) P_c: the following probability. Based on the detailed experiments and the parameter setting values in the published literature, the parameter levels for each parameter are all set to {0.2, 0.4, 0.6, 0.9}.

The DOE (Design of Experiment) Taguchi method is used, in which an orthogonal array, namely, L_{16}, is used. For each parameter combination, the proposed algorithm is independently run 30 times, and the average fitness value that is obtained via the proposed algorithm is collected as the Response Variable (RV). Fig. 6 reports the factor level and intersections. According to the results, the proposed algorithm yields better performance when the three parameters are set as follows: R_a is set to 0.4, C_r is set to 0.6, and P_c is set to 0.2.

5.3. Efficiency of the preying behavior

To verify the effectiveness of the proposed preying behavior, we compare the following two heuristics, i.e., the preying heuristic I and preying heuristic II. Heuristic I is when the current artificial fish mutate, one customer in one vehicle is randomly selected and inserted into other vehicles of the same type. Heuristic II is when a vehicle is randomly selected and several of its customers are inserted into other vehicles of the same type.

Table 3 displays the experimental comparison of two approaches for 55 extended VRPTW instances. The first column in

Table 5
Comparisons of the different AFS.

Ins	Best	Compared algorithms		dev	
		AFS	IAFS	AFS	IAFS
cc101	1108.61	1192.51	1108.61	7.57	**0.00**
cc102	909.36	909.36	912.03	**0.00**	0.29
cc103	797.28	866.59	797.28	8.69	**0.00**
cc104	642.69	684.46	642.69	6.50	**0.00**
cc105	950.81	1033.92	950.81	8.74	**0.00**
cc106	899.83	912.69	899.83	1.43	**0.00**
cc107	675.93	743.49	675.93	9.99	**0.00**
cc108	592.33	592.33	601.55	**0.00**	1.56
cc109	638.37	655.50	638.37	2.68	**0.00**
cc201	1434.43	1707.69	1434.43	19.05	**0.00**
cc202	856.38	923.80	856.38	7.87	**0.00**
cc203	484.93	557.69	484.93	15.00	**0.00**
cc204	421.59	481.73	421.59	14.26	**0.00**
cc205	855.04	855.04	896.92	**0.00**	4.90
cc206	727.03	808.15	727.03	11.16	**0.00**
cc207	489.35	682.15	489.35	39.40	**0.00**
cc208	623.66	677.59	623.66	8.65	**0.00**
cr101	1308.52	1441.03	1308.52	10.13	**0.00**
cr102	936.20	1012.08	936.20	8.10	**0.00**
cr103	679.96	680.89	679.96	0.14	**0.00**
cr104	458.86	475.81	458.86	3.70	**0.00**
cr105	761.73	775.13	761.73	1.76	**0.00**
cr106	679.61	728.46	679.61	7.19	**0.00**
cr107	527.96	557.92	527.96	5.68	**0.00**
cr108	407.72	409.69	407.72	0.48	**0.00**
cr109	632.15	662.46	632.15	4.79	**0.00**
cr110	490.59	524.09	490.59	6.83	**0.00**
cr111	524.39	592.39	524.39	12.97	**0.00**
cr112	430.51	509.11	430.51	18.26	**0.00**
cr201	730.49	738.29	730.49	1.07	**0.00**
cr202	619.40	681.02	619.40	9.95	**0.00**
cr203	513.92	534.51	513.92	4.01	**0.00**
cr204	351.36	351.36	364.91	**0.00**	3.86
cr205	537.00	553.90	537.00	3.15	**0.00**
cr206	417.10	489.28	417.10	17.31	**0.00**
cr207	354.99	376.07	354.99	5.94	**0.00**
cr208	318.70	320.62	318.70	0.60	**0.00**
cr209	443.31	443.31	443.69	**0.00**	0.09
cr210	477.20	530.71	477.20	11.21	**0.00**
cr211	334.93	364.36	334.93	8.79	**0.00**
crc101	935.94	968.55	935.94	3.49	**0.00**
crc102	743.09	790.55	743.09	6.39	**0.00**
crc103	591.71	626.37	591.71	5.86	**0.00**
crc104	499.25	530.12	499.25	6.18	**0.00**
crc105	867.52	881.91	867.52	1.66	**0.00**
crc106	740.15	775.45	740.15	4.77	**0.00**
crc107	592.45	651.53	592.45	9.97	**0.00**
crc108	534.35	588.76	534.35	10.18	**0.00**
crc201	1014.66	1075.29	1014.66	5.98	**0.00**
crc202	753.63	793.43	753.63	5.28	**0.00**
crc203	534.56	550.88	534.56	3.05	**0.00**
crc204	429.09	471.00	429.09	9.77	**0.00**
crc205	786.05	938.15	786.05	19.35	**0.00**
crc206	648.87	688.36	648.87	6.08	**0.00**
crc207	551.04	612.87	551.04	11.22	**0.00**
Mean	657.46	707.02	658.67	7.56	**0.19**

the table shows the instance name, the second column gives the best value for heuristic II to compare to heuristic I, and the next two columns show the best fitness value for each instance obtained by the two heuristics. The last two columns give the mean square difference obtained by each heuristic with respect to the best value, and the calculation formula is as follows:

$$dev = (f_c - f_b)/f_b \times 100\% \quad (6)$$

It can be seen from Table 3 that (1) for solving the 55 instances, heuristic II can obtain 31 optimal values, while heuristic I can only obtain 24 better values; (2) the last line shows that, on average, the proposed heuristic II obtains 1.33, which is obviously better than the first heuristic; and (3) the comparison results verify that the preying heuristic II in the proposed IAFS algorithm shows better performance.

