




# Hybrid evolutionary optimization for takeaway order selection and delivery path planning utilizing habit data

Min-Xia Zhang<sup>1</sup> · Jia-Yu Wu<sup>1</sup> · Xue Wu<sup>1</sup> · Yu-Jun Zheng<sup>2</sup> 

Received: 10 April 2021 / Accepted: 19 May 2021 / Published online: 2 June 2021  
© The Author(s) 2021

## Abstract

The last years have seen a rapid growth of the takeaway delivery market, which has provided a lot of jobs for deliverymen. However, increasing numbers of takeaway orders and the corresponding pickup and service points have made order selection and path planning a key challenging problem to deliverymen. In this paper, we present a problem integrating order selection and delivery path planning for deliverymen, the objective of which is to maximize the revenue per unit time subject to maximum delivery path length, overdue penalty, reward/penalty for large/small number of orders, and high customer scoring reward. Particularly, we consider uncertain order ready time and customer satisfaction level, which are estimated based on historical habit data of stores and customers using a machine-learning approach. To efficiently solve this problem, we propose a hybrid evolutionary algorithm, which adapts the water wave optimization (WVO) metaheuristic to evolve solutions to the main order selection problem and employs tabu search to route the delivery path for each order selection solution. Experimental results on test instances constructed based on real food delivery application data demonstrate the performance advantages of the proposed algorithm compared to a set of popular metaheuristic optimization algorithms.

**Keywords** Takeaway delivery · Order selection · Path planning · Evolutionary optimization · Water wave optimization (WVO) · Machine learning

## Introduction

In our modern society, deep labor-division and fast-paced lifestyles have made most people hard to find time to cook for themselves. Therefore, more and more people resort to takeaway (take-out) food, which can be selected online and brought by deliverymen to their home or offices. This requirement has boosted the food takeaway market in the last years. According to data from the Statista company [36], in the UK, 2019, the total food service delivery market value was around 8.5 billion British pounds, 55% of which belonged to online orders. In China, from 2015 to 2019, the total amount of takeaway orders increased from 134.8 billion RMB yuan to 603.5 billion, the penetration (i.e., the ratio of the total takeaway order amount to the national catering revenue) increased

from 4.2 to 14.2% (Fig. 1), and the number of online takeaway customers reached 421 million in 2019, accounting for 49.3% of the total netizens (Fig. 2, data from Trustdata [38]). Nowadays, takeaway has been one of the most popular and fastest-growing service industries in the country.

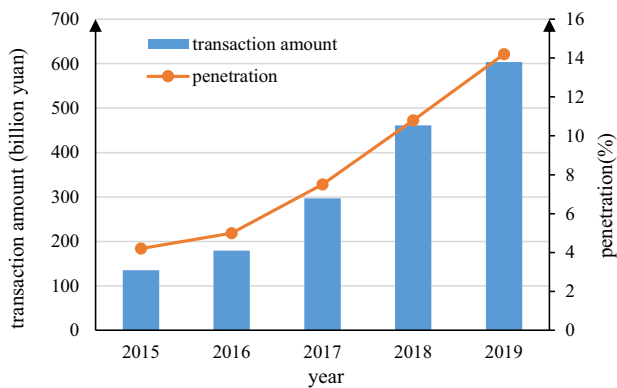
The takeaway industry has provided numerous jobs, particularly deliveryman jobs, for the society. In popular online takeaway ordering and delivery platforms, such as Meituan Waimai and Baidu's Eleme, the typical workflow can be described as follows (as illustrated in Fig. 3):

1. Customers place orders online;
2. Takeaway stores receive the orders, determine which orders they accept, and post the accepted order information online (visible to deliverymen);
3. Deliverymen explore the candidate orders, among which select those they want to deliver;
4. Deliverymen go to the stores, and if orders are ready, pick up the orders and deliver them to the corresponding customers.

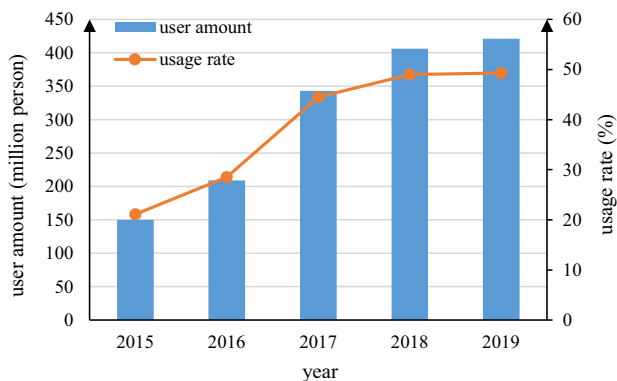
✉ Yu-Jun Zheng  
yujun.zheng@computer.org

<sup>1</sup> College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou, China

<sup>2</sup> School of Information Science and Engineering, Hangzhou Normal University, Hangzhou, China



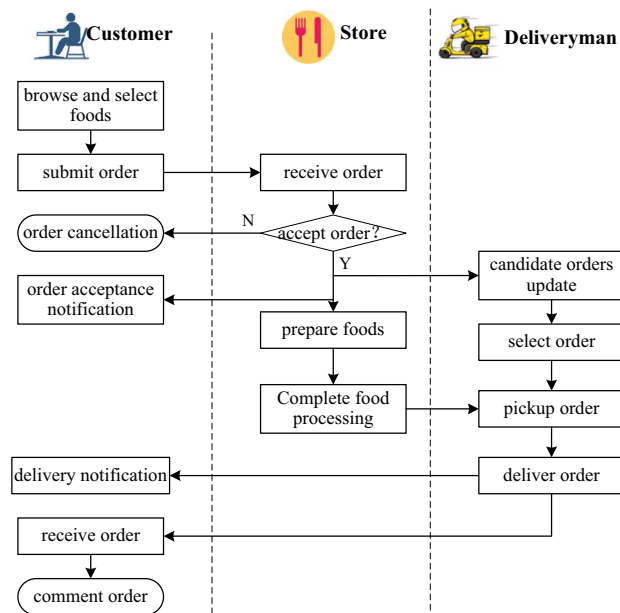
**Fig. 1** The developments of takeaway industry in China from 2015 to 2019



**Fig. 2** The scale and utilization rate of online takeaway users in China from 2015 to 2019

With the rapid growth of the takeaway industry, the workload of deliverymen increases dramatically. For example, in 2019, the average daily number of takeaway orders sent per deliveryman was around 40, and the average delivery time per order was around 30 min. Therefore, to improve the working efficiency and the corresponding revenue, every deliveryman wants to optimize his order selection and delivery path routing decision. However, the number of candidate orders is often large, different orders have different delivery fees, and their pickup points (i.e., takeaway stores) and service points (i.e., customer locations) are distributed in different locations. Therefore, it is difficult to optimize order selection and delivery path planning for a deliveryman to maximize his revenue.

This paper is a substantial extension of a conference paper [40]. The conference paper proposed a hybrid optimization algorithm for a basic problem of takeaway order selection and delivery path planning. The basic problem assumed that the ready time of each order is exactly known in advance; however, in practice, the actual order ready time often deviates widely from the expected order ready time, which can badly affect the work proficiency of deliverymen. Moreover, the basic problem did not consider customers' scoring on



**Fig. 3** A typical takeaway order processing workflow

orders, which, in real-world takeaway delivery platforms, is an important factor influencing the revenue of deliverymen. In addition, the basic problem did not limit the delivery path length, which may cause infeasible solutions. In this paper, we add the limit of maximum delivery path length to the problem, and use a machine-learning approach to estimate both the order ready time and customer satisfaction level based on the historical habit data of takeaway stores and customers, so as to evaluate delivery solutions in a more accurate way. For the extended problem, we adapt the hybrid optimization algorithm to optimize the revenue of the deliveryman in a more effective manner, and conduct more extensive experiments to validate the performance of the proposed method. The main contributions of this paper can be summarized as follows:

- We present a problem of takeaway order selection and delivery path planning for deliverymen, which utilizes store and customer habit data to better estimate order ready time and customer satisfaction level;
- We propose a hybrid evolutionary algorithm to efficiently solve the problem;
- We demonstrate the performance of the proposed method on test instances constructed based on real-world data of online takeaway platforms.

In the rest of the paper, we first introduce related work in the literature, and then present the problem of takeaway order selection and delivery path planning for deliverymen; next, we propose the machine-learning approach for estimating the problem parameters and the hybrid evolutionary algo-

rithm for solving the problem, and then validate the proposed method in computational experiment; finally, we conclude with a discussion.

## Related work

With the rapid growth of electronic commerce, vehicle routing problems (VRPs) for scheduling vehicles to deliver goods to a given number of service points (customers) have been extensively studied in the literature [13]. Planning deliverymen's paths passing through a set of pickup points and drop-off points subject to delivery time limits can be categorized as a special class of VRPs, known as the pickup and delivery problem with time windows (PDPTW) [12]. For these *NP*-hard problems, traditional mathematical programming methods are only effective on small-size problem instances; therefore, metaheuristic and evolutionary algorithms, such as genetic algorithms (GAs) and particle swarm optimization (PSO), are widely used to find near-optimal solutions for medium- and large-size problem instances within an acceptable solution time [1,4,6].

