Received: 15 November 2021 | Revised: 27 April 2022 | Accepted: 12 May 2022

IET Intelligent Transport Systems

DOI: 10.1049/itr2.12206

The Institution of Engineering and Technology WILEY

ORIGINAL RESEARCH

Meal delivery routing optimization with order allocation strategy based on transfer stations for instant logistics services

Jiulin Li[1] | **Senyan Yang**[2] | **Wenbo Pan**[2] | **Ziwen Xu**[1] | **Bo Wei**[3]

[1]International School, Beijing University of Posts and Telecommunications, Beijing, China

[2]School of Modern Post (School of Automation), Beijing University of Posts and Telecommunications, Beijing, China

[3]School of Electronic Engineering, Beijing University of Posts and Telecommunications, Beijing, China

Correspondence
Senyan Yang, School of Modern Post (School of Automation), Beijing University of Posts and Telecommunications, Beijing 100876, China.
Email: yangsenyan@bupt.edu.cn

Funding information
Research Innovation Fund for College Students of Beijing University of Posts and Telecommunications; National Natural Science Foundation of China, Grant/Award Number: 72001029; Research Start-up Fund for Introduction of Talents of Beijing University of Posts and Telecommunications

Abstract
The rapid development of instant logistics services has brought not only convenience to people's life but also a great challenge to traffic management. Due to the limited delivery range of instant delivery systems, customers are usually recommended the meals nearby or pay much higher delivery fees for long-distance delivery. This study proposes a novel order splitting strategy based on transfer stations for long-distance meal orders to satisfy the diverse customer demands. The meal delivery routing problem is addressed with an order allocation strategy based on transfer stations through developing a three-stage modelling framework consisting of order combination, splitting and delivery routing for the online-to-offline instant logistics services. Normal meal orders are combined by the DBSCAN algorithm, and the cross-regional long-distance orders are split by transfer stations. Based on order combination and splitting, a mixed integer programming model is constructed for the meal delivery routing problem and solved by the adaptive large neighbourhood search algorithm. The proposed algorithm converges quickly for the tested instances constructed based on real platform data. The proposed order allocation strategy can expand the delivery scopes of couriers effectively, stimulate more potential orders and guarantee the timeliness of meal delivery.

1 | INTRODUCTION

The rapid development of the Internet on information transmission speed, spreading scope, and exchange platform, make many traditional business models change [1]. The online-to-offline (O2O) meal delivery industry has been greatly developed in the past few years with a steady growth of customers and demands and brings great convenience to people's daily life [2, 3]. As the COVID-19 epidemic spreads around the world, more customers select the O2O meal delivery services for safe social distancing [3]. Online platforms provide customers with convenient ordering and delivery services [4]. Meanwhile, meal delivery services have caused a series of problems, such as traffic congestion, pollutant emissions, and crowding at restaurants [5]. By the end of 2019, the number of meal delivery consumers in China had reached 460 million, accounting for 50.7% of the Internet customers, and the transaction amount increased by 38.9% year-on-year to 392.7 billion yuan [6].

Restaurants, customers, couriers, and platforms are the main stakeholders of the O2O meal delivery service. The platform informs restaurants to prepare meals and dispatch couriers to pick up the meals from restaurants and deliver them to customers. Compared to the general last-mile express delivery, meal orders are expected to be served within an hour or much less if possible, with stricter requirements for timeliness [7, 8]. As the market competition for the O2O delivery business becomes increasingly fierce, many market-based strategies have been adopted by meal delivery platforms to reduce the waiting time of customers and improve delivery efficiency. Meal delivery companies strive to make order scheduling systems more intelligent and devote huge costs to the maintenance and upgrading of systems.

The instant delivery systems face challenges caused by the highly dynamic customer orders and system states [9]. Couriers are usually plagued by tight delivery time and high fines for overtime. Driven by the profits and pressure of the platforms, they tend to drive at a high speed to complete more orders and get better reviews from customers. Especially during peak hours, couriers even may speed, run red lights, and retrograde, which results in more traffic accidents. Intelligent order allocation and dispatch strategies are quite necessary for improving the efficiency, safety, and benefits of meal delivery platforms.

The O2O meal delivery platforms generate millions of orders every day. To ensure the timeliness of meal delivery services, the courier is usually restricted to serving the orders within a specified delivery region with higher route familiarity. The underlying spatio-temporal distribution of orders, customer demands, and courier behaviours can be extracted from the enormous data, which can be applied to improve service quality. In this study, similar normal orders with close locations of customers and restaurants are combined to obtain the delivery regions by the density-based spatial clustering of applications with noise (DBSCAN) algorithm based on the real data of a meal delivery platform.

The O2O meal delivery platforms usually recommend the meals from the restaurants within the limited range to customers. However, in real life, customers may want to order specified meals in other delivery regions far away, which need to be delivered by special couriers with higher prices instead of being served by the normal couriers. Because of such high expenses, few customers would order meals from restaurants far away. However, if these potential long-distance orders could be served by normal couriers like other normal orders, the delivery expenses will be greatly reduced. To satisfy the diverse demands of customers, this study addresses a transfer station-based order allocation strategy to solve the cross-regional delivery problem for long-distance orders of the O2O instant delivery systems. This order splitting strategy could effectively expand the scope of instant logistics services, stimulate customers' potential demands, and bring more benefits to the restaurants, customers, couriers and platform. This study aims to solve the integrated order combination, splitting and delivery routing problem for the instant logistics service. The main contributions are summarized as follows:

1. Meal orders are combined by the DBSCAN algorithm based on the location data of customers and restaurants to balance the allocation of delivery service resources. The orders for each cluster are served by specific couriers with higher familiarity with the working regions and routes, which can promote higher service efficiency and reliability of delivery systems.
2. A novel order splitting strategy based on transfer stations is proposed for the cross-regional long-distance delivery orders, which can expand the scope of instant logistics services and satisfy the diverse customer requests.
3. Meal instant delivery routing problem with order allocation strategy based on transfer stations is proposed for O2O meal delivery systems. The mixed-integer programming model minimizing the total travel time costs and overtime losses is solved by the heuristic algorithm based on the adaptive large neighbourhood search algorithm (ALNS).
4. The proposed model and solution approach is tested based on the instances of different sizes, which are constructed based on the real-life order data of instant delivery systems.