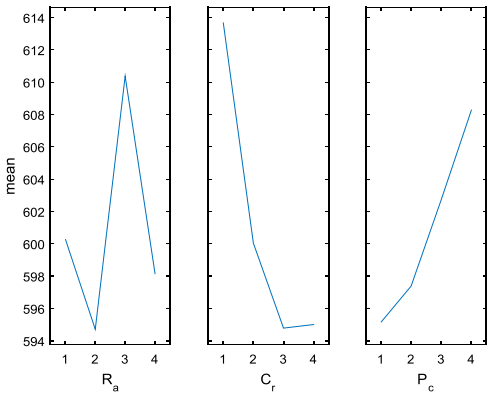


Fig. 6. Factor level trends of the three key parameters.

To further verify whether the difference is significant, we also take a multifactor analysis of variance (ANOVA) where the compared heuristics are considered as factors. Fig. 7(a) shows the means and the 95% LSD interval for fitness values of the two heuristics. We can see that the p-value is close to zero, revealing significant differences between the compared heuristics.

5.4. Efficiency of the following behavior

To verify the effectiveness of the proposed following behavior, we compare the two following heuristics, i.e., following heuristic I and following heuristic II. Heuristic I is the canonical AFS algorithm following behavior, in which the artificial fish moves towards the center of the fish swarm, and heuristic II is an improved following behavior, in which the artificial fish moves towards the center of the fish swarm with the probability of $1-P_c$ and in any direction with the probability of P_c.

Table 4 displays the experimental comparison of two heuristics for 55 extended VRPTW instances. The first column in the table shows the instance name, the second column gives the best value for heuristic II to compare to heuristic I, and the next two columns show the best fitness value for each instance obtained by the two heuristics. The last two columns give the mean square difference obtained by each heuristic with respect to the best value.

It can be seen from Table 4 that (1) for solving the 55 instances, heuristic II can obtain 36 optimal values, while heuristic I can only obtain 19 better values; (2) the last line shows that, on average, the proposed heuristic obtains 1.72, which is obviously better than the heuristic I; and (3) the comparison results verify that the proposed following heuristic shows better performance.

Fig. 7(b) shows the means and the 95% LSD interval for fitness values of the two heuristics. We can see that the p-value is close to zero, revealing significant differences between the two following heuristics.

5.5. Efficiency of the initialization heuristic

To verify the effectiveness of the proposed initialization heuristic, we compare the two following heuristics, i.e., IAIS-NI with the canonical PFIH heuristic, and the IAIS with the proposed initialization heuristic discussed in Section 4.3. Fig. 7(c) shows the means and the 95% LSD interval for fitness values of the two heuristics, where the resulted p-value is 0.0172. From the comparison analysis, it can be concluded that the proposed initialization heuristic is efficient and show competitive performance compared with the canonical efficient PFIH heuristic.

Table 6
Comparisons with other efficient algorithms.