In addition to common features of VRP and PDPTW, takeaway delivery path planning has some special features. First, as meal is perishable and customers are often waiting anxiously, takeaway orders are typically expected to be delivered within a short time (an hour or even much less) and within minutes of the food becoming ready. Hsu et al. [17] presented a VRP with time windows (VRPTW) for delivering perishable food from a distribution center, the objective of which considers not only the costs for dispatching vehicles, but also those of transportation, inventory, energy and penalty costs for violating time windows. Huang et al. [18] applied an ant colony optimization (ACO) algorithm to plan the delivery route to minimize the total time for takeaway distributions. Reyes et al. [32] formalized a meal delivery routing problem to model the essential structure of dynamic delivery systems, and developed an algorithm based on rolling-horizon repeated matching to solve courier assignment and capacity management. Gao and Jiang [14] applied a firework algorithm [54] to optimize takeaway delivery paths in the condition of safety. Yu and Luo [43] studied an online PDPTW with single pickup point for routing a deliveryman with a constant capacity to serve requests released over time so as to minimize the total latency. They proved the lower bound of this problem for various capacities of the deliveryman, and presented online wait-and-return and wait-and-ignore online algorithms for a half line case. Yildiz and Savelsbergh [42] presented a meal delivery routing problem that assumes perfect information about order arrivals; they proposed a simultaneous column- and row-generation method to solve the problem. Liao et al. [23] presented a green meal delivery routing problem with the objectives

to simultaneously maximize customer satisfaction and rider balance utilization and minimize carbon footprint; they proposed an algorithm based on nondominated Sorting GA [11] and adaptive large neighborhood search for the problem. Liu and Liu [26] presented an integrated production and distribution problem with a single machine, multiple customers, and homogeneous vehicle; they solved this problem using an improved large neighborhood search algorithm. Shan et al. [29] proposed a deep reinforcement learning approach combined with Dijkstra's algorithm for food delivery route planning, which can provide accurate navigation when road network information is unknown. Ulmer et al. [39] studied a stochastic PDPTW for delivering food from a set of restaurants to ordering customers. They presented an anticipatory customer assignment policy, which is able to improve service significantly for all stakeholders. To solve an integrated problem of production-inventory-routing of perishable goods with transshipment and uncertain demand, Liu et al. [28] presented an algorithm that begins with an initial solution and then iteratively improves it using two local search strategies including inserting the best and removing the worst solutions. In [30], Liu considered on-demand meal delivery service using drones, and proposed a progressive algorithm for drone dispatch and order delivery in a dynamic, real-time operational environment. When addressing a stochastic online route-planning problem, Zheng et al. [48] proposed an end-to-end deep-learning model for finding optimal routes in milliseconds by learning policy from training data.

Second, orders are not available for pickup at the beginning of the planning period, which was considered by Liu et al. [25] in the capacitated VRP with order available time and solved using a tabu search algorithm. In [27], the authors proposed a hybrid harmony search and tabu search algorithm for the problem. Li et al. [22] studied a similar VRP with order release time, where a vehicle often needs multiple trips due to the relatively short delivery distance. They proposed an adaptive large neighborhood search algorithm combined with a labeling procedure for the problem.

Third, one customer may order food multiple times from a store or from multiple stores. Consolidating orders of the same customer can reduce the delivery times and distance. However, multiple deliveries to the same customer cannot be completely removed. Zhang et al. [46] presented an integer programming model of order consolidation aiming to reduce the number of trips, while achieving a tradeoff between splitting and consolidating orders; they proposed a three-phase heuristic algorithm to solve the problem, and demonstrated the superiority of the order consolidation approach over the first-in-first-out approach. To solve a time-critical third-party logistics problem with order consolidation and transshipment point selection, Salhi et al. [33] proposed an effective metaheuristic based on the greedy randomized adaptive search procedure. Soman and Patil [35] studied a heterogeneous

VRP with release and due dates in the presence of order consolidation and warehousing capacity limits; they proposed a scatter search method with strategic oscillation, which is able to solve large-size instances. Ji et al. [20] proposed a method for grouping food delivery tasks to improve food delivery efficiency, using heuristics consisting of a greedy algorithm and a replacement algorithm.

Most existing studies either integrate order assignment and delivery path routing, or assume that the orders have been assigned and hence focus on path routing. Nevertheless, in takeaway delivery systems, deliverymen are not simply passive entities; instead, they create their own “organic algorithms” to manage, and in some cases, even subvert the system [37]. For example, in popular food delivery platforms such as Baidu Deliveries, Eleme, and Meituan, deliverymen pay close attention to “grab orders” to improve their revenue. However, studies on deliverymen’s proactive strategies for takeaway order selection are relatively few. İç et al. [19] studied an order selection problem for a bakery firm, for which they used a fuzzy TOPSIS method to obtain the order ranking incorporated in the knapsack problem to determine the lot size and which orders to select. Ma et al. [31] considered a combined order selection and VRP for perishable product delivery, for which they proposed a hybrid ACO and local search method. Zhang and Liu [44] formulated a takeaway distribution problem as a bi-objective, mixed integer programming model; they proposed a two-stage solution strategy based on human–computer interaction to solve the problem. Nevertheless, to the best of our knowledge, there is no study on methods integrated order selection and path planning for takeaway deliverymen, the revenue of which not only consists of the basic delivery fee of each order overdue penalty, but also is subject to reward/penalty for large/small number of orders, and high customer scoring reward.

## Problem formulation

### Basic inputs

The consider problem aims to make an optimal decision of order selection and delivery path planning for a deliveryman. There are a set  $O$  of  $n$  candidate orders. For each order  $o \in O$ , the pickup point (store) is denoted by  $p_o$ , the expected order ready time is denoted by  $r_o$ , and the corresponding service point (customer) is denoted by  $s_o$ . For convenience, we use  $p_0$  to denote the initial location of the deliveryman, and let  $P = (\cup_{o \in O} p_o) \cup (\cup_{o \in O} s_o) \cup \{p_0\}$  be the set of all pickup points, service points, and the initial location of the deliveryman. The travel time between each pair of points  $i$  and  $j$  is denoted by  $\Delta t(i, j)$  ( $\forall i, j \in P$ ). The vehicle (typically, electronic bicycle) of the deliveryman has a maximum distance; here, we transform the maximum distance to the maximum travel

time  $\widehat{T}$ , which neglects acceleration and deceleration in the path for simplicity.

If an order  $o$  is selected by the deliveryman, the basic delivery fee is  $v_o$ , and it is required to deliver the order to  $p_o$  before the delivery deadline  $\widehat{t}_o$  to earn the delivery fee. However, if the actual delivery time is later than  $\widehat{t}_o$ , an overdue penalty will be posed. In this study, we consider a three-level overdue penalization rule that is employed by most food delivery platforms in China as follows:

- If the overdue time is shorter than 15 min, the basic delivery fee will be deducted by a percent  $e_1$ ;
- If the overdue time is between [15, 30] min, the basic delivery fee will be deducted by a percent  $e_2$ ;
- If the overdue time is longer than 30 min, no delivery fee will be paid, and an additional penalty fee which is a percent  $e_3$  of the basic delivery fee will be deducted.

The delivery platform also encourages deliverymen to take more orders: if the number of orders completed by a deliveryman in a given period (e.g., per week) exceeds a threshold, an additional reward will be granted; on the contrary, if the number is below a lower limit, his base salary will be deducted. To reflect this effect on the deliveryman’s revenue per unit time, in this problem, we set a lower limit  $\underline{n}_a$  and two reward thresholds  $n_a^\dagger$  and  $n_a^\ddagger$  ( $\underline{n}_a < n_a^\dagger < n_a^\ddagger$ ) on the number  $n_p$  of orders per hour completed by the deliveryman, and use the following rule according to the reward/penalization levels in popular platforms and number conversion based on average working hours:

- If  $n_p$  is less than the lower limit  $\underline{n}_a$ , there is an additional penalty of  $\epsilon_1(\underline{n}_a - n_p)$  yuan;
- If  $n_p$  is between  $[n_a^\dagger, n_a^\ddagger)$ , there is an additional reward of  $\epsilon_2$  yuan per order;
- If  $n_p$  reaches or exceeds  $n_a^\ddagger$ , there is an additional reward of  $\epsilon_3$  yuan per order.

Moreover, when the order is completed, if the customer gives a five-star (highest) score on the order delivery, the deliveryman will receive an award fee of  $e$ .

**Note 1** The possibility of negative scoring and the corresponding penalty are not considered in the revenue, to avoid that orders from low-scoring customers would not be selected by any deliverymen.

**Note 2** The above rules and parameters can be adjusted and tailored to different delivery platforms, which will not have side effect on our formulation and solution method.

**Table 1** The inputs of the problem

Variable	Description
$n$	Number of the candidate orders
$O$	Set of candidate orders
$p_o$	Pickup point of order $o$ ( $\forall o \in O$ )
$s_o$	Service point of order $o$ ( $\forall o \in O$ )
$v_o$	Basic delivery fee of order $o$ ( $\forall o \in O$ )
$\widehat{t}_o$	Delivery deadline of order $o$ ( $\forall o \in O$ )
$r_o$	Expected ready time of order $o$ ( $\forall o \in O$ )
$p_0$	Initial location of the deliveryman
$P$	Set of all involved points
$\Delta t(i, j)$	Travel time from point $i$ to point $j$ ( $\forall i, j \in P$ )
$\widehat{T}$	Maximum travel time of the delivery path
$e_1$	First-level overtime penalty
$e_2$	Second-level overtime penalty
$e_3$	Third-level overtime penalty
$n_a$	Lower limit of the number of orders per hour
$n_a^\dagger$	First threshold of the number of orders per hour
$n_a^\ddagger$	Second threshold of the number of orders per hour
$\epsilon_1$	Additional penalty on the number of orders per hour
$\epsilon_2$	First-level award on the number of orders per hour
$\epsilon_3$	Second-level award on the number of orders per hour
$e$	Additional award per five-star score
$\widehat{r}(o)$	Ready time of order $o$ calculated using machine learning
$\rho(o)$	Probability of five-star scoring calculated using machine learning

### Uncertain factors

In particular, in this problem, we consider two uncertain factors. The first is that, at the beginning of the planning period, the expected ready time  $r_o$  of each order is estimated and given by the store, but the estimation is not always accurate. In many cases, the actual ready time, denoted by  $\widehat{r}(o)$ , is later, which will postpone all subsequent orders, and therefore, has a significant side effect on the delivery time. We employ a data-driven, machine-learning approach described in the next section to estimate  $\widehat{r}(o)$  based on  $r_o$  and the historical habit data of the store.

The second is about customer satisfaction level. Although the satisfaction level generally depends on the delivery time [7], some customers are more likely to give five-star scores, while others are not. We also employ a machine-learning approach described in the next section to estimate the probability  $\rho(o)$  that the customer of the order will give a five-star score on the delivery (under the condition that the delivery time is not overdue) based on the delivery time and the historical habit data of the customer.