The remainder of this paper is organized as follows. Section 2 is the literature review. Section 3 describes the meal delivery routing problem for O2O meal delivery systems. Section 4 presents the related methods and models for order combination, order allocation, transfer station location, and delivery routing problems. Section 5 describes the solution algorithm based on ALNS. Section 6 analyses the computational results of the test instances. Section 7 analyses the economic benefit of transfer stations. Section 8 summarizes the conclusions and provides suggestions for future work.

2 | LITERATURE REVIEW

The meal delivery routing problem is an emerging research branch of the vehicle routing problem (VRP). Different from traditional research on VRP, the dynamic customer demands, the pairwise pickup and delivery, and the visiting precedence of restaurants and customers are usually considered for this problem. Considering the unprecedented large scale and timeliness of orders in instant delivery systems, a series of efficient solution approaches have been developed to solve it. Liu [10] proposes the optimization-driven dynamic vehicle routing problem for meal delivery systems to dispatch drones and deliver orders. Ulmer et al. [11] present a meal delivery problem with dynamic pickup and delivery by defining a parametrizable cost function approximation, considering the stochasticity of the customer's deadlines and the restaurant's ready time. Reyes et al. [7] define the meal delivery routing problem (MDRP) and develop a dynamic deterministic modelling framework and rolling-horizon solution algorithm. Yildiz et al. [8] develop a novel framework based on work packages for the MDRP and a column and row generation solution method.

Order allocation strategies are essential for improving the service efficiency of platforms and optimizing delivery routing, attracting much attention. Liu et al. [4] study the order-assignment problem integrated with travel-time predictors for food delivery service solved by the branch-and-price algorithm, which can capture routing behaviour from actual operational data. Liao et al. [12] propose a green meal delivery routing problem, focusing on the order combination and delivery routing, and developed a two-stage solution strategy. Steever et al. [13] study the dynamic vehicle routing problem for food delivery with the situation that multiple restaurants are included in a single order based on a split or no-split delivery policy and develop a proactive heuristic algorithm to solve it.

The related pickup and delivery problem (PDP) has been extensively studied in recent years, which can be classified into many-to-many, one-to-many-to-one, and one-to-one problems [14]. Savelsbergh and Sol [15] introduce the general PDP, and the loads are transported from origins to destinations without transshipment. Ropke and Pisinger [16] develop a heuristic solution framework based on ALNS to solve the PDP with time

TABLE 1 Summary of the related studies

Paper	Problem	Order combination	With transfer or not	Objective function	Algorithm
Liu et al. [4]	Order assignment	YES	NO	Minimize total delay	Branch-and-price
Yildiz et al. [8]	MDRP	NO	NO	Minimize total courier compensation	Column and row generation
Liao et al. [12]	MDRP	YES	NO	Maximize customer satisfaction and rider balance utilization; Minimize carbon footprint	ALNS
Steever et al. [13]	MDRP	NO	NO	Maximize the total earliness with penalizing lateness to customers	Proactive heuristic
Aziez et al. [19]	MPDPTW	NO	NO	Minimize total transportation cost	Branch-and-cut
Kohar and Jakhar [20]	MPDPTW	NO	NO	Minimize total travel cost	Branch-and-cut
Voigt et al. [21]	PDP	NO	YES	Minimize total traveling costs and compensation	ALNS
Wolfinger [22]	PDPTW	NO	YES	Minimize total travel costs and transshipment costs	LNS
Masson et al. [23]	PDP	NO	YES	Minimize the number of vehicles and total distance travelled	ALNS
This paper	MDRP	YES	YES	Minimize total travel time costs and overtime losses	ALNS

windows (PDPTW). Keçeci et al. [17] consider the heterogeneous fixed fleet for the VRP with simultaneous pickup and delivery and develop a hybrid metaheuristic approach based on simulated annealing and local search algorithms. Wang et al. [18] study the PDPTW with the mixed-load strategy for catering distribution services, and construct the multi-commodity flow optimization model in the framework space-time-state network. Aziez et al. [19] propose the multi-pickup and delivery problem with time windows (MPDPTW) and develop an exact branch-and-cut algorithm. Kohar and Jakhar [20] study the MPDPTW for an online food delivery system, allowing a customer can order meals from different restaurants.

To increase the flexibility of delivery systems, some studies focus on the PDP with transshipments. Voigt et al. [21] propose the PDP considering the possibility of transshipments between drivers for crowdsourced logistics in last-mile delivery, and develop a heuristic solution approach based on ALNS to solve it. Wolfinger [22] addresses a PDPTW with split loads and transshipments and constructs an arc-based mixed integer model, which is solved by the large neighbourhood search (LNS) algorithm. Masson et al. [23] study the PDP with transfer, and the delivery requests can change vehicle at intermediate points between the pickup and delivery points, and propose a heuristics ALNS algorithm.