Ins	Best	Compared algorithms				dev			
		HEA	TS	ALNS	IAFS	HEA	TS	ALNS	IAFS
cc101	1108.61	1498.95	1344.91	1140.17	1108.61	35.21	21.31	2.85	**0.00**
cc102	912.03	1105.81	976.94	977.30	912.03	21.25	7.12	7.16	**0.00**
cc103	781.37	881.54	781.37	954.44	797.28	12.82	**0.00**	22.15	2.04
cc104	642.69	685.18	658.73	720.16	642.69	6.61	2.50	12.05	**0.00**
cc105	858.49	1214.18	1100.91	858.49	950.81	41.43	28.24	**0.00**	10.75
cc106	899.83	1041.22	1010.31	926.93	899.83	15.71	12.28	3.01	**0.00**
cc107	675.93	826.42	704.85	773.95	675.93	22.26	4.28	14.50	**0.00**
cc108	581.15	658.68	581.15	746.14	601.55	13.34	**0.00**	28.39	3.51
cc109	638.37	738.44	668.99	754.49	638.37	15.68	4.80	18.19	**0.00**
cc201	1434.43	1912.53	2168.87	1700.27	1434.43	33.33	51.20	18.53	**0.00**
cc202	856.38	971.62	1106.50	986.13	856.38	13.46	29.21	15.15	**0.00**
cc203	484.93	519.21	534.98	690.01	484.93	7.07	10.32	42.29	**0.00**
cc204	400.69	430.47	400.69	637.35	421.59	7.43	**0.00**	59.06	5.22
cc205	896.92	1080.68	1308.13	956.61	896.92	20.49	45.85	6.65	**0.00**
cc206	786.26	786.26	830.92	845.97	787.03	**0.00**	5.68	7.59	0.10
cc207	489.35	685.10	683.27	874.63	489.35	40.00	39.63	78.73	**0.00**
cc208	555.31	644.21	555.31	854.85	623.66	16.01	**0.00**	53.94	12.31
cr101	1285.53	1596.23	1618.02	1285.53	1308.52	24.17	25.86	**0.00**	1.79
cr102	936.20	1109.41	1101.30	953.18	936.20	18.50	17.64	1.81	**0.00**
cr103	679.96	770.93	697.55	704.14	679.96	13.38	2.59	3.56	**0.00**
cr104	458.86	523.35	481.39	494.59	458.86	14.05	4.91	7.79	**0.00**
cr105	761.73	879.57	850.40	767.98	761.73	15.47	11.64	0.82	**0.00**
cr106	679.61	784.04	799.17	723.02	679.61	15.37	17.59	6.39	**0.00**
cr107	527.96	643.43	596.04	643.97	527.96	21.87	12.89	21.97	**0.00**
cr108	402.94	487.47	402.94	473.63	407.72	20.98	**0.00**	17.54	1.19
cr109	632.15	782.20	748.46	713.97	632.15	23.74	18.40	12.94	**0.00**
cr110	490.59	587.64	527.78	574.07	490.59	19.78	7.58	17.02	**0.00**
cr111	524.39	674.03	607.08	583.09	524.39	28.54	15.77	11.19	**0.00**
cr112	430.51	587.82	524.58	551.33	430.51	36.54	21.85	28.06	**0.00**
cr201	849.03	894.62	852.91	849.03	851.49	5.37	0.46	**0.00**	0.29
cr202	619.40	775.06	749.29	694.55	619.40	25.13	20.97	12.13	**0.00**
cr203	513.92	601.65	617.89	542.13	513.92	17.07	20.23	5.49	**0.00**
cr204	338.60	359.22	338.60	419.68	364.91	6.09	**0.00**	23.95	7.77
cr205	537.00	537.23	559.71	626.33	537.00	0.04	4.23	16.63	**0.00**
cr206	417.10	469.89	474.77	517.56	417.10	12.66	13.83	24.08	**0.00**
cr207	354.99	388.78	380.12	435.59	354.99	9.52	7.08	22.71	**0.00**
cr208	318.70	340.13	329.18	349.39	318.70	6.72	3.29	9.63	**0.00**
cr209	413.98	473.20	413.98	579.92	483.69	14.31	**0.00**	40.08	16.84
cr210	477.20	569.13	522.85	581.25	477.20	19.26	9.57	21.80	**0.00**
cr211	334.93	365.82	349.92	415.05	334.93	9.22	4.48	23.92	**0.00**
crc101	915.65	1140.53	1098.36	915.65	935.94	24.56	19.95	**0.00**	2.22
crc102	743.09	868.64	769.68	803.54	743.09	16.90	3.58	8.14	**0.00**
crc103	591.71	708.40	691.75	620.82	591.71	19.72	16.91	4.92	**0.00**
crc104	499.25	632.74	578.88	567.18	499.25	26.74	15.95	13.61	**0.00**
crc105	867.52	966.92	960.76	912.53	867.52	11.46	10.75	5.19	**0.00**
crc106	740.15	976.02	903.83	822.90	740.15	31.87	22.11	11.18	**0.00**
crc107	592.45	727.46	700.41	652.83	592.45	22.79	18.22	10.19	**0.00**
crc108	534.35	713.56	697.54	625.22	534.35	33.54	30.54	17.01	**0.00**
crc201	917.38	990.43	935.61	917.38	1014.66	7.96	1.99	**0.00**	10.60
crc202	680.55	850.94	680.55	763.52	753.63	25.04	**0.00**	12.19	10.74
crc203	534.56	555.67	536.90	676.66	534.56	3.95	0.44	26.58	**0.00**
crc204	429.09	504.76	477.12	521.76	429.09	17.63	11.19	21.60	**0.00**
crc205	786.05	956.68	934.48	859.50	786.05	21.71	18.88	9.34	**0.00**
crc206	648.87	666.23	697.68	710.09	648.87	2.68	7.52	9.44	**0.00**
crc207	551.04	621.15	611.83	616.88	551.04	12.72	11.03	11.95	**0.00**
Mean	654.90	777.48	749.75	742.97	664.64	17.80	12.59	16.02	**1.55**

5.6. Comparison with the canonical AFS algorithm

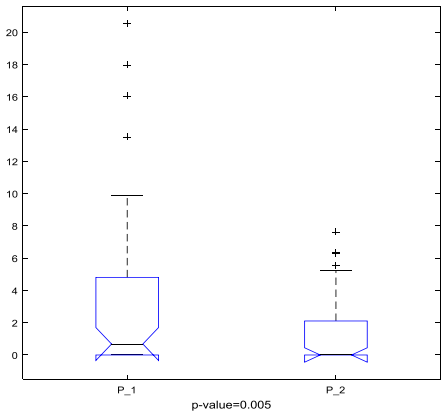
To verify the effectiveness of the IAFS algorithm, we compare the improved AFS algorithm with the canonical AFS algorithm. Table 5 displays the experimental comparison of two algorithms for 55 extended VRPTW instances. The first column in the table shows the instance name, the second column gives the best value for IAFS to compare to canonical AFS, and the next two columns show the best fitness value for each instance obtained by the two algorithms. The last two columns give the mean square difference obtained by each heuristic with respect to the best value.

It can be seen from Table 5 that (1) for solving the 55 instances, the IAFS can obtain 50 optimal values, while the canonical AFS can only obtain 5 better values; (2) the last line shows that, on average, the IAFS obtains 0.19, which is obviously better than the canonical AFS; and (3) the comparison results verify that the IAFS shows better performance.

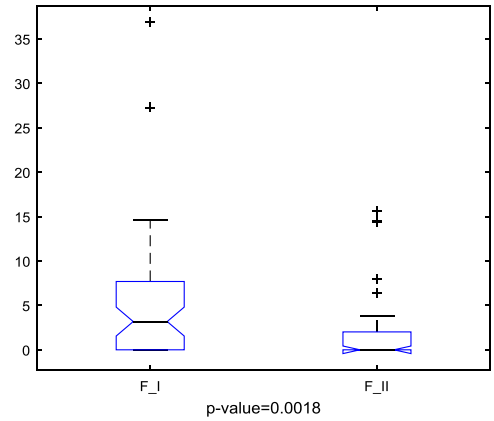
Fig. 7(d) shows the means and the 95% LSD interval for fitness values of the two heuristics. We can see that the p-value is close to zero, revealing significant differences between the algorithms.