Table 1 lists the above input variables of the problem.

### Decision variables

The deliveryman needs to select a subset  $O_x$  of orders from the candidate order set  $O$ , and then determine a path to deliver the selected orders. Therefore, the decision variables of the problem can be represented by the following three parts:

- An  $n$ -dimensional vector  $\mathbf{x} = \{x_1, x_2, \dots, x_n\}$ , where  $x_k = 1$  denotes that  $o_k$  is selected and  $x_k = 0$  otherwise ( $1 \leq k \leq n$ ); then the subset of selected orders is  $O_x = \{o_k | o_k \in O \wedge x_k = 1\}$ .
- A sequence  $\mathbf{y} = \{y_1, y_2, \dots, y_{l(\mathbf{y})}\}$  of the set of all pickup points and service points of the orders in  $O_x$ , where  $l(\mathbf{y})$  denotes the length of  $\mathbf{y}$ . In other words,  $\mathbf{y}$  represents the delivery path of the deliveryman. For each  $y_j$  in  $\mathbf{y}$ , if  $y_j$  is a pickup point, we let  $O(y_j) = \{o | o \in O_x \wedge p_o = y_j\}$  be the set of orders from store  $y_j$ , and suppose that the orders in  $O(y_j)$  are sorted in increasing order of ready time; if  $y_j$  is a service point, we let  $O'(y_j) = \{o | o \in O_x \wedge s_o = y_j\}$  be the set of orders for customer  $y_j$ .
- For each pickup point  $y_j$  in  $\mathbf{y}$ , an integer  $z(y_j)$  that denotes the deliveryman’s decision on how many orders the deliveryman will pick up from  $y_j$  at this time. That is,



if  $|O(y_j)| = 1$ , then  $z(y_j)$  is 1; else,  $z(y_j)$  is an integer in  $[1, |O(y_j)|]$ , indicating that the deliveryman will pick up the first  $z(y_j)$  of these  $|O(y_j)|$  orders and then leave (and will go back for the remaining orders if exist).

**Note 3** As the deliveryman may visit a store or a customer more than once if the store or customer is related to multiple orders, the length of permutation  $\mathbf{y}$  is variable. Anyway, the length  $l(\mathbf{y})$  is at most  $2|O_x|$ . For simplicity, we never place the initial location 0 of the deliveryman in the permutation.

### Calculation of delivery time and revenue

The actual delivery time of each order depends on the order ready time, pickup time, and delivery path  $\mathbf{y}$ . Obviously, the first point  $y_1$  in  $\mathbf{y}$  must be a pickup point, and the time at which the deliveryman arrives at  $y_1$  is

$$t(y_1) = \Delta t(p_0, y_1). \tag{1}$$

At the first pickup point  $y_1$ , the deliveryman’s decision is to pick up the first  $z(y_1)$  orders in  $O(y_1)$  and then leaves  $y_1$ . Let  $O[z]$  denotes the  $z$ -th element in  $O$ ; the time at which the deliveryman leaves  $y_1$  is

$$t'(y_1) = \max(t(y_1), \widehat{r}(O(y_1)[z(y_1)])) \tag{2}$$

Afterward, we remove the first  $z(y_1)$  orders from  $O(y_1)$ :

$$O(y_1) = O(y_1) \setminus \{O(y_1)[1..z(y_1)]\} \tag{3}$$

The times at which the deliveryman arrives at and leaves each subsequent point in  $y_j$  can be iteratively calculated as follows ( $2 \leq j \leq l(\mathbf{y})$ ):

$$t(y_j) = t'(y_{j-1}) + \Delta t(y_{j-1}, y_j) \tag{4}$$

$$t'(y_j) = \begin{cases} t(y_j), & y_j \text{ is a service point} \\ \max(t(y_j), \widehat{r}(O(y_j)[z(y_j)])), & y_j \text{ is a pickup point} \end{cases} \tag{5}$$

When leaving each pickup point  $y_j$ , we remove the first  $z(y_j)$  orders from  $O(y_j)$ :

$$O(y_j) = O(y_j) \setminus \{O(y_j)[1..z(y_j)]\} \tag{6}$$

When arriving each service point  $y_j$ , for each order  $o \in O'(y_j)$ , if the order has been picked up before  $y_j$ , then its delivery time  $d(o)$  is determined:

$$d(o) = t(y_j), \forall o \in O'(y_j) \wedge (\exists j' < j : o \text{ is among the first } z(y_{j'}) \text{ orders in } O(y_{j'})) \tag{7}$$

Therefore, we can calculate the revenue of each order  $o \in O_x$  as follows:

$$f(o) = \begin{cases} v_o + \rho(o)e, & d(o) < \widehat{t}_o \\ (1 - e_1)v_o, & \widehat{t}_o \leq d(o) \leq \widehat{t}_o + 15 \\ (1 - e_2)v_o, & \widehat{t}_o + 15 < d(o) \leq \widehat{t}_o + 30 \\ -e_3v_o, & d(o) > \widehat{t}_o + 30 \end{cases} \tag{8}$$

Here, we specify time in minutes, and hence the number of orders per hour is

$$n_p = 60|O_x|/t(y_{l(x)}) \tag{9}$$

And the additional penalty/reward of a solution is calculated as follows:

$$g(\mathbf{x}, \mathbf{y}, \mathbf{z}) = \begin{cases} -\epsilon_1(n_a - n_p), & n_p < n_a \\ \epsilon_2|O_x|, & n_a^\dagger \leq n_p < n_a^\ddagger \\ \epsilon_3|O_x|, & n_p \geq n_a^\ddagger \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

The problem objective is to maximize the revenue per unit time, i.e., the ratio of the total revenue to the completion time  $t(y_{l(x)})$ :

$$\max F(\mathbf{x}, \mathbf{y}, \mathbf{z}) = \frac{(\sum_{o \in O_x} f(o)) + g(\mathbf{x}, \mathbf{y}, \mathbf{z})}{t(y_{l(x)}) + \epsilon} \tag{11}$$

where  $\epsilon$  is a very small number to avoid division by zero (i.e., if the deliveryman does not select any order, the objective function value should be zero).

### Constraints

We specify the following constraints for the problem:

- For each selected order  $o$ , the delivery path must contain its pickup point  $p_o$  and service point  $s_o$ , and the (first) occurrence of  $p_o$  should be before that of  $s_o$ :

$$p_o \in \mathbf{y} \wedge s_o \in \mathbf{y} \wedge \text{ind}(p_o, \mathbf{y}) < \text{ind}(s_o, \mathbf{y}), \quad \forall o \in O_x \tag{12}$$

where  $\text{ind}(i, \mathbf{y})$  denotes the index of element  $i$  in sequence  $\mathbf{y}$  (if the element occurs multiple time, it returns the first index).

- The decision  $z(y_j)$  at each pickup point is not larger than the cardinality of  $O(y_j)$ :

$$1 \leq z(y_j) \leq |O(y_j)|, \quad \forall \text{ pickup point } y_j \text{ in } \mathbf{y} \tag{13}$$

- The total travel time cannot exceed the maximum travel time  $\widehat{T}$ :

$$\Delta t(p_0, y_1) + \sum_{j=1}^{l(y)-1} \Delta t(y_j, y_{j+1}) \leq \widehat{T} \tag{14}$$

### Data-driven machine learning for estimating order ready time and customer satisfaction level

As aforementioned, for the considered problem, we identify two uncertain factors, i.e., the actual order ready time and customer satisfaction level, which are regarded as main challenges in order selection and delivery routing [9]. We employ a data-driven, machine-learning approach to provide more accurate predictions to address these challenges based on historical habit data of stores and customers, i.e., the overdue records of stores and five-star scoring records of customers.

To predict the actual order ready time, we consider the following influence factors of the corresponding store:

- Expected ready time  $r_o$  of the current order given by the store;
- Number of orders whose ready times are overdue in the recent month;
- Percentage of orders whose ready times are overdue in the recent month;
- Maximum, minimum, median, and standard deviations of the overdue time of the overdue orders in the recent month;
- Number of orders whose ready times are overdue in the recent 3 days;
- Percentage of orders whose ready times are overdue in the recent 3 days;
- Maximum, minimum, median, and standard deviations of the overdue time of the overdue orders in the recent 3 days;
- Number of orders that are accepted by the store and to be delivered in the next hour.

We construct a three-layer, feed-forward artificial neural network (ANN) to calculate the actual order ready time based on the above 14 inputs. The training data is limited to the recent 1 month, as the habit of a takeaway store often changes.

To predict the probability of five-star scoring on the order, we consider the following influence factors of the corresponding customer:

- Calculated delivery time  $d(o)$  of the current order;
- Number of orders placed by the customer in the recent 3 months, recent week, and recent day;

- Percentage of orders receiving five-star scores to all orders that are not overdue in the recent 3 months, recent week, and recent day;
- Number of orders received or to be received by the customer 1 h before and after.

Similarly, we construct an ANN to calculate the probability based on the above eight inputs. The training data are limited to the recent 3 months.