To present the contributions and innovations of this study, we summarize the relevant literature in terms of problems, models, and algorithms in Table 1. Previous studies on the MDRP mainly optimize the couriers' routes in traditional delivery mode and do not consider setting up a transfer node for meal orders, and only a few studies combine orders based on clustering algorithms. Some studies on PDP consider the transshipment during the delivery process, but not for the meal delivery scenario, and thus do not consider the combination of orders. To the best of our knowledge, this study first proposes a novel order splitting strategy based on transfer stations for long-distance meal orders to expand the scope of instant logistics services. To improve the service efficiency of couriers, the delivery regions are divided by combining similar meal orders by the DBSCAN algorithm. In addition, the objective functions of previous studies are mainly to minimize total traveling cost, compensation, or delay, while this study proposes a piecewise overtime loss function as the penalty cost of overtime delivery from the perspective of the practical operation of instant delivery systems.

3 | PROBLEM DESCRIPTION

To ensure service efficiency, couriers usually serve the orders in the specified regions due to high route familiarity. However, the diversity of customer preferences in meals leads to the unbalanced distribution of orders. Some customers may order the meals of the restaurants in other delivery regions, which would be served by special couriers with higher delivery costs. These cross-regional long-distance orders would increase the work pressure of instant delivery systems, especially during meal rush hours. To reduce the service cost and stimulate potential customer needs, this study focuses on the instant delivery routing problem with the order allocation strategy based on transfer stations for the O2O meal delivery services. We develop a three-stage research framework consisting of order combination, splitting, and delivery routing to optimize the resource allocation of instant delivery systems, as shown in Figure 1.

First, the DBSCAN clustering algorithm is utilized for order combination. The basic input unit for clustering is the normal orders, rather than the customer or restaurant nodes. Normal meal orders with close locations of customers and restaurants are clustered together based on the historical coordinate data,

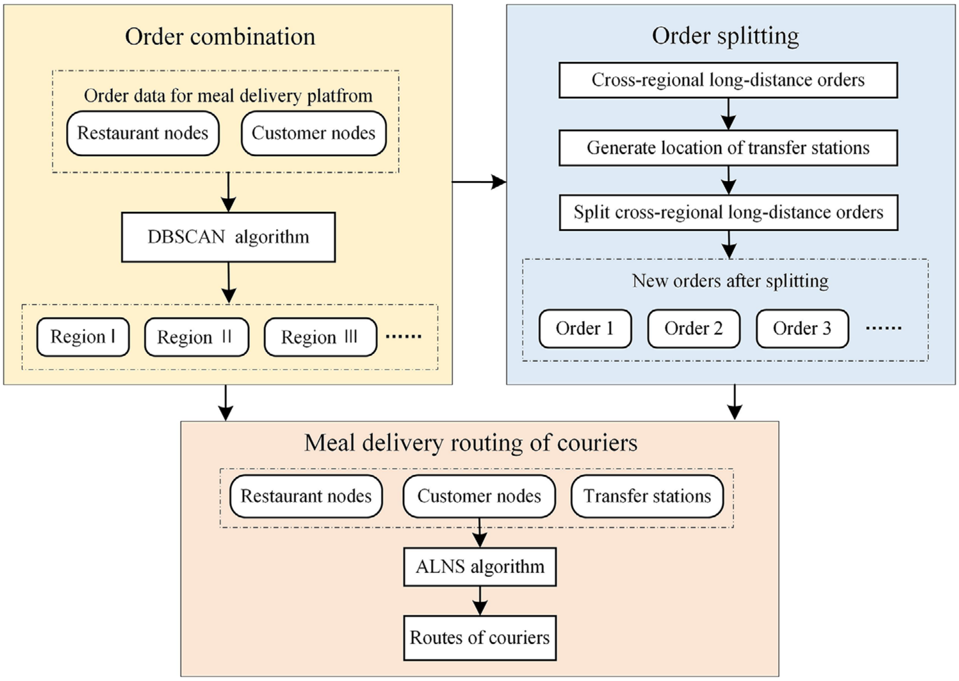


FIGURE 1 Research framework

forming different delivery regions. The customer node and the restaurant node of each normal order appear in pairs in their corresponding cluster. In other words, based on the order combination, the service range of the platform can be divided into multiple regions. Couriers' familiarity with their working regions can be guaranteed through the order combination.

Second, a novel order splitting strategy based on transfer stations is proposed to overcome the limitation of short delivery distances. Based on the division of delivery regions, when the customer and restaurant nodes of a new order are located in different delivery regions, and the distance between the customer and restaurant node is larger than the pre-set threshold, it can be treated as a cross-regional long-distance order. The threshold value is determined based on the statistical analysis of historical data, which is larger than the distance of any meal order. These special orders are split into short-distance orders by transfer stations. Transfer stations can be treated as the ending nodes for the front part orders and the starting nodes for the latter part orders. Once a meal order is delivered to a transfer station, the delivery system would dispatch another courier in the adjacent region to pick up the meal from the transfer station and complete the rest of the delivery process.

The order splitting strategy based on transfer stations is illustrated in Figure 2. There are three delivery regions, including delivery region I, II and II. Order 1 and order 2 are across two different delivery regions. The distances of order 1 and order 2 are both larger than the threshold value, and then they are split into two short-distance orders by transfer stations A and B, respectively. Order 1 is delivered by couriers *a* and *b*, passing through transfer station A. Order 2 is delivered by couriers *c* and *d*, passing through transfer station B. The restaurant and customer of order 3 are in the same delivery region, and thus it does not need to be split and is directly served by courier *e*.

Finally, the instant delivery system aims to find the optimal delivery routes for couriers with the minimum total operation cost and overtime loss after order combination and splitting. The ALNS algorithm is used to solve the meal instant delivery routing problem. Different from previous studies, transfer station nodes need to be visited to serve the cross-regional long-distance orders.