5.7. Comparisons with other efficient algorithms

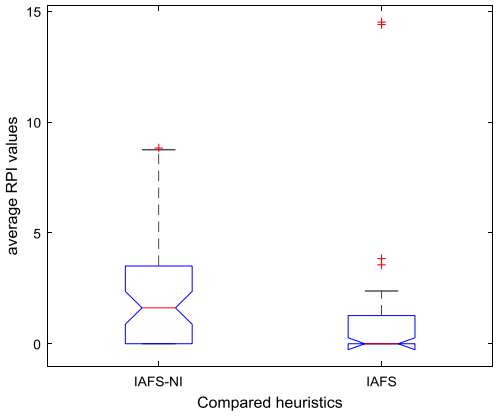
To evaluate the performance of the IAFS algorithm that is presented in this paper, the hybrid evolutionary algorithm (HEA) [10], TS [9] and the adaptive large neighborhood search heuristic (ALNS) [75] are selected as comparison algorithms. The main reasons to select them as the comparison algorithms are as follows: (1) HEA was proposed to solve the VRPTW with heterogeneous



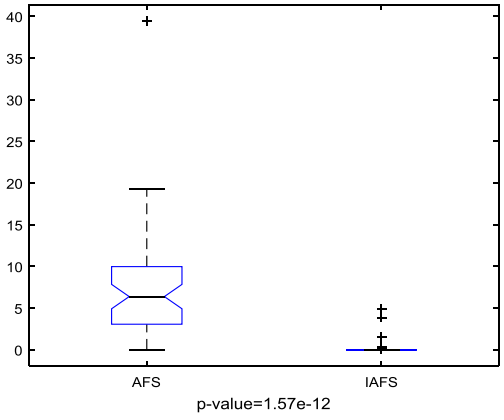
(a) Analysis of the preying heuristics



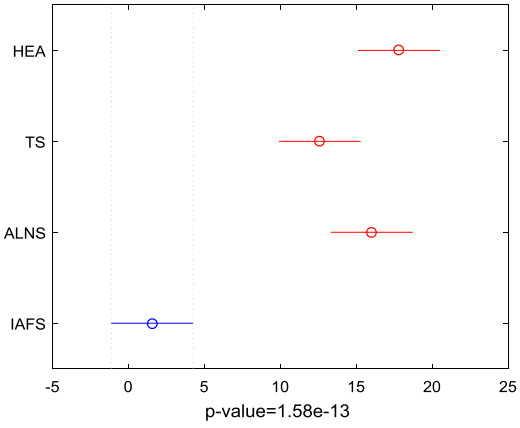
(b) Analysis of the following heuristics



(c) Analysis of the initialization heuristics



(d) Analysis of AFS and IAFS



(e) Analysis of the IAFS and other efficient algorithms

Fig. 7. Detailed comparisons by using ANOVA ways.

fleet vehicle constraints, and verified to be an efficient algorithm for this type of problem; and (2) TS is also an efficient algorithm to solve the VRPTW. Because the considered HVRPTW in this study is an extension version of the VRPTW, the two compared algorithms can be extended to solve the considered problem with a simple change. Therefore, in this study, we recoded the two compared HEA and TS algorithms, with the main components proposed in their literature and modified them to adapt to solve the HVRPTW; and (3) ALNS is one of the efficient algorithms to solve one certain extension of the HVRPTW problems. Therefore, we recoded all the compared algorithms and adapted the main components and parameters in their literatures. In addition, to make a fair comparison, the four compared algorithms are performed 30 independent runs and the best fitness values for each instance are collected for detailed comparisons.

Table 6 shows the results of the comparisons, where the first column shows the instance name, and the second column lists the best values for each instance collected by the four compared algorithms. The next four columns list the fitness values for each instance collected by the four compared algorithms, i.e., HEA, TS, ALNS and IAFS algorithm, respectively. Then, the last four columns show the *dev* between the results of the four comparison