### A hybrid evolutionary algorithm for the problem

Due to the complex combinatorial nature of the considered problem, we propose a hybrid evolutionary algorithm, which consists of a main procedure for optimizing the solution to the main order selection problem and a subprocedure for optimizing path planning for each main solution. The flowchart of the algorithm can be described by the following steps (as illustrated in Fig. 4):

- (1) Randomly initialize a population of order selection solutions;
- (2) For each order selection solution  $x$  in the population do:

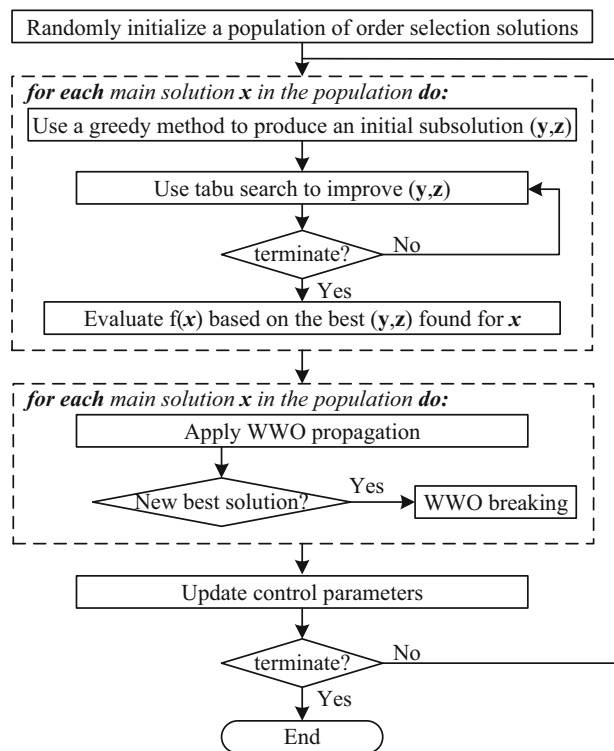


Fig. 4 The flowchart of the hybrid evolutionary algorithm

- (2.1) Use a greedy method to generate an initial path  $\mathbf{y}$  together with pickup decisions  $\mathbf{z}$ ;
  - (2.2) Use the subprocedure to iteratively improve the path and decisions;
  - (2.3) Evaluate the fitness of  $\mathbf{x}$  based on the best path and decisions found so far;
- (3) Use the main procedure to evolve the order selection solutions;
  - (4) Repeat steps (2) and (3) until the stopping condition is satisfied.

For the subprocedure for optimizing the delivery path and the pickup decisions, we propose a heuristic method based on tabu search [15,16], which is much faster than those population-based evolutionary algorithms, as the subprocedure will be invoked as many times as the evaluations of main solutions. For the main procedure for optimizing solutions to the main order selection problem, we have tested a set of popular evolutionary algorithms, and found that the WWO metaheuristic [50] exhibits performance advantages over other popular evolutionary algorithms on the test instances. We describe the tabu search subprocedure (including the greedy initialization method) and the main procedure in details in the following two subsections, respectively.

### Tabu search for path planning

Given a main order selection solution  $\mathbf{x}$ , we first use the greedy method to produce an initial subsolution  $(\mathbf{y}, \mathbf{z})$  of path with pickup decisions as follows:

- (1) Initialize an empty sequence for  $\mathbf{y}$  and an empty set  $\Omega$  of picked up orders;
- (2) Choose the pickup point  $y$  closest to the deliveryman and add it to the sequence, and calculate the arrival time  $t(y)$ ;
- (3) Set the pickup decision  $z(y)$  as follows:
  - (3.1) If  $|O(y)| = 1$ , i.e.,  $O(y)$  has only one order denoted by  $o_y$ , then set  $z(y) = 1$ ;
  - (3.2) Else, find the last order  $o_y^\dagger$  whose ready time is not later than  $t(y)$ , let  $j_y^\dagger$  be the index of  $o_y^\dagger$  in  $O(y)$ , and set  $z(y)$  to a random integer in  $[j_y^\dagger, |O(y)|]$ ;
  - (3.3) Remove the first  $z(y)$  orders from  $O(y)$  to  $\Omega$ , and set  $t'(y) = \max(t(y), \widehat{r}(O(y)[z(y)]))$ ; if  $O(y)$  becomes empty, remove  $y$  from the candidate pickup points;
- (4) From all candidate pickup points and those service points that are related to at least one order in  $\Omega$ , choose the point  $y$  closest to the deliveryman and add it to the sequence, calculate the arrival time  $t(y)$ , and
  - (4.1) If  $y$  is a pickup point, go to Step (3);

- (4.2) If  $y$  is a service point, for all orders  $o \in \Omega$  and  $s_o = y$ , set  $d(o) = t(y)$ , remove these orders from  $O'(y)$ , and set  $t'(y) = t(y)$ ; if  $O'(y)$  becomes empty, remove  $y$  from the candidate service points;
- (5) Repeat Step (4) until  $\Omega = O_{\mathbf{x}}$ .

From the initial  $(\mathbf{y}, \mathbf{z})$ , we use tabu search that iteratively searches the neighborhood of the subsolution and goes to the best neighboring subsolution that is better than the current one or is not tabued. As the subsolution consists of two parts, the path and pickup decisions, we consider two types of neighborhood search. The first type conducts point swapping operations on the path  $\mathbf{y}$ . Considering the problem constraints, we design the following four swapping operations:

- (a) Swap two adjacent pickup points  $p_1$  and  $p_2$ , as illustrated by Fig. 5a.
- (b) Swap a service point  $s$  and a subsequent pickup point  $p$ ; in particular, if  $s$  has occurred again after  $p$  as a service point of the order from  $p$  but not as a service point of that from any other pickup point after  $p$ , then the next occurrence of  $s$  will be removed, as illustrated by Fig. 5b.
- (c) Swap a pickup point  $p$  and a subsequent service point  $s$ , if  $s$  is a service point of the order from another pickup point before  $p$ ; however, if  $s$  is also a service point of the order from  $p$ ,  $s$  will be reinserted after  $p$ , as illustrated by Fig. 5c.
- (d) Swap two service points  $s_1$  and  $s_2$ , where there is no any pickup points between them, as illustrated by Fig. 5d.

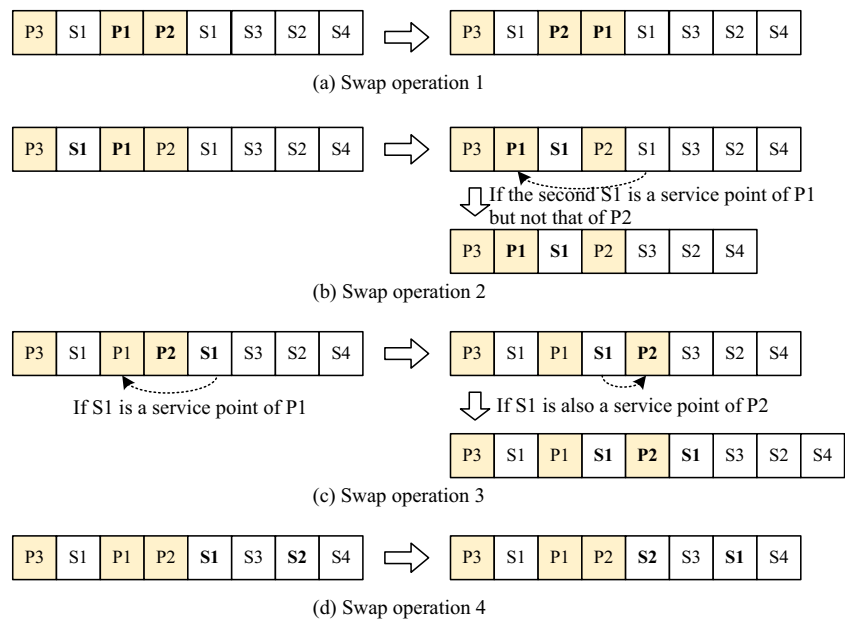
In addition, if the swapping operations involves a pickup point, Step (3) of the greedy initialization method is employed to reset the pickup decision on the point, and each service point related to a decreased decision is moved to a random position after the corresponding order is picked up.

The second type of neighborhood search modifies pickup decisions  $\mathbf{z}$  by randomly choosing a pickup point  $y$  satisfying  $|O(y)| > j_y^\dagger$  and changing  $z(y)$  to another random value in  $[j_y^\dagger, |O(y)|]$ ; this can be divided into two case:

- (a)  $z(y)$  is increased; in this case, if  $z(y) = |O(y)|$ , the later occurrence(s) of  $y$  in  $\mathbf{y}$  will be removed.
- (b)  $z(y)$  is decreased; in this case, for each later occurrence of  $y'$  in  $\mathbf{y}$ , Step 3) of the greedy initialization method is employed to reset the corresponding pickup decision, and each service point related to a decreased decision is moved to a random position after the corresponding order is picked up.



**Fig. 5** Illustration of the four specific swapping operations used in tabu search

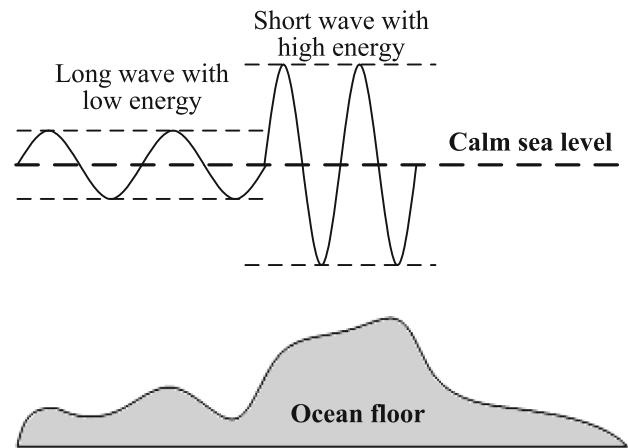


Algorithm 1 presents the pseudo-code of tabu search, where  $t_{max}$  is the maximum number of iterations,  $n_b$  is the neighborhood size (i.e., the number of neighbors generated at each generation),  $tlen$  is the maximum tabu length, and  $rnd()$  produces a random number in  $[0,1]$ .

**Algorithm 1:** Tabu search algorithm for path planning optimization.

```

1 Initialize an empty tabu list;
2 Use the greedy method to produce an initial path (y, z);
3 Let t = 0 and the best known subsolution (y*, z*) = (y, z);
4 while t < t_max do
5   for k = 1 to n_b do
6     if rnd() < 0.5 then
7       Generate a neighbor of (y, z) by randomly
       choosing and performing a neighborhood search
       on the path y;
8     else
9       Generate a neighbor of (y, z) by randomly
       performing a neighborhood search on the pickup
       decisions;
10    Let (y', z') to the best one among the n_b neighbors that are
        better than or the move from (y, z) to the neighbor is not
        tabued;
11    Add the move from (y, z) to (y', z') to the tabu list;
12    if the tabu list length exceeds the capacity tlen then
13      Remove the first element from the tabu list;
14    if (y', z') is better than (y, z) then
15      Set (y, z) ← (y', z');
16      if (y, z) is better than (y*, z*) then
17        Set (y*, z*) ← (y, z);
18 return (y*, z*).
    
```



**Fig. 6** Wave propagation in WWO [50]

**Water wave optimization for order selection**

For order selection optimization, we propose an evolutionary algorithm based on the WWO metaheuristic [50] that takes inspiration from shallow water wave models for solving optimization problems. In particular, WWO has demonstrated superior performance on a variety of selection problems that have same or similar structure of solution space [3,8,24,41,49,53]. In WWO, each solution is analogous to a wave and is assigned with a wavelength inversely proportional to the solution fitness. The key principle of WWO is that the higher (lower) the solution fitness, the smaller (larger) the wavelength, and the smaller (larger) range the solution explores (as illustrated in Fig. 6), which results in a good balance of global search and local search.