To guarantee the feasibility of order allocation strategy based on transfer stations, this study makes the assumptions as follows:

1. All the cross-regional long-distance orders would be split by transfer stations;
2. The front/latter part orders after splitting are served by the couriers in the corresponding delivery regions;
3. Transfer stations can be operated by specific workers or setting intelligent storage cabinets, undertaking the responsibility of keeping meals safe while the split orders are temporarily at the stations;
4. The construction, maintenance and operation expenses of transfer stations are paid for by the platform.

The delivery fee of each cross-regional long-distance order paid by the customer consists of three parts, including the delivery fee of the front part order, the delivery fee of the latter part order, and the service fee of the transfer station. The safety of the meal orders can be guaranteed by the corresponding responsible parties. If a transfer station is operated by specific workers, the platform needs to pay the salary of the workers. For the split orders, the corresponding couriers are responsi-

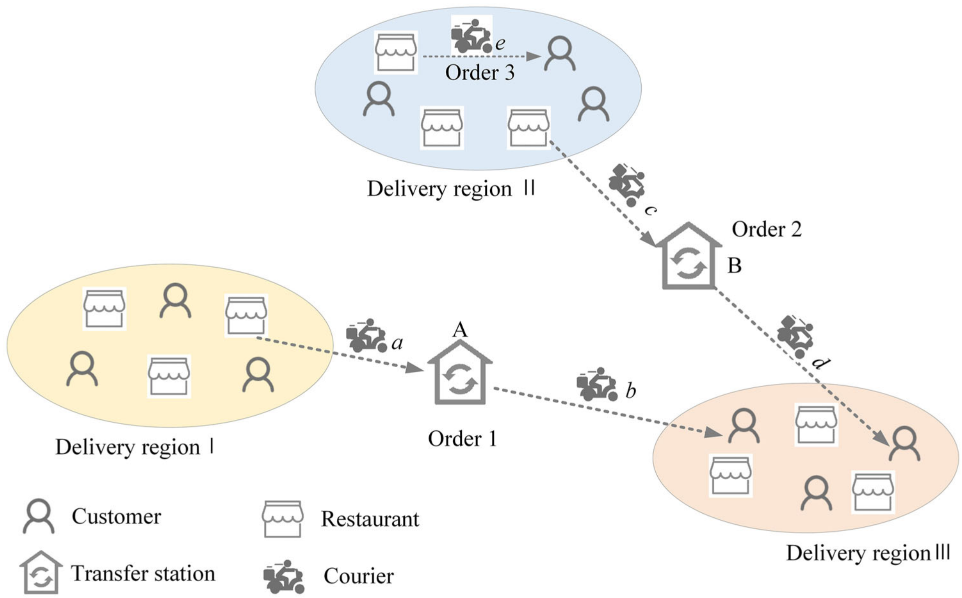


FIGURE 2 Schematic diagram of order splitting strategy based on transfer stations

ble for the meals during the delivery process, while the workers are responsible for the meals when they are temporarily kept at the transfer stations. If intelligent storage cabinets are set to serve as the transfer stations, the platform does not need to pay the employee salary, because it does not need to employ workers to operate the stations. The safety of meals can be guaranteed by taking photos and entering the password in the cabinets by the corresponding couriers.

The benefits of this novel order allocation strategy based on transfer stations can be guaranteed by designing a reasonable business operation mode. The lower delivery cost for the delivery mode based on transfer stations can attract more restaurants and customers to the platform. A larger delivery range means more choices for customers, which can increase the number of orders, and make the platform and restaurants get more profits.

4 | METHODS AND MODELS

This section presents the related methods and models for order combination, order splitting, and meal delivery routing. First, the DBSCAN algorithm is applied for order combination. Then, the conditions of order splitting and the rules for the location of transfer stations are described, and the order splitting strategy is proposed based on transfer stations. Finally, based on order combination and splitting, a mixed-integer programming model is built for the meal delivery routing problem.

4.1 | O2O meal delivery process

The O2O meal delivery process is presented in Figure 3, involving the participation of the platform, restaurants, couriers, and customers. At a certain time, some customers order meals through the platform. After receiving the meal orders, the restaurants begin to prepare the meals. Meanwhile, the platform would generate order records, combine similar meal orders, split the cross-regional long-distance orders, and estimates the delivery time for each order. Then, the platform assigns these orders to the couriers in corresponding regions. If the couriers receive normal orders, they would pick up the meals from restaurants, and directly deliver them to customers within the promised delivery time. If the couriers receive the front part orders after splitting, they need to pick up the meals from restaurants and deliver them to transfer stations. If the couriers receive the latter part orders after splitting, they need to pick up the meals from transfer stations and deliver them to customers.

4.2 | Order combination based on DBSCAN

The massive orders of the instant delivery system are widely distributed in space. In this study, to guarantee the service efficiency and reliability of delivery systems, the large-scale regions are divided into several sub-regions. The DBSCAN algorithm is utilized to combine similar orders during a certain period, and it is a density-based spatial clustering algorithm [24]. Neighbourhood size and density threshold are the key parameters of the DBSCAN algorithm. The clusters of arbitrary shapes in space can be discovered by connecting high-density points in the neighbourhood, and the number of clusters does not need to be determined in advance. The number of objects in a cluster should not be less than a given threshold. DBSCAN algorithm is an efficient approach for dealing with large spatial data with noise.