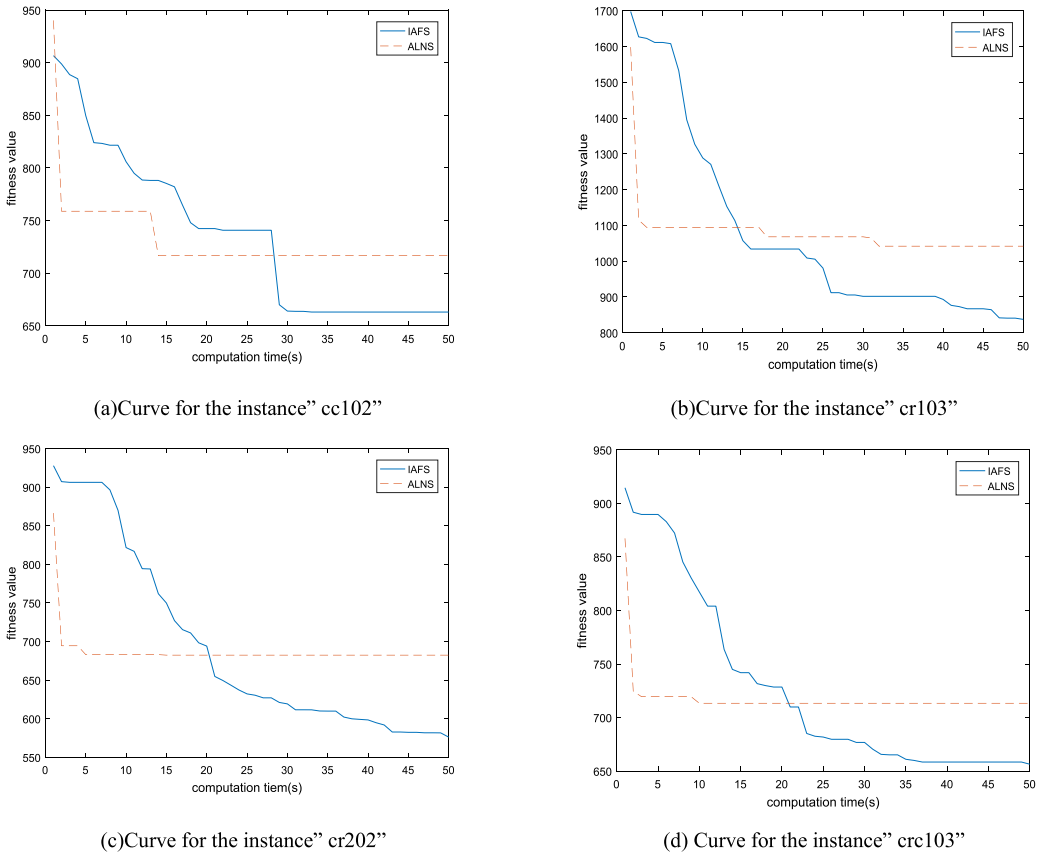


Fig. 8. Convergence curve comparisons of IAFS and ALNS.

Fig. 9. Gantt chart for the best solution for "cc101".

algorithms and the optimal value. It can be concluded from Table 6 that: (1) the proposed IAFS obtained 41 better values out of the given 55 instances, which is obviously better than the second best algorithm, i.e., the TS algorithm, which can only obtain 8 better results; (2) the average performance of the last line shows that IAFS obtains an average of 1.55, which is obviously better

than the other three algorithms. For example, the second best algorithm TS obtain an average value of 12.59, which is about ten times more than the proposed IAFS algorithm; and (3) for comparison with the recently published ALNS algorithm, which is a non-population based algorithm and also an efficient algorithm for solving the special version of VRPTW problems, we can see that, the ALNS algorithm obtained an average value of 16.02, which is obvious worse than the result of IAFS. Fig. 7(e) shows the multiple comparison result by using the ANOVA, which further verify that the proposed IAFS is significantly better than the other three algorithms.

Furthermore, to demonstrate the convergence abilities of the proposed algorithm, we also conduct detailed comparisons between IAFS and ALNS, and the convergence curves for the four instance with different features are described in Fig. 8. The four curves of the selected instances, including cc102, cr103, cr202, and crc103, are shown in Fig. 8(a)–(d), respectively. From the convergence curves for different types of instances, we can see that the proposed algorithm has efficient convergence abilities for solving the HVRPTW problems. Fig. 9 shows the Gantt chart of the best solution for "cc101". In Fig. 9, different colors illustrate the customers assigned on different type of vehicles. For example, the customer 81 is assigned on the refrigerated vehicle. The number right below the customer number is the service completion time. It can be observed from Fig. 9 that the obtained solution is feasible and efficient.

5.8. Experimental result analysis

From the detailed comparisons, we can see that the proposed components in the proposed IAFS algorithm are efficient, which makes the proposed algorithm show a better performance compared with other efficient algorithms, the main reasons are as follows: (1) a special encoding approach is designed to consider the problem feature with different type of vehicles, and the decoding process makes the solution feasible. Therefore, the decoding process will not need any repair process and therefore save the computational times; (2) the improved preying and following heuristics are developed to perform the exploitation and exploration tasks. With the proposed mutation and crossover operators, the proposed IAFS algorithm can enhance the local search and global search abilities; (3) a novel customer satisfaction level heuristic is embedded in the proposed algorithm, which makes the problem close to the reality, and the calculation process of the customer satisfaction level is utilized to reflect the urgency of the time window for each customer; (4) to further improve the performance of the algorithm, a shift heuristic is designed to increase the customer satisfaction level without increasing the energy consumption; and (5) a novel initialization method based on the canonical PFIH is proposed to generate initial solutions with better performance.

6. Conclusions

In this study, we consider a canonical VRP problem in the cold chain logistics system, where three special constraints are included, i.e., the time windows, different types of vehicles and different energy consumption of vehicles. To solve this type of NP-hard problem, we propose an improved artificial fish swarm algorithm, where the following features are developed: (1) a novel encoding approach containing two-dimensional vectors is developed; (2) to apply the canonical AFS algorithm for solving the discrete optimization problem, improved preying, following, and swarming behaviors are proposed; (3) a right shift heuristic is designed to improve the performance; and (4) an improved PFIH-based initialization method is utilized.

Future works are mainly focused on the following: (1) to apply the proposed algorithm to solve more realistic VRP with other constraints, such as distributed optimization problem in a cloud system [76]; and (2) to consider multi-objective optimization method and to minimize energy consumption criteria.

CRediT authorship contribution statement

Mei-xian Song: Proposed an improved artificial fish swarm (IAFS) algorithm for the considered problem, Design several heuristics for the problem, Design the simulation tests, Make a detailed analysis for the results. **Jun-qing Li:** Code the compared algorithms and test the simulation, Design several heuristics for the problem, Test all the compared algorithms. **Yun-qi Han:** Give a detailed analysis of the cold chain logistics, Propose the VRPTW problem in the cold chain logistics, Analyzed the algorithm's performance. **Yu-yan Han:** Analyze the problem features and proposed problem-specific heuristic, Design the algorithm parameter and test them. **Li-li Liu:** Give detailed report of the related literatures, Written the problem descriptions and examples. **Qun Sun:** Verify the mathematical model, Recode and test the compared algorithms, Give a carefully check for the grammar in the paper.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was partially supported by the National Science Foundation of China under Grants 61773246, and 61803192; the Shandong Province Higher Educational Science and Technology Program, China (J17KZ005); the Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, China (K93-9-2017-02); the State Key Laboratory of Synthetical Automation for Process Industries, China (PAL-N201602), and major basic research projects in Shandong, China (ZR2018ZB0419).