WWO starts by initializing a population of solutions, which are then evolved by three operators named propagation, refraction, and breaking. As the original WWO is proposed for continuous optimization, here we need to adapt the algorithm to evolve solutions in the discrete search space [52]. First, we adapt the propagation to perform a number of local search steps on each solution  $\mathbf{x}$ , where each local search step changes a random dimension  $x_k$  from 0 to 1 or from 1 to 0. The maximum number of local search steps is controlled by the wavelength  $\lambda(\mathbf{x})$ , which is an integer calculated as

$$\lambda(\mathbf{x}) = \lceil n^{(f(\mathbf{x}) - f_{\min}) + \epsilon} / (f_{\max} - f_{\min} + \epsilon) \rceil \quad (15)$$

where  $\lceil \cdot \rceil$  denotes rounding to the nearest integer, and  $f_{\max}$  and  $f_{\min}$  are the maximum and minimum fitness among the population, respectively.

After propagation, if the new solution is better than the original one, it will replace the original one in the population.

Second, we adapt the breaking operator on each newly found best solution  $\mathbf{x}^*$  by generating  $n^*$  one-step neighboring solutions around  $\mathbf{x}^*$ . Here we introduce an adaptive method for controlling the number  $n^*$  of neighboring solutions as follows:

$$n^* = \text{rnd\_int} \left( 1, \hat{n} \frac{f_{\text{old}}^* + \epsilon}{f(\mathbf{x}^*) + \epsilon} \right) \quad (16)$$

where  $\hat{n}$  is a control parameter, and  $f_{\text{old}}^*$  is the objective function value of the old best solution. In this way, the more improvement of the new best over the old one, the larger number of neighboring solutions exploited.

Following the work of simplified WWO [55], we replace the refraction operator with a population reduction strategy in order to accelerate convergence. The strategy iteratively decreases the population size  $NP$  from an upper limit  $NP_{\max}$  to a lower limit  $NP_{\min}$  as follows:

$$NP_g = NP_{\max} - (NP_{\max} - NP_{\min}) \frac{g}{g_{\max}} \quad (17)$$

where  $g$  is the current number of generations (or function evaluations), and  $g_{\max}$  is the maximum allowable number of the generations (or function evaluations). Whenever the size is decreased by one, the worst solution in the population is removed.

Algorithm 2 presents the pseudo-code of the WWO algorithm with adaptive breaking (denoted by WWO-AB) for the main problem of order selection.

---

### Algorithm 2: WWO algorithm with adaptive breaking for order selection optimization.

---

```

1 Randomly initialize a population of  $NP$  solutions to the main
  problem;
2 Let  $\mathbf{x}^*$  be the best among the solutions;
3 while the stopping condition is not satisfied do
4   Calculate the solution wavelengths according to (15);
5   foreach  $\mathbf{x}$  in the population do
6     Let  $K = \text{rnd\_int}(1, \lambda(\mathbf{x}))$ ;
7     for  $k = 1$  to  $K$  do
8       randomly choose and reverse a dimension of  $\mathbf{x}$ ;
9     if the new  $\mathbf{x}'$  is better than  $\mathbf{x}$  then
10      Replace  $\mathbf{x}$  with  $\mathbf{x}'$  in the population;
11      if  $\mathbf{x}$  is better than  $\mathbf{x}^*$  then
12        Set  $\mathbf{x}^* \leftarrow \mathbf{x}$ ;
13        Calculate  $n^*$  according to Eq. (16);
14        for  $k = 1$  to  $n^*$  do
15          Generate a one-step neighbor by
16            reversing a random dimension of  $\mathbf{x}^*$ ;
17            if the neighbor is better than  $\mathbf{x}^*$  then
18              Set  $\mathbf{x}^*$  to the neighbor;
18      Update the population size according to (17);
19      if the population size is decreased by one then
20        Remove the worst solution from the population;
21 return  $\mathbf{x}^*$ .

```

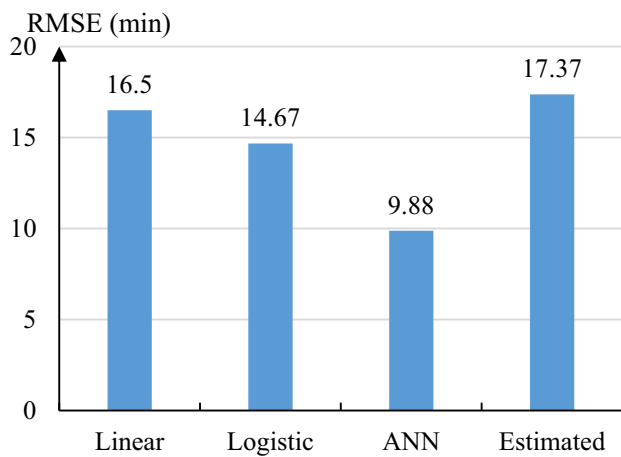
---

## Computational experiments

### Experimental results of machine learning

We train the ANNs to predict the two uncertain factors for each order based on historical data of two popular food delivery applications. For predicting order ready time, we use a dataset of 1330 samples related to 121 takeaway stores. For predicting five-star scoring probability, we use a dataset of 1750 samples related to 204 customers. We use a fivefold cross-validation, that is, we partition each dataset into five equal-size pieces and run validation five times, each using four pieces as the training set and the remaining piece as the test set.

We also employ WWO to tune the ANN parameters [56], and compare the performance of ANN with linear regression and logistic regression. Figure 7 presents the root mean squared errors (RMSE) of the three models as well as the RMSE of the order ready time estimated by the store. The results show that the average deviation of the order ready time estimated by the store to the actual order ready time is about 15.37 min, which will not only delay the delivery of current order, but also have a knock-on effect on all remaining orders. The three machine-learning models utilize historical data to predict the order ready time. However, the error of linear regression model is only slightly lower than that of the manual estimation. The logistic regression model is more



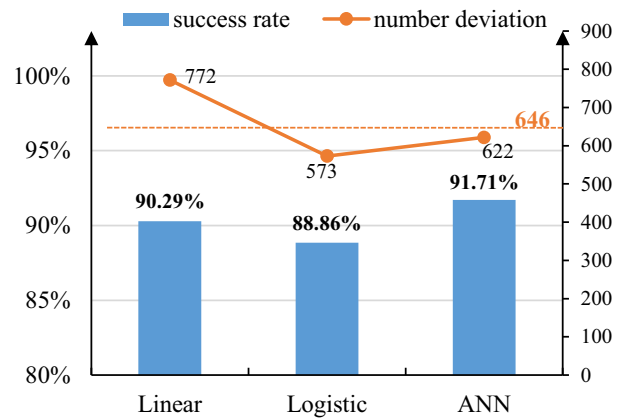
**Fig. 7** RMSE of order ready times predicted by the three machine-learning models and the stores

accurate than the linear one, but its average deviation is still more than 11 min. Compared to the two regression models, the ANN achieves a significant lower error of 7.88 min, which can effectively reduce the side effect on the delivery plan.

For five-star scoring probability prediction, we use two metrics. The first is the success rate, i.e., the percentage of successful predictions, where a probability larger than 0.5 for a five-star scoring or a probability smaller than 0.5 for a non-five-star scoring is considered successful. The second is the deviation of the sum of probabilities to the actual number of five-star scoring. Figure 8 presents the results, where the orange line denotes the actual number (646) of five-star scoring of the three models as well as the RMSE of the order ready times estimated by the stores. The results show that ANN achieves the highest success rate, while the success rates of the two regression models are not much lower. However, the differences among the deviations of the number of five-star scoring obtained by the three models are relatively big. The linear regression model overestimates 126 five-star scoring, and the logistic regression model underestimates 73; in comparison, ANN only underestimates 24, and such a small deviation will make the calculation of the revenue of the deliveryman (i.e., objective function of the problem) much more accurate.

**Experimental results of evolutionary optimization**

To test the performance of solving the takeaway order selection and delivery path planning problem, we construct a test set of 11 instances, which are generated based on historical data of two popular food delivery applications. Table 2 describes numbers of orders and points of each instance, which represent the size/difficulty of the instance. Some other important parameters of the instances are set as  $e_1 = 0.3$ ,



**Fig. 8** Success rate and deviation of the number of five-star scoring of the three machine-learning models

**Table 2** Numbers of orders and points of the problem instances

No. instance	$n$	$n_p$	$n_s$
#1	25	3	8
#2	25	6	17
#3	25	9	26
#4	50	9	26
#5	50	12	35
#6	50	20	59
#7	75	20	59
#8	75	27	81
#9	100	20	59
#10	100	32	95
#11	100	36	121

$n$ : number of the candidate orders;  $n_p$ : number of the pickup points;  $n_s$ : number of the service points; MNFEs: maximum number of fitness evaluations

$e_2 = 0.5, e_3 = 0.7, e = 1, \epsilon_1 = 0.5, \epsilon_2 = 0.5, \epsilon_3 = 1, n_a = 5, n_a^\dagger = 12, \text{ and } n_a^\ddagger = 16.$

To validate the performance of proposed WWO-AB algorithm, we compare it with the following eight popular metaheuristic evolutionary algorithms for subset selection optimization:

- GA [10];
- ACO [21];
- PSO [5];
- Differential evolution (DE) [2];
- Biogeography-based optimization (BBO) [34,47];
- Ecogeography-based optimization (EBO) [51];
- Artificial algae algorithm (AAA) [45];
- Basic WWO, where the number of neighboring solutions generated by a breaking operation is a random value between 1 and a fixed threshold [41].