The normal orders with compact points of restaurants and customers in space are clustered together, which can be treated

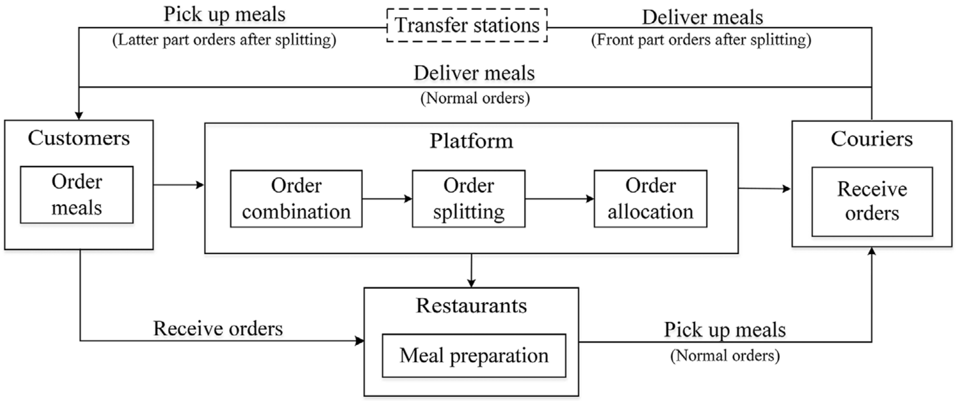


FIGURE 3 O2O meal delivery process

as a delivery region and served by serval specific couriers. In this way, the normal orders served by each courier can be more concentrated. Couriers are not allowed to cross regions, and their familiarity with the regions and routes is conducive to improving delivery efficiency.

The main order information provided by the instant delivery systems includes the locations of pickup points (restaurants) and delivery points (customers), meal ready time, promising delivery time, etc. In this study, the longitude and latitude of customers and restaurants are treated as four input features. The sub-regions with the corresponding points of customers and restaurants are output. The detailed algorithm is shown in Algorithm 1.

4.3 | Order splitting strategy based on transfer stations

The order splitting strategy based on transfer stations is proposed to split the cross-regional long-distance orders. These orders to be split should meet the following criteria: (1) the distances between the customer node and restaurant node are larger than the threshold value; (2) the customer node and restaurant node are located in different delivery regions. First, the distances between the restaurants and customers for orders are calculated based on their latitude and longitude. Then, the threshold value is determined based on the statistics of order distances after data cleaning. This threshold is larger than the distances between the restaurants and customers of all orders. If an order satisfies the above two rules, it can be treated as a cross-regional long-distance order, which needs to be split by a transfer station. The setting of transfer stations can expand the scope of instant logistics services and satisfy the diverse demands of customers.

Considering the characteristics of the complex urban road network traffic state, the time-weighted Manhattan distance is calculated to determine the location (x, y) of transfer stations, as shown in Equations (1) and (2). The locations of transfer stations are calculated for the cross-regional long-distance orders to be split by taking the time-weighted average of the latitude and longitude of the related restaurant and customer points based on the estimated order delivery time.

$$x = \sum_{i \in P} (t_i \times x_i) \Big/ \sum_{i \in P} t_i \tag{1}$$

$$y = \sum_{i \in P} (t_i \times y_i) \Big/ \sum_{i \in P} t_i \tag{2}$$

where x denotes the longitude of the transfer station; y denotes the latitude of the transfer station; P denotes the set of points, including the restaurants and customers related to the long-distance orders to be split; x_i denotes the longitude of node i, $i \in P$; y_i denotes the latitude of node i, $i \in P$; t_i denotes the estimated delivery time of the O2O meal delivery platform for the order related to node i, $i \in P$. Each cross-regional long-distance order can be split into short-distance orders by transfer stations.

The order splitting strategy based on transfer stations brings certain benefits to the customers, couriers, restaurants, and platform from different perspectives. As for customers, the main advantage of the proposed order splitting strategy is that the meal requests far away from customers can be served by normal couriers with lower delivery expenses, improving the availability of rich meals. For the traditional instant delivery mode, customers need to be served by specific couriers with the high delivery expense for the original long-distance orders. Moreover, not all restaurants can receive long-distance meal orders, restricting customers' consumption. In this study, the proposed order splitting strategy based on transfer stations can greatly expand the delivery range and satisfy the diverse needs of customers. In short, customers can get more meal choices with lower delivery prices.

The cross-regional long-distance orders can be split into short-distance orders by transfer stations. Couriers only deliver meals in specific regions they belong to, so they are more familiar with the streets and buildings in the regions, which can reduce the time consumption caused by the unacquainted delivery environment and the occurrence of a delivery timeout to a certain extent. Before the front courier arrives at the transfer station, the platform can allocate the latter courier in advance and send the order information such as the expected arrival time

ALGORITHM 1 DBSCAN algorithm for the combination of normal orders

Input: The normal order set N, the radius ε, the neighborhood density threshold *Minpts*.

Output: Clustering result U.

Initialization: $N \leftarrow$ normalization (N), initial order cluster $C_{order} = \emptyset$, clustering result $U = \emptyset$.

Repeat:

$n \leftarrow N$, select an unvisited order from N and mark it as visited;

If n belongs to a cluster **or** n is marked as a noise point **Then**:

Continue;

Else:

$G \leftarrow RangeQuery(n, \varepsilon)$, return all points within n's $\varepsilon -$ neighborhood (including n);

If $|G| < MinPts$ **Then**:

Mark n as a noise point

Else:

$C_{order} \leftarrow$ Next order cluster, created a new cluster and n is marked as the core point of this cluster;

Add n to C_{order};

Repeat:

$n' \leftarrow$ Select an unvisited order from G and mark it as visited;

$G' \leftarrow RangeQuery(n', \varepsilon)$, return all points within n''s $\varepsilon -$ neighborhood (including n');

If $|G| \geq MinPts$ **Then**:

$G \leftarrow G \cup G'$;

End if

If $n' \notin$ any cluster **Then**:

Add n' to C_{order};

End if

Until all orders in G are traversed;

Add C_{order} to clustering result U;

End if

End if

Until all the normal orders in N are traversed;

Output: Clustering result U.

of the meals. Then, the courier of the latter part can arrive at the transfer station on time or even in advance to complete the remaining delivery process, which can guarantee service efficiency. In addition, the setting of transfer stations can expand the delivery scopes of normal couriers, stimulate more potential customer demands, and then attract more restaurants to receive long-distance orders, bringing more revenues to both the restaurants and the platform.