References

[1] G.B. Dantzig, J.H. Ramser, The truck dispatching problem, Manag. Sci. 6 (1959) 80–91.
[2] M.M. Solomon, Algorithms for the vehicle routing and scheduling problems with time window constraints, Oper. Res. 35 (1987) 254–265.
[3] K.C. Tan, L.H. Lee, Q.L. Zhu, K. Ou, Heuristic methods for vehicle routing problem with time windows, Artif. Intell. Eng. 15 (2001) 281–295.
[4] R. Liu, Y. Tao, X. Xie, An adaptive large neighborhood search heuristic for the vehicle routing problem with time windows and synchronized visits, Comput. Oper. Res. 101 (2019) 250–262.
[5] Y. Xia, Z. Fu, Improved tabu search algorithm for the open vehicle routing problem with soft time windows and satisfaction rate, Cluster Comput. (2018) 1–9.
[6] S. Naccache, J.F. Côté, L.C. Coelho, The multi-pickup and delivery problem with time windows, European J. Oper. Res. 269 (2018) 353–362.
[7] D.C. Paraskevopoulos, P.P. Repoussis, C.D. Tarantilis, G. Ioannou, G.P. Prastacos, A reactive variable neighborhood tabu search for the heterogeneous fleet vehicle routing problem with time windows, J. Heuristics 14 (2008) 425–455.
[8] P. Belfiore, H.T. Yoshida Yoshizaki, Scatter search for a real-life heterogeneous fleet vehicle routing problem with time windows and split deliveries in Brazil, European J. Oper. Res. 199 (2009) 750–758.
[9] J. Jiang, K.M. Ng, K.L. Poh, K.M. Teo, Vehicle routing problem with a heterogeneous fleet and time windows, Expert Syst. Appl. 41 (2014) 3748–3760.
[10] Ç. Koç, T. Bektaş, O. Jabali, G. Laporte, A hybrid evolutionary algorithm for heterogeneous fleet vehicle routing problems with time windows, Comput. Oper. Res. 64 (2015) 11–27.