**Table 3** Comparative results on the test instances

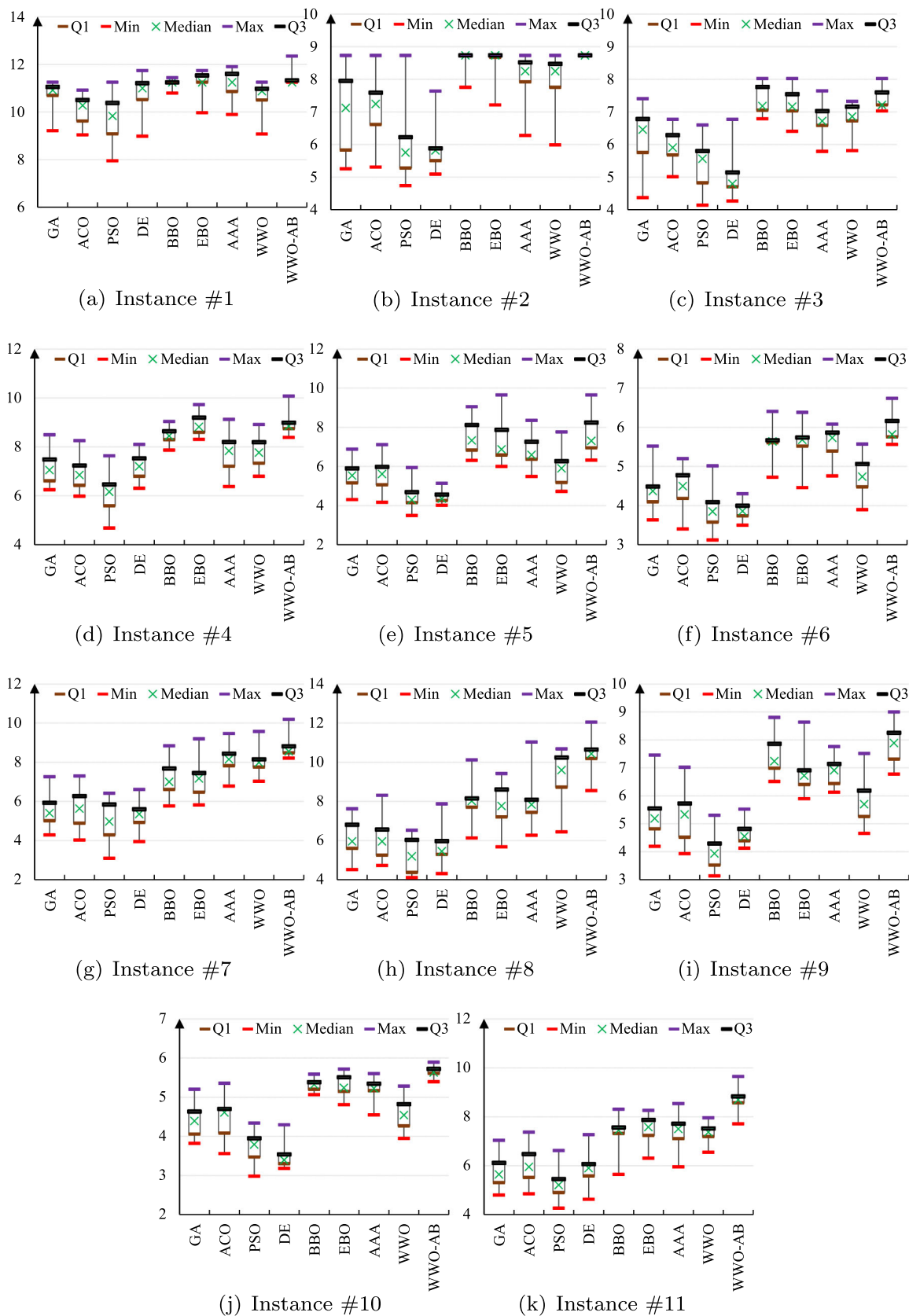
#	Metric	GA	ACO	PSO	DE	BBO	EBO	AAA	WVO	WVO-AB
1	med	<sup>†</sup> 10.91	<sup>†</sup> 10.27	<sup>†</sup> 9.84	<sup>†</sup> 11.00	<sup>†</sup> <b>11.25</b>	<b>11.25</b>	<b>11.25</b>	<sup>†</sup> 10.87	<b>11.25</b>
	std	0.51	0.82	0.92	0.77	0.10	0.41	0.79	0.51	0.20
2	med	<sup>†</sup> 7.12	<sup>†</sup> 7.24	<sup>†</sup> 5.75	<sup>†</sup> 5.82	<b>8.73</b>	<sup>†</sup> <b>8.73</b>	<sup>†</sup> 8.25	<sup>†</sup> 8.25	<b>8.73</b>
	std	1.15	1.01	1.08	0.51	0.24	0.42	0.67	0.71	0.00
3	med	<sup>†</sup> 6.46	<sup>†</sup> 5.91	<sup>†</sup> 5.56	<sup>†</sup> 4.80	7.18	7.16	<sup>†</sup> 6.72	<sup>†</sup> 6.86	<b>7.21</b>
	std	0.77	0.64	0.67	0.67	0.38	0.46	0.49	0.44	0.33
4	med	<sup>†</sup> 7.05	<sup>†</sup> 6.85	<sup>†</sup> 6.16	<sup>†</sup> 7.20	<sup>†</sup> 8.45	8.82	<sup>†</sup> 7.83	<sup>†</sup> 7.77	<b>8.85</b>
	std	0.63	0.59	0.67	0.48	0.29	0.39	0.56	0.53	0.35
5	med	<sup>†</sup> 5.53	<sup>†</sup> 5.62	<sup>†</sup> 4.29	<sup>†</sup> 4.35	<b>7.33</b>	<sup>†</sup> 6.88	<sup>†</sup> 6.59	<sup>†</sup> 5.90	7.31
	std	0.57	0.60	0.55	0.26	0.82	1.03	0.70	0.73	0.99
6	med	<sup>†</sup> 4.37	<sup>†</sup> 4.50	<sup>†</sup> 3.85	<sup>†</sup> 3.86	<sup>†</sup> 5.64	<sup>†</sup> 5.68	5.73	<sup>†</sup> 4.74	<b>5.82</b>
	std	0.41	0.45	0.42	0.21	0.29	0.35	0.35	0.41	0.32
7	med	<sup>†</sup> 5.40	<sup>†</sup> 5.63	<sup>†</sup> 4.97	<sup>†</sup> 5.35	<sup>†</sup> 7.00	<sup>†</sup> 7.17	<sup>†</sup> 8.15	<sup>†</sup> 7.99	<b>8.56</b>
	std	0.68	0.85	0.96	0.49	0.72	0.74	0.52	0.36	0.30
8	med	<sup>†</sup> 5.96	<sup>†</sup> 5.96	<sup>†</sup> 5.19	<sup>†</sup> 5.45	<sup>†</sup> 8.02	<sup>†</sup> 7.77	<sup>†</sup> 7.04	<sup>†</sup> 9.60	<b>10.43</b>
	std	0.78	0.97	0.90	0.48	0.41	0.84	0.51	0.81	0.36
9	med	<sup>†</sup> 5.19	<sup>†</sup> 5.33	<sup>†</sup> 3.93	<sup>†</sup> 4.55	<sup>†</sup> 7.24	<sup>†</sup> 6.73	<sup>†</sup> 6.91	<sup>†</sup> 5.70	<b>7.89</b>
	std	0.70	0.85	0.51	0.34	0.67	0.67	0.65	0.66	0.62
10	med	<sup>†</sup> 4.39	<sup>†</sup> 4.61	<sup>†</sup> 3.79	<sup>†</sup> 3.39	<sup>†</sup> 5.32	<sup>†</sup> 5.24	<sup>†</sup> 5.22	<sup>†</sup> 4.55	<b>5.64</b>
	std	0.44	0.54	0.36	0.31	0.16	0.28	0.30	0.39	0.17
11	med	<sup>†</sup> 5.64	<sup>†</sup> 5.95	<sup>†</sup> 5.21	<sup>†</sup> 5.9	<sup>†</sup> 7.45	<sup>†</sup> 7.58	<sup>†</sup> 7.50	<sup>†</sup> 7.37	<b>8.70</b>
	std	0.46	0.52	0.38	0.40	0.28	0.37	0.45	0.26	0.21

WVO-AB and the eight comparative algorithms invoke the same tabu search procedure given in Algorithm 1 for path planning optimization for each main solution. The control parameters of tabu search are set as  $len = 7$ ,  $n_b = 10$ , and  $t_{max} = 10$  for instances #1 and #2, 20 for #3–#5, and 30 for #6–#11. The control parameters of the nine evolutionary algorithms are first set as suggested in the literature and then tuned on the whole test set. For WVO-AB, the control parameters are set as  $NP_{max} = 30$ ,  $NP_{min} = 6$ , and  $\hat{n} = 12$ . The computational environment is a computer with Intel core i7-8700 3.20 GHz CPU, and 16 GB DDR4 memory. For a fair comparison, all algorithms use the same stopping condition that the number of fitness evaluations reaches the maximum allowable number, which is set to 4000 for instances #1–#3, 8000 for #4–#6, 12000 for #7 and #11, and 16000 for #9–#11. In this setting, the CPU time consumed to solve the largest-size instance #11 is less than one second, which makes it appropriate to employ the algorithms to work out solutions for deliverymen selection and path planning in practice.

On each test instance, each algorithm is run 30 times, and the performance is evaluated based on the results over the 30 runs. Table 3 presents the median (med) and standard deviation (std) of the objective function values obtained by the algorithms on each test instance. For each instance, the best median value among the nine algorithms is shown

in bold. A superscript <sup>†</sup> indicates that there is a statistically significant difference (at 95% confidence level). We conduct Wilcoxon rank sum tests to compare the result of WVO with that of each other algorithm, and use a superscript <sup>†</sup> before the median value of the corresponding algorithm to indicate that there is a statistically significant difference (at 95% confidence level). Moreover, we present the median, maximum, minimum, first quartile (25%) and third quartile (75%) of the objective function values obtained by each algorithm among 30 runs on each instance in the box plots in Fig. 9.