4.4 | Model for meal delivery routing problem

Some assumptions are made for the mathematical model of the meal delivery routing problem, as follows:

TABLE 2 Sets, indices, parameters and variables used for model formulation

Notation	Definition
C	Set of customers
R	Set of restaurants
Z	Set of transfer stations
M	Set of couriers dispatched by the system
N	Set of orders
V	Set of all nodes, including customers, restaurants, and transfer stations, $V = C \cup R \cup Z$
A	Set of arcs between nodes
c	Index of customers, $c \in C$
r	Index of restaurants, $r \in R$
i, j, k	Index of nodes, $i, j, k \in V$
m	Index of couriers dispatched by the system, $m \in M$
o_m	Index of departure nodes for courier m, $o_m \in C$
d_m	Index of arrival nodes for courier m, $d_m \in R$
n	Index of orders, $n \in N$
(c_n, r_n)	The customer and restaurant node pair of order n, $c_n \in C$, $r_n \in R$
T_{im}	The time when the courier $m \in M$ arrives at node $i \in V$
T_{im}'	The time when the courier $m \in M$ leaves node $i \in V$
t_{ij}	Travel time from node i to node j
t_{si}	Pickup or delivery service time length at node i, $i \in V$
τ_{rn}	The time the restaurant r prepares the order n well, $\forall r \in R$, $\forall n \in N$
L_c	The promising delivery time for the customer c, $\forall c \in C$
L_c'	The latest delivery time that customer c can accept, and $L_c' > L_c$
ρ_1, ρ_2, ρ_3	The penalty factor for arriving later than the customer's service time window
x_{ijm}	Binary decision variable equals 1 if courier m travels to node j from node i, otherwise 0.
y_{mn}	Binary decision variable equals 1 if order n is served by courier m, otherwise 0.

1. Due to the limitation of delivery ranges, normal orders are served by the couriers for each delivery region;
2. The delivery ability gap among different couriers is negligible;
3. Loading capacity of couriers is large enough for the meal delivery process, not considering vehicle capacity limits.
4. The construction, maintenance and operation costs of transfer stations are not considered.

The sets, indices, parameters and variables used for model formulation are shown in Table 2.

Considering the negative impacts on the customers and platform caused by overtime delivery, overtime loss function $Loss(t_c)$ is defined for each customer in Equation (3), which is piecewise related to the delivery time t_c and promising time L_c, illustrated in Figure 4. T_{c1}, T_{c2}, T_{c3} are the time demarcation

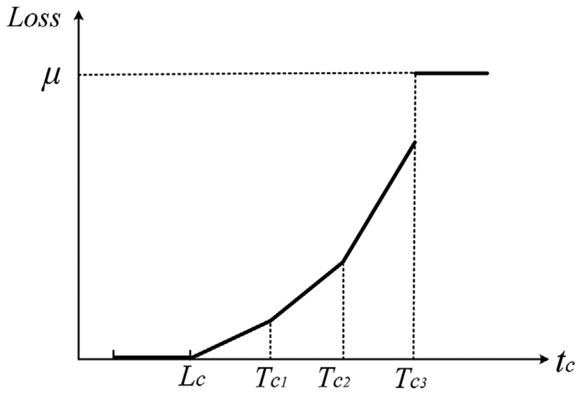


FIGURE 4 Overtime loss function $Loss(t_c)$

points, and $T_{c1} = L_c + \Delta t_1$, $T_{c2} = T_{c1} + \Delta t_2$, $T_{c3} = T_{c2} + \Delta t_3$. If the final delivery time $t_c$is within the promising delivery time L_c, $Loss(t_c)$ is 0. If the final delivery time $t_c$is slightly beyond the promising delivery time L_c, additional penalty cost would be considered for the calculation of $Loss(t_c)$. Moreover, the longer the overtime of orders, the larger the positive penalty parameter, that is $\rho_3 > \rho_2 > \rho_1 > 0$. T_{c3} is the latest delivery time that customers can accept. Once the overtime is longer than T_{c3}, $Loss(t_c)$ is set as a very large value μ, meaning that delivery would not be allowed. The setting of $Loss(t_c)$ allows overtime service, which can make the instant logistics service flexible.

$$Loss(t_c) = \begin{cases} 0, & t_c \le L_c \\ \rho_1(t_c - L_c), & L_c < t_c \le T_{c1} \\ \rho_1(T_{c1} - L_c) + \rho_2(t_c - T_{c1}) & T_{c1} < t_c \le T_{c2} \\ \rho_1(T_{c1} - L_c) + \rho_2(T_{c2} - T_{c1}) + \rho_3(t_c - T_{c2}), & T_{c2} < t_c \le T_{c3} \\ \mu, & t_c > T_{c3} \end{cases} \tag{3}$$

A mixed-integer programming model is constructed for the meal delivery routing problem with an order allocation strategy based on transfer stations.