[11] R. Dondo, J. Cerdá, A cluster-based optimization approach for the multi-depot heterogeneous fleet vehicle routing problem with time windows, European J. Oper. Res. 176 (2007) 1478–1507.
[12] M. Adelzadeh, V.M. Asl, M. Koosha, A mathematical model and a solving procedure for multi-depot vehicle routing problem with fuzzy time window and heterogeneous vehicle, Int. J. Adv. Manuf. Technol. 75 (2014) 793–802.
[13] Y. Xiao, A. Konak, The heterogeneous green vehicle routing and scheduling problem with time-varying traffic congestion, Transp. Res. e-Logist. Transp. Rev. 88 (2016) 146–166.
[14] Y. Yu, S. Wang, J. Wang, M. Huang, A branch-and-price algorithm for the heterogeneous fleet green vehicle routing problem with time windows, Transp. Res. B 122 (2019) 511–527.
[15] G. Hiermann, J. Puchinger, S. Ropke, R.F. Hartl, The electric fleet size and mix vehicle routing problem with time windows and recharging stations, European J. Oper. Res. 252 (2016) 995–1018.
[16] J. Li, D. Wang, J. Zhang, Heterogeneous fixed fleet vehicle routing problem based on fuel and carbon emissions, J. Cleaner Prod. 201 (2018) 896–908.
[17] F. Belmecheri, C. Prins, F. Yalaoui, L. Amodeo, Particle swarm optimization algorithm for a vehicle routing problem with heterogeneous fleet, mixed backhauls, and time windows, J. Intell. Manuf. 24 (2013) 775–789.
[18] M.N. Kritikos, G. Ioannou, The heterogeneous fleet vehicle routing problem with overloads and time windows, Int. J. Prod. Econ. 144 (2013) 68–75.
[19] Z. Jia, J. Yu, X. Ai, X. Xu, D. Yang, Cooperative multiple task assignment problem with stochastic velocities and time windows for heterogeneous unmanned aerial vehicles using a genetic algorithm, Aerosp. Sci. Technol. 76 (2018) 112–125.
[20] S.F. Ghannadpour, A. Zarrabi, Multi-objective heterogeneous vehicle routing and scheduling problem with energy minimizing, Swarm Evol. Comput. 44 (2019) 728–747.
[21] Z. Wang, Y. Li, X. Hu, A heuristic approach and a tabu search for the heterogeneous multi-type fleet vehicle routing problem with time windows and an incompatible loading constraint, Comput. Ind. Eng. 89 (2015) 162–176.
[22] X. J. Guo, S. Wang, M. Fan, Forward and reverse logistics network and route planning under the environment of low-carbon emissions: a case study of Shanghai fresh food E-commerce enterprises, Comput. Ind. Eng. 106 (2017) 351–360.
[23] L. Zhang, Y. Wang, T. Fei, H. Ren, The research on low carbon logistics routing optimization based on DNA-ant colony algorithm, Discrete Dyn. Nat. Soc. (2014) 1–13.
[24] L.Y. Zhang, M.L. Tseng, C.H. Wang, C. Xiao, T. Fei, Low-carbon cold chain logistics using ribonucleic acid-ant colony optimization algorithm, J. Cleaner Prod. (2019).
[25] L. Mingyong, C. Erbao, An improved differential evolution algorithm for vehicle routing problem with simultaneous pickups and deliveries and time windows, Eng. Appl. Artif. Intell. 23 (2) (2010) 188–195.
[26] Y. Marinakis, M. Marinaki, A. Migdalas, A multi-adaptive particle swarm optimization for the vehicle routing problem with time windows, Inform. Sci. 481 (2019) 311–329.
[27] J. Chen, J. Shi, A multi-compartment vehicle routing problem with time windows for urban distribution–A comparison study on particle swarm optimization algorithms, Comput. Ind. Eng. 133 (2019) 95–106.
[28] T. Vidal, T.G. Crainic, M. Gendreau, A hybrid genetic algorithm for multidepot and periodic vehicle routing problems, Oper. Res. 60 (3) (2012) 611–624.
[29] Y.A. Koskosidis, W.B. Powell, M.M. Solomon, An optimization-based heuristic for vehicle routing and scheduling with soft time window constraints, Transp. Sci. 26 (1992) 69–85.
[30] T.R. Sexton, Y.M. Choi, Pickup and delivery of partial loads with "soft" time windows, Am. J. Math. Manag. Sci. 6 (1986) 369–398.
[31] C.D. Tarantilis, C.T. Kiranoudis, A meta-heuristic algorithm for the efficient distribution of perishable foods, J. Food Eng. 50 (2001) 1–9.
[32] G. Zhang, W. Habenicht, W.E.L. Spieß, Improving the structure of deep frozen and chilled food chain with tabu search procedure, J. Food Eng. 60 (2003) 67–79.
[33] C.I. Hsu, S.F. Hung, H.C. Li, Vehicle routing problem with time-windows for perishable food delivery, J. Food Eng. 80 (2007) 465–475.
[34] C.I. Hsu, W.T. Chen, Optimizing fleet size and delivery scheduling for multi-temperature food distribution, Appl. Math. Model. 38 (2014) 1077–1091.
[35] J.Q. Li, M.X. Song, L. L. Wang, P.Y. Duan, Y.Y. Han, H.Y. Sang, Q.K. Pan, Hybrid artificial bee colony algorithm for a parallel batching distributed flow shop problem with deteriorating jobs, IEEE T. Cybern. (2019) http://dx.doi.org/10.1109/TCYB.2019.2943606.
[36] J.Q. Li, Q.K. Pan, M.F. Tasgetiren, A discrete artificial bee colony algorithm for the multi-objective flexible job-shop scheduling problem with maintenance activities, Appl. Math. Model. 38 (2014) 1111–1132.
[37] J.Q. Li, X.R. Tao, B.X. Jia, Y.Y. Han, C. Liu, P. Duan, Z.X. Zheng, H.Y. Sang, Efficient multi-objective algorithm for the lot-streaming hybrid flowshop with variable sub-lots, Swarm Evol. Comput. (2019) http://dx.doi.org/10.1016/j.swevo.2019.100600.
[38] Z.X. Zheng, J.Q. Li, Y.Y. Han, An improved invasive weed optimization algorithm for solving dynamic economic dispatch problems with valve-point effects, J. Exp. Theor. Artif. Intell. (2019) http://dx.doi.org/10.1080/0952813X.2019.1673488.
[39] Y.Y. Han, J.Q. Li, Deng J.W., Li C.Y., J. Tian, B. Zhang, C.G. Wang, An improved jaya algorithm for solving the flexible job shop scheduling problem with transportation and setup times, Knowl-Based. Syst. (2020) http://dx.doi.org/10.1016/j.knosys.2020.106032.
[40] J.Q. Li, Y.Q. Han, P.Y. Duan, Y.Y. Han, B. Niu, C.D. Li, Z.X. Zheng, Y.P. Liu, A metaheuristic algorithm for solving vehicle routing problems with time window and synchronized visit constraints in a prefabricated system, J. Cleaner Prod. 250 (2020) http://dx.doi.org/10.1016/j.jclepro.2019.119464.
[41] W.Y. Szeto, Y. Wu, S.C. Ho, An artificial bee colony algorithm for the capacitated vehicle routing problem, European J. Oper. Res. 215 (2011) 126–135.
[42] S. Zhang, C.K.M. Lee, K.L. Choy, W. Ho, W.H. Ip, Design and development of a hybrid artificial bee colony algorithm for the environmental vehicle routing problem, Transp. Res. D 31 (2014) 85–99.
[43] Z.X. Zheng, J.Q. Li, P.Y. Duan, Optimal chiller loading by improved artificial fish swarm algorithm for energy saving, Math. Comput. Simulation 155 (2019) 227–243.
[44] M.A.K. Azad, A.M.A. Rocha, E.M. Fernandes, Improved binary artificial fish swarm algorithm for the 0–1 multidimensional knapsack problems, Swarm Evol. Comput. 14 (2014) 66–75.
[45] M.A.K. Azad, A.M.A. Rocha, E.M. Fernandes, Solving large 0–1 multidimensional knapsack problems by a new simplified binary artificial fish swarm algorithm, J. Math. Model. Algor. Oper. Res. 14 (2015) 313–330.
[46] D.Y. Zhuang, K. Ma, C.A. Tang, Z.Z. Liang, K.K. Wang, Z.W. Wang, Mechanical parameter inversion in tunnel engineering using support vector regression optimized by multi-strategy artificial fish swarm algorithm, Tunn. Undergr. Space Technol. 83 (2019) 425–436.
[47] W. Zhao, C. Du, S. Jiang, An adaptive multiscale approach for identifying multiple flaws based on XFEM and a discrete artificial fish swarm algorithm, Comput. Methods Appl. Mech. Engrg. 339 (2018) 341–357.
[48] Z. Zhang, K. Wang, L. Zhu, Y. Wang, A Pareto improved artificial fish swarm algorithm for solving a multi-objective fuzzy disassembly line balancing problem, Expert Syst. Appl. 86 (2017) 165–176.
[49] X.Y. Luan, Z.P. Li, T.Z. Liu, A novel attribute reduction algorithm based on rough set and improved artificial fish swarm algorithm, Neurocomputing 174 (2016) 522–529.
[50] M.F.P. Costa, A.M.A. Rocha, E.M. Fernandes, An artificial fish swarm algorithm based hyperbolic augmented Lagrangian method, J. Comput. Appl. Math. 259 (2014) 868–876.
[51] Y. Li, H. Soleimani, M. Zohal, An improved ant colony optimization algorithm for the multi-depot green vehicle routing problem with multiple objectives, J. Cleaner Prod. (2019) http://dx.doi.org/10.1016/j.jclepro.2019.03.185.
[52] H. Zhang, Q. Zhang, L. Ma, Z. Zhang, Y. Liu, A hybrid ant colony optimization algorithm for a multi-objective vehicle routing problem with flexible time windows, Inform. Sci. 490 (2019) 166–190.
[53] B. Yu, Z.Z. Yang, An ant colony optimization model: The period vehicle routing problem with time windows, Transp. Res. e-Logist. Transp. Rev. 47 (2011) 166–181.
[54] J. Long, Z. Sun, P.M. Pardalos, Y. Hong, S. Zhang, C. Li, A hybrid multi-objective genetic local search algorithm for the prize-collecting vehicle routing problem, Inform. Sci. 478 (2019) 40–61.
[55] D. Rezgui, J. Siala-Chaouachi, W. Aggoune-Mtalaa, H. Bouziri, Application of a variable neighborhood search algorithm to a fleet size and mix vehicle routing problem with electric modular vehicles, Comput. Ind. Eng. 130 (2019) 537–550.
[56] F. Arnold, K. Sörensen, Knowledge-guided local search for the vehicle routing problem, Comput. Oper. Res. 105 (2019) 32–46.
[57] M. Qiu, Z. Fu, R. Eglese, Q. Tang, A tabu search algorithm for the vehicle routing problem with discrete split deliveries and pickups, Comput. Oper. Res. 100 (2018) 102–116.
[58] D.S. Lai, O.C. Demirag, J.M. Leung, A tabu search heuristic for the heterogeneous vehicle routing problem on a multigraph, Transp. Res. e-Logist. Transp. Rev. 86 (2016) 32–52.
[59] T. Hintsch, S. Irnich, Large multiple neighborhood search for the clustered vehicle-routing problem, European J. Oper. Res. 270 (2018) 118–131.
[60] J. Brandão, Iterated local search algorithm with ejection chains for the open vehicle routing problem with time windows, Comput. Ind. Eng. 120 (2018) 146–159.
[61] P. Vansteenwegen, M. Mateo, An iterated local search algorithm for the single-vehicle cyclic inventory routing problem, European J. Oper. Res. 237 (2014) 802–813.