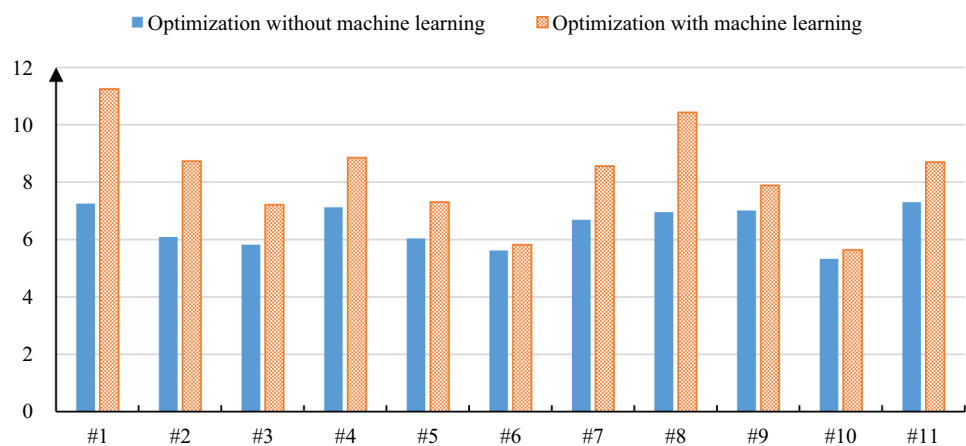
Among the 11 test instances, WVO-AB obtains the best median value on 10 instances except instance #5. On the smallest-size instances #1, WVO-AB, BBO, EBO, and AAA obtain the same best median value; on instance #2, WVO-AB, BBO and EBO obtain the same best median value; on the instance #5, BBO obtains the best median value, while WVO-AB obtains the second best; on each of the remaining eight instances, WVO-AB uniquely obtains the best median value. According to the statistical test results, the results of WVO-AB are significantly better than GA, ACO, PSO, DE, and the basic WVO on all 11 instances, significantly better than AAA on nine instances, and significantly better than BBO and EBO on eight instances. On the contrary, none of the other algorithms performs significantly better than WVO-AB on any instance. Although BBO achieves the best



**Fig. 9** Box plots of the objective function values of obtained by the nine algorithms on the test instances. Max: maximum; Min: minimum, Q1: the first quartile (25%); Q3: the third quartile (75%)



**Fig. 10** Revenues obtained by the method with machine learning and the method without



median value on instance #5, there is no significant difference between the results of BBO and WWO-AB on this instance.

Among the other eight comparative algorithms, the overall performance of PSO is the worst, mainly because the PSO's learning-from-history mechanism often causes the algorithm to be trapped in local optima. The crossover operators of GA and DE and the pheromone accumulation mechanism of ACO have similar negative effects on the search abilities of the algorithms. Therefore, in general, the performances of these four algorithms are significantly worse than those of the other five algorithms that have special mechanisms for balancing global exploration and local exploitation. Such mechanisms include migration operations of BBO and EBO, helical movement of AAA, and propagation of WWO, which can effectively maintain solution diversity, and therefore, suppress premature convergence. Compared to the basic WWO, WWO-AB uses adaptive breaking and population size reduction, which can further improve solution accuracy and accelerate the search process.

Moreover, the box plots in Fig. 9 and the standard deviation values in Table 3 show that, on most instances, the variance of the objective function values obtained by WWO-AB over 30 runs is much smaller than those of the other comparative algorithms. This indicates that, compared to the other algorithms, the results of WWO-AB algorithm are more robust. That is, when a deliveryman employs an algorithm to solve a problem instance, different runs of WWO-AB typically result in similar solutions, which helps to improve the user confidence to the algorithm.

In summary, WWO-AB exhibits the best overall performance among the nine algorithms on the test instances, which validates the effectiveness of the WWO evolutionary operators adapted to solve the considered takeaway order selection and delivery path planning problem. As the maximum running time to produce a solution is less than one second, it is practical for deliverymen to use the proposed WWO-AB to optimize order selection and delivery path planning to improve their performance and the corresponding revenue.

### Contributions of machine learning to optimization

Finally, we test the effects of the data-driven machine learning on the revenues of deliverymen. On the 11 test instances, we compare our method using machine learning to estimate order ready time and customer satisfaction level with the method without machine learning, i.e., simply using the expected order ready time given by the store and assuming the middle-level customer satisfaction. Figure 10 compares the revenues obtained by the two methods on the instances. As we can observe, on each instance, the method using machine learning achieves a higher revenue than that without machine learning. The higher the revenue, the larger the percentage of improvement of the method using machine learning over that without is. The results demonstrate that, using machine learning to estimate order ready time and customer satisfaction level more accurately, the deliverymen can select the orders and plan the routes in a more cost-effective manner to improve their revenues. The average revenue obtained by the method using machine learning is 8.22 yuan per hour, significantly better than the 6.47 of the method without machine learning. Such a significant improvement can greatly help both the takeaway deliverymen and company.

### Conclusion

The last years have observed a rapid growth of the takeaway delivery market. The increasing number of candidate orders and the corresponding pickup and service points has made order selection and path planning a key challenging problem to deliverymen. In this article, we formulate an integrated takeaway order selection and delivery path optimization problem, which involves uncertain order ready time and customer satisfaction level. We employ a machine-learning approach to infer the uncertain factors based on habit data of takeaway stores and customers. To efficiently solve the problem, we propose a hybrid evolutionary algorithm that adapts the WWO metaheuristic to solve the main prob-

lem of order selection and employs the tabu search method to quickly optimize the delivery path for each main solution. Experimental results on test instances constructed based on real food delivery application data demonstrate the performance advantages of the proposed algorithm compared to a set of popular evolutionary algorithms.

Our future work will be devoted to three aspects. First, the present work takes the uncertainty of order ready time into consideration, but assumes that the delivery time can be estimated in an accurate manner. However, in practice, the delivery time is also significantly affected by external factors such as traffic conditions. Therefore, we will consider the uncertain delivery time in the problem, and utilize interfaces from map navigation applications such as Baidu and Gaode to estimate the delivery time. Second, the delivery path is limited by the maximum distance of the vehicle (typically, electronic bicycle) used by the deliveryman. Therefore, we will consider the capacity of battery and its recharging in the problem and solution method [57]. Third, we will extend the problem and algorithm to enable dynamical order selection and path re-planning for deliverymen when they are on the way.

**Funding** This work was supported by National Natural Science Foundation of China (Grant No. 61872123) and Natural Science Foundation of Zhejiang Province (Grant No. LR20F030002).

## Declarations

**Conflict of interest** The authors declare that they have no conflict of interest.

**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

## References

1. Ai TJ, Kachitvichyanukul V (2009) A particle swarm optimization for the vehicle routing problem with simultaneous pickup and delivery. *Comput Oper Res* 36(5):1693–1702. <https://doi.org/10.1016/j.cor.2008.04.003>
2. Ali IM, Essam D, Kasmarik K (2021) Novel binary differential evolution algorithm for knapsack problems. *Inf Sci* 542:177–194. <https://doi.org/10.1016/j.ins.2020.07.013>
3. Azadi Hematabadi A, Akbari Foroud A (2018) Optimizing the multi-objective bidding strategy using min–max technique and modified water wave optimization method. *Neural Comput Appl*. <https://doi.org/10.1007/s00521-018-3361-0>
4. Baker BM, Ayechev MA (2003) A genetic algorithm for the vehicle routing problem. *Comput Oper Res* 30(5):787–800. [https://doi.org/10.1016/S0305-0548\(02\)00051-5](https://doi.org/10.1016/S0305-0548(02)00051-5)
5. Bansal JC, Deep K (2012) A modified binary particle swarm optimization for knapsack problems. *Appl Math Comput* 218(22):11042–11061. <https://doi.org/10.1016/j.amc.2012.05.001>
6. Bianchessi N, Righini G (2007) Heuristic algorithms for the vehicle routing problem with simultaneous pick-up and delivery. *Comput Oper Res* 34(2):578–594. <https://doi.org/10.1016/j.cor.2005.03.014>
7. Budak G, Chen X (2020) Evaluation of the size of time windows for the travelling salesman problem in delivery operations. *Complex Intell Syst* 6:681–695. <https://doi.org/10.1007/s40747-020-00167-y>
8. Chen H, Hou Y, Luo Q, Hu Z, Yan L (2018) Text feature selection based on water wave optimization algorithm. In: 10th International conference on advanced computational intelligence, pp 546–551. <https://doi.org/10.1109/ICACI.2018.8377518>
9. Chu H, Zhang W, Bai P, Chen Y (2021) Data-driven optimization for last-mile delivery. *Complex Intell Syst*. <https://doi.org/10.1007/s40747-021-00293-1>
10. Chu PC, Beasley JE (1998) A genetic algorithm for the multidimensional knapsack problem. *J Heurist* 4(1):63–86. <https://doi.org/10.1023/A:1009642405419>
11. Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans Evol Comput* 6(2):182–197. <https://doi.org/10.1109/4235.996017>
12. Dumas Y, Desrosiers J, Soumis F (1991) The pickup and delivery problem with time windows. *Eur J Oper Res* 54(1):7–22. [https://doi.org/10.1016/0377-2217\(91\)90319-Q](https://doi.org/10.1016/0377-2217(91)90319-Q)
13. Eksioglu B, Vural AV, Reisman A (2009) The vehicle routing problem: a taxonomic review. *Comput Ind Eng* 57(4):1472–1483. <https://doi.org/10.1016/j.cie.2009.05.009>
14. Gao W, Jiang G (2018) The research based on the path optimization problem of takeaway delivery. *Inf Commun* 5:20–22
15. Glover F (1990) Tabu search - part II. *ORSA J Comput* 2(1):4–32
16. Glover F (1991) Multilevel tabu search and embedded search neighborhoods for the traveling salesman problem. Working paper, University of Colorado
17. Hsu CI, Hung SF, Li HC (2007) Vehicle routing problem with time-windows for perishable food delivery. *J Food Eng* 80(2):465–475. <https://doi.org/10.1016/j.jfoodeng.2006.05.029>
18. Huang X, Wu XQ, Yuan QL (2017) Application of ant colony algorithm in the planning of takeaway distribution route. *Value Eng* 36(5):65–67
19. İç YT, Özel M, Kara I (2017) An integrated fuzzy TOPSIS-knapsack problem model for order selection in a bakery. *Arab J Sci Eng* 42:5321–5337. <https://doi.org/10.1007/s13369-017-2809-3>
20. Ji S, Zheng Y, Wang Z, Li T (2019) Alleviating users' pain of waiting: effective task grouping for online-to-offline food delivery services. In: The World Wide Web Conference, pp 773–783. <https://doi.org/10.1145/3308558.3313464>
21. Ke L, Feng Z, Ren Z, Wei X (2010) An ant colony optimization approach for the multidimensional knapsack problem. *J Heurist* 16(1):65–83. <https://doi.org/10.1007/s10732-008-9087-x>
22. Li W, Wu Y, Kumar PNR, Li K (2020) Multi-trip vehicle routing problem with order release time. *Eng Optim* 52(8):1279–1294. <https://doi.org/10.1080/0305215X.2019.1642880>
23. Liao W, Zhang L, Wei Z (2020) Multi-objective green meal delivery routing problem based on a two-stage solution strategy. *J Cleaner Prod* 258:120627. <https://doi.org/10.1016/j.jclepro.2020.120627>
24. Ling HF, Su ZL, Jiang XL, Zheng YJ (2021) Multi-objective optimization of integrated civilian-military scheduling of medical supplies for epidemic prevention and control. *Healthcare* 9(2). <https://doi.org/10.3390/healthcare9020126>