$$\text{Min } L = \sum_{m \in M} \sum_{j \in V} \sum_{i \in V} t_{ij} x_{ijm} + \sum_{c \in C} Loss\,(t_c) \tag{4}$$

$$\sum_{m \in M} \sum_{j \in V} x_{ijm} = 1,\ \forall i \in C \cup R \tag{5}$$

$$\sum_{j \in V} x_{o_m jm} = 1, \forall m \in M \tag{6}$$

$$\sum_{i \in V} x_{id_m m} = 1, \forall m \in M \tag{7}$$

$$\sum_{i \in V} x_{ikm} - \sum_{j \in V} x_{kjm} = 0,\ \forall m \in M,\ \forall k \in V \tag{8}$$

$$\sum_{m \in M} y_{mn} = 1, \forall n \in N \tag{9}$$

$$T_{im} + t_{si} + t_{ij} \le T_{jm}, \text{ when } x_{ijm} = 1, \forall i, j \in V,\ \forall m \in M \tag{10}$$

$$T_{rm}{}' \ge \tau_{rn}, \forall r \in R, \forall n \in N,\ \forall m \in M \tag{11}$$

$$T_{cm} \le L_c', \forall m \in M, \forall c \in C \tag{12}$$

$$T_{c_n m} \le T_{r_n m}, \forall m \in M, \forall n \in N \tag{13}$$

The objective function (4) minimizes the total travel time costs and overtime losses; Constraints (5) guarantee each customer or restaurant node should be visited exactly once by one courier. Constraints (6), (7) and (8) are the network flow balance constraints, guaranteeing that the arcs of each courier dispatched by the system can be connected into a path. Constraints (9) guarantee that each order should be served exactly by one courier. Constraints (10) show the arriving time relationship for adjacent nodes along the routes; Constraints (11) indicate that the time couriers leave the restaurant should not be earlier than the time when meals are prepared well. If couriers arrive at restaurants in advance, they would wait until meals are ready; Constraints (12) ensure that the arriving time of couriers should not be later than the latest delivery time customers can accept; Constraints (13) guarantee that for each order, the restaurant node is visited earlier than the customer node by the courier.

5 | SOLUTION ALGORITHM

In this study, a two-stage solution algorithm based on ALNS is proposed, as shown in Figure 5. The initial solution of the optimal number of couriers is generated in the first stage. A courier removal operator is used to continuously reduce the number of couriers used in the solution, and then use the ALNS algorithm to judge whether these couriers can complete the delivery task through a small number of iterations. In the second stage, the ALNS algorithm is used to solve the meal delivery routing problem. For the iterative process of the algorithm, the transfer station node is set as two virtual nodes with the same location to realize the cross-regional delivery on the premise of the couriers can only deliver orders in their corresponding regions. One virtual node is treated as a delivery node of the front part order, and the other virtual node is treated as a pickup node of the latter part order.

5.1 | Solution representation

The iterative process of the proposed algorithm consists of many order removal and insertion operations for the solutions, and thus the structure of solutions has a great impact on the efficiency of the algorithm and the representation of the

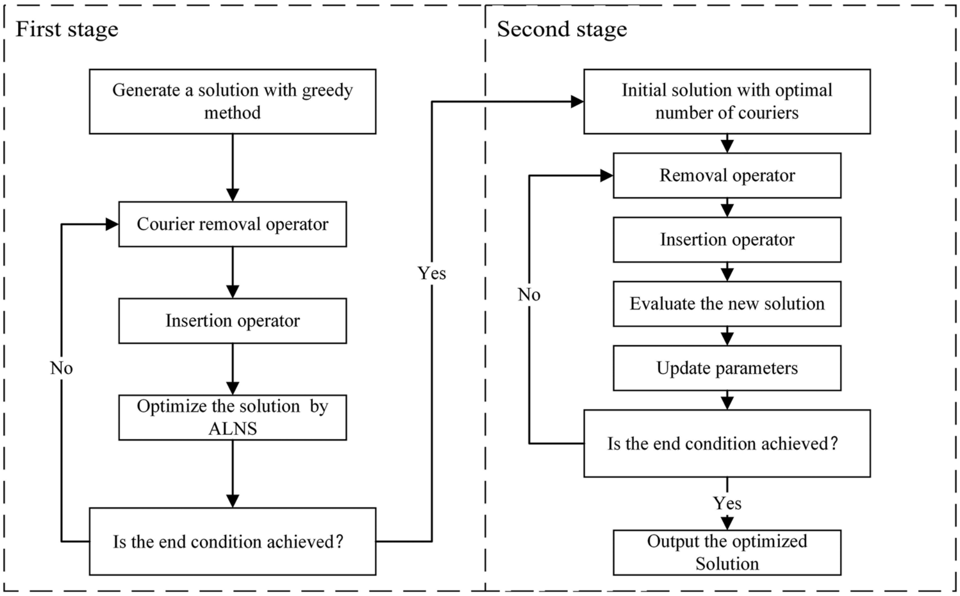


FIGURE 5 Flow chart of the two-stage algorithm

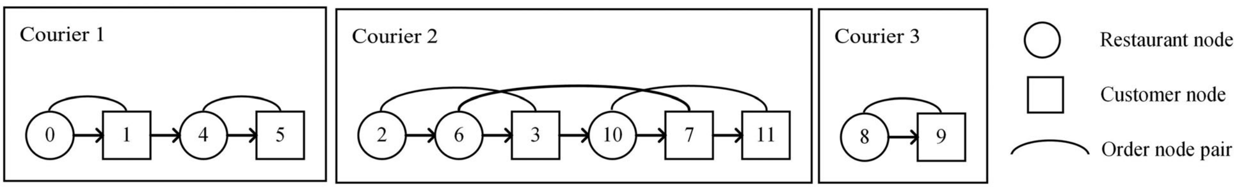


FIGURE 6 Solution representation

problem. In this study, a solution is encoded by using the odd-even encoding method. To present the structure of solutions more intuitively, an example of a routing solution for a region is given, as shown in Figure 6. There are 6 orders served by 3 couriers for this region. Odd numbers represent customer nodes, while even numbers represent restaurant nodes. The restaurant node and customer node of an order are represented by two consecutive even and odd numbers, respectively. The optimal route of Courier 1 is “0-1-4-5”, and the odd-even node pair “0-1” and “4-5” represent different meal orders, respectively. Couriers can visit the restaurant or customer nodes of different orders in mixed sequences, such as the route of Courier 2.