[62] H. Rau, S.D. Budiman, G.A. Widyadana, Optimization of the multi-objective green cyclical inventory routing problem using discrete multi-swarm PSO method, Transp. Res. e-Logist. Transp. Rev. 120 (2018) 51–75.
[63] L. Hou, D. Li, D. Zhang, Ride-matching and routing optimisation: Models and a large neighbourhood search heuristic, Transp. Res. e-Logist. Transp. Rev. 118 (2018) 143–162.
[64] I. Dayarian, T.G. Crainic, M. Gendreau, W. Rei, An adaptive large-neighborhood search heuristic for a multi-period vehicle routing problem, Transp. Res. e-Logist. Transp. Rev. 95 (2016) 95–123.
[65] Y. Xiao, A. Konak, A genetic algorithm with exact dynamic programming for the green vehicle routing & scheduling problem, J. Cleaner Prod. 167 (2017) 1450–1463.
[66] A. Baniamerian, M. Bashiri, F. Zabihi, Two phase genetic algorithm for vehicle routing and scheduling problem with cross-docking and time windows considering customer satisfaction, J. Ind. Eng. Int. 14 (2018) 15–30.
[67] M. Afshar-Bakeshloo, A. Mehrabi, H. Safari, M. Maleki, F. Jolai, A green vehicle routing problem with customer satisfaction criteria, J. Ind. Eng. Int. 12 (2016) 529–544.
[68] R. Gupta, B. Singh, D. Pandey, Multi-objective fuzzy vehicle routing problem: a case study, Int. J. Contemp. Math. Sci. 5 (29) (2010) 1439–1454.
[69] M. Barkaoui, J. Berger, A. Boukhtouta, Customer satisfaction in dynamic vehicle routing problem with time windows, Appl. Soft Comput. 35 (2015) 423–432.
[70] F. Guerriero, R. Surace, V. Loscri, A multi-objective approach for unmanned aerial vehicle routing problem with soft time windows constraints, Appl. Math. Model. 38 (3) (2014) 839–852.
[71] O. Bräysy, M. Gendreau, Vehicle routing problem with time windows, Part I: Route construction and local search algorithms, Transp. Sci. 39 (2005) 104–118.
[72] J.Y. Potvin, J.M. Rousseau, A parallel route building algorithm for the vehicle routing and scheduling problem with time windows, European J. Oper. Res. 66 (1993) 331–340.
[73] G. Ioannou, M. Kritikos, G. Prastacos, A greedy look-ahead heuristic for the vehicle routing problem with time windows, J. Oper. Res. Soc. 52 (2001) 523–537.
[74] J.B. Atkinson, A greedy look-ahead heuristic for combinatorial optimization: An application to vehicle scheduling with time windows, J. Oper. Res. Soc. 45 (1994) 673–684.
[75] R. Liu, Z. Jiang, A hybrid large-neighborhood search algorithm for the cumulative capacitated vehicle routing problem with time-window constraints, Appl. Soft Comput. (2019) http://dx.doi.org/10.1016/j.jclepro.2019.03.185.
[76] J.Q. Li, Y. Han, A hybrid multi-objective artificial bee colony algorithm for flexible task scheduling problems in cloud computing system, Cluster Comput. (2020) http://dx.doi.org/10.1007/s10586-019-03022-z.