25. Liu L, Li K, Liu Z (2017) A capacitated vehicle routing problem with order available time in e-commerce industry. *Eng Optim* 49(3):449–465. <https://doi.org/10.1080/0305215X.2016.1188092>
26. Liu L, Liu S (2020) Integrated production and distribution problem of perishable products with a minimum total order weighted delivery time. *Mathematics* 8(2). <https://doi.org/10.3390/math8020146>
27. Liu L, Liu S, Niu B, Tan H (2020) A capacitated vehicle routing problem with order release time based on a hybrid harmony search. In: Chen X, Yan H, Yan Q, Zhang X (eds) *Machine learning for cyber security*, Springer, Cham, pp 235–249. [https://doi.org/10.1007/978-3-030-62460-6\\_21](https://doi.org/10.1007/978-3-030-62460-6_21)
28. Liu P, Hendarianpour A, Razmi J, Sangari MS (2021) A solution algorithm for integrated production-inventory-routing of perishable goods with transshipment and uncertain demand. *Complex Intell Syst*. <https://doi.org/10.1007/s40747-020-00264-y>
29. Liu S, Jiang H, Chen S, Ye J, He R, Sun Z (2020) Integrating Dijkstra's algorithm into deep inverse reinforcement learning for food delivery route planning. *Transp Res Part E* 142:102070. <https://doi.org/10.1016/j.tre.2020.102070>
30. Liu Y (2019) An optimization-driven dynamic vehicle routing algorithm for on-demand meal delivery using drones. *Comput Oper Res* 111:1–20. <https://doi.org/10.1016/j.cor.2019.05.024>
31. Ma ZJ, Wu Y, Dai Y (2017) A combined order selection and time-dependent vehicle routing problem with time widows for perishable product delivery. *Comput Ind Eng* 114:101–113. <https://doi.org/10.1016/j.cie.2017.10.010>
32. Reyes D, Erera A, Savelsbergh M, Sahasrabudhe S, O'Neil R (2018) The meal delivery routing problem. Tech. rep., Stewart School of Industrial Engineering, Georgia Institute of Technology
33. Salhi S, Gutierrez B, Wassan N, Wu S, Kaya R (2020) An effective real time GRASP-based metaheuristic: application to order consolidation and dynamic selection of transshipment points for time-critical freight logistics. *Expert Syst Appl* 158:113574. <https://doi.org/10.1016/j.eswa.2020.113574>
34. Simon D (2008) Biogeography-based optimization. *IEEE Trans Evol Comput* 12(6):702–713. <https://doi.org/10.1109/TEVC.2008.919004>
35. Soman JT, Patil RJ (2020) A scatter search method for heterogeneous fleet vehicle routing problem with release dates under lateness dependent tardiness costs. *Expert Syst Appl* 150:113302. <https://doi.org/10.1016/j.eswa.2020.113302>
36. Statista: Share of meal deliveries ordered online in the united kingdom (UK) from 2008 to 2018 (2019). <https://www.statista.com/statistics/675788/food-deliveries-ordered-onlineunited-kingdom-uk/>
37. Sun P (2019) Your order, their labor: an exploration of algorithms and laboring on food delivery platforms in China. *Chin J Commun* 12(3):308–323. <https://doi.org/10.1080/17544750.2019.1583676>
38. Trustdata: Analysis report on the development of china's takeaway industry in the first half of 2019 (2019). <http://report.itrustdata.com/report/pdf/2019%E5%B9%B4%E4%B8%8A%E5%8D%8A%E5%B9%B4%E4%B8%AD%E5%9B%BD%E5%A4%96%E5%8D%96%E8%A1%8C%E4%B8%9A%E5%8F%91%E5%B1%95%E5%88%86%E6%9E%90%E6%8A%A5%E5%91%8A.pdf>
39. Ulmer MW, Thomas BW, Campbell AM, Woyak N (2021) The restaurant meal delivery problem: dynamic pickup and delivery with deadlines and random ready times. *Transp Sci* 55(1):75–100. <https://doi.org/10.1287/trsc.2020.1000>
40. Wu JY, Zhang MX, Wu X, Zheng YJ (2021) A water wave optimization algorithm for order selection and delivery path optimization for takeaway deliverymen. In: 11th International conference on information science and technology. IEEE
41. Yan HF, Cai CY, Liu DH, Zhang MX (2019) Water wave optimization for the multidimensional knapsack problem. In: International conference on intelligent computing, Springer, Nanchang, pp 688–699
42. Yildiz B, Savelsbergh M (2019) Provably high-quality solutions for the meal delivery routing problem. *Transp Sci* 53(5):1372–1388. <https://doi.org/10.1287/trsc.2018.0887>
43. Yu H, Luo X (2019) Minimizing latency in online pickup and delivery problem with single pickup point. In: International conference on industrial engineering and systems management, pp 1–6. <https://doi.org/10.1109/IESM45758.2019.8948173>
44. Zhang L, Liao W (2020) Interactively solving the takeout delivery problem based on customer satisfaction and operation cost. In: *HCI International 2020 - Posters*, Springer, Cham, pp 738–745
45. Zhang X, Wu C, Li J, Wang X, Yang Z, Lee JM, Jung KH (2016) Binary artificial algae algorithm for multidimensional knapsack problems. *Appl Soft Comput* 43:583–595. <https://doi.org/10.1016/j.asoc.2016.02.027>
46. Zhang Y, Sun L, Hu X, Zhao C (2019) Order consolidation for the last-mile split delivery in online retailing. *Transp Res Part E* 122:309–327. <https://doi.org/10.1016/j.tre.2018.12.011>
47. Zhao B, Deng C, Yang Y, Peng H (2012) Novel binary biogeography-based optimization algorithm for the knapsack problem. In: International conference in swarm intelligence, Springer, New York, pp 217–224
48. Zheng J, Wang L, Wang S, Liang Y, Pan J (2021) Solving two-stage stochastic route-planning problem in milliseconds via end-to-end deep learning. *Complex Intell Syst*. <https://doi.org/10.1007/s40747-021-00288-y>
49. Zheng Y, Zhang B, Xue J (2016) Selection of key software components for formal development using water wave optimization. *J Softw* 27(4):933–942. <https://doi.org/10.13328/j.cnki.jos.004964> (in Chinese)
50. Zheng YJ (2015) Water wave optimization: a new nature-inspired metaheuristic. *Comput Oper Res* 55(1):1–11. <https://doi.org/10.1016/j.cor.2014.10.008>
51. Zheng YJ, Ling HF, Xue JY (2014) Ecogeography-based optimization: enhancing biogeography-based optimization with ecogeographic barriers and differentiations. *Comput Oper Res* 50:115–127. <https://doi.org/10.1016/j.cor.2014.04.013>
52. Zheng YJ, Lu XQ, Du YC, Xue Y, Sheng WG (2019) Water wave optimization for combinatorial optimization: design strategies and applications. *Appl Soft Comput* 83:105611. <https://doi.org/10.1016/j.asoc.2019.105611>
53. Zheng YJ, Wang Y, Ling HF, Xue Y, Chen SY (2017) Integrated civilian-military pre-positioning of emergency supplies: a multi-objective optimization approach. *Appl Soft Comput* 58:732–741. <https://doi.org/10.1016/j.asoc.2017.05.016>
54. Zheng YJ, Xu XL, Ling HF, Chen SY (2015) A hybrid fireworks optimization method with differential evolution operators. *Neurocomputing* 148(1):75–82. <https://doi.org/10.1016/j.neucom.2012.08.075>
55. Zheng YJ, Zhang B (2015) A simplified water wave optimization algorithm. In: IEEE congress on evolutionary computation, pp 807–813. <https://doi.org/10.1109/CEC.2015.7256974>
56. Zhou XH, Zhang MX, Xu ZG, Cai CY, Huang YJ, Zheng YJ (2019) Shallow and deep neural network training by water wave optimization. *Swarm Evol Comput* 50:1–13. <https://doi.org/10.1016/j.swevo.2019.100561>
57. Zhou Y, Huang J, Shi J, Wang R, Huang K (2021) The electric vehicle routing problem with partial recharge and vehicle recycling. *Complex Intell Syst*. <https://doi.org/10.1007/s40747-021-00291-3>. (in press)

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.