The solution represents how to assign the orders to the couriers in the corresponding region, and plan the route of each courier to deliver the orders. Different from the models of classical VRP, the route of a courier does not start from the depot but starts from the pickup (restaurant) node of the first order, nor ends at the depot, but ends at the delivery (customer) node of the last order. The reason for designing this solution structure is that considering the real situation of the O2O meal delivery system, couriers usually do not start or end at a unified depot, and the locations where the courier starts and ends are relative to the restaurant node and customer node of the order. The solutions of different regions are iteratively calculated separately. The final route optimization solution is acquired by combining the solutions of different regions.

5.2 | Generate initial solution

The process of generating the initial solution with the optimal number of couriers is described in this section. First, the greedy algorithm considering the couriers' compensation cost is used to find a solution with the estimated number of couriers. Then, the courier removal operator is used to reduce the number of couriers used and the insertion operator is used to insert the reduced courier's orders into other couriers' paths. Then, the ALNS algorithm is iterated a small number of times, and after iteration, whether the solution is feasible depends on whether the couriers can complete the delivery within the acceptable time of customers. Finally, the feasible solution of the optimal number of couriers is selected as the initial solution and also the input of the ALNS algorithm in the next stage.

The process of generating the initial solution is shown in Algorithm 2. The path set of all the dispatchable couriers is

ALGORITHM 2 Generating initial solution

Initialize: K: generate M empty paths, and M is large enough;

Repeat:

$n \leftarrow$ select one of the orders to optimize;

$bestposR \leftarrow$ select one position in K_m randomly as the position to insert node r;

$bestposC \leftarrow bestposR + 1$, insert node c after node r;

$\Delta Mincloss(K_{mn}), \Delta Min loss(K_{mn}) \leftarrow$ calculate the change of loss and compensation loss after inserting order n at $bestposR$ and $bestposC$ in K_m;

Repeat:

$posR \leftarrow$ select a new position in K_m;

Repeat:

$posC \leftarrow$ select a position after $posR$;

$\Delta closs(K_{mn}), \Delta loss(K_{mn}) \leftarrow$ calculate the change of loss and compensation loss after inserting node r at $bestposR$ and node c at $bestposC$;

If $\Delta Mincloss(K_{mn}) + \Delta Min loss(K_{mn}) < \Delta loss(K_{mn}) + \Delta closs(K_{mn})$ **Then**:

$bestposR \leftarrow posR$;

$bestposC \leftarrow posC$;

$\Delta Mincloss(K_{mn}) \leftarrow \Delta closs(K_{mn})$;

$\Delta Min loss(K_{mn}) \leftarrow \Delta loss(K_{mn})$;

End if

Until all positions after $posR$ have been tried;

Until: all positions in K_m have been tried;

$K_m \leftarrow$ insert node r at $bestposR$ and node c at $bestposC$ into K_m;

Until the path set K contains all the orders;

$s \leftarrow$ select all non-empty paths in K to construct a solution;

$S \leftarrow$ record this feasible solution s to solution set S;

Repeat:

$io \leftarrow$ Select an insertion operator by roulette based on the current weights of insertion operators;

$s \leftarrow io(cro(s))$ use courier removal operator cro to remove a path with least orders, and use io to insert these orders;

Repeat:

$s \leftarrow ALNS(s)$ use ALNS to iterate s;

Until reach the maximum number of iterations or s becomes a feasible solution;

If s is a feasible solution **Then**:

$S \leftarrow$ record this feasible solution s to solution set S;

End if

Until s is not a feasible solution;

$s \leftarrow$ select a solution of the optimal number of couriers from S;

Output s.

set as K. The path of courier m is denoted by K_m ($K_m \in K$). $\Delta loss(K_{mn})$represents the change of dispatching loss after the insertion of order n into the path K_m. $\Delta closs(K_{mn})$represents the change of couriers' compensation cost after the insertion of order n into the path K_m which is determined by the number of non-empty paths in K. The $\Delta closs(K_{mn})$ is much bigger than the $\Delta loss(K_{mn})$ on the premise that the generated solution is feasible. If there is an order in K_{mn} timed out too long that makes K_{mn} not feasible, $\Delta loss(K_{mn})$ is much bigger than $\Delta closs(K_{mn})$. In this way, the number of couriers used can be reduced on the premise that the generated solution is feasible. Each order has a restaurant node r and a customer node c. For each path K_m, node r must be before node c. First, generate an empty path K_m. Second, one order is randomly selected from the order set N and inserted into K_m, to minimize $\Delta loss(K_{mn})+\Delta closs(K_{mn})$. Repeat the above process until all orders are inserted into the path. Third, select all non-empty paths in K to construct a solution and record it. Then, the courier removal operator and insertion operator are used to construct a solution with fewer couriers. The courier removal operator can remove a path with the minimum number of orders in the solution and the insertion operator can insert these orders into other couriers' paths. The ALNS is used to optimize the solution until the maximum number of iterations is reached, or the solution is feasible. Repeat the above process until the solution is no longer feasible. Finally, select a solution of the optimal number of couriers in the solution set S as the initial solution s in the next stage.

5.3 | Adaptive large neighbourhood search algorithm

The mixed-integer programming model for the proposed meal instant delivery routing problem with order allocation strategy is solved by the heuristic solution algorithm based on ALNS, as shown in Algorithm 3.

A variety of removal operators and insertion operators are utilized, and the probability of each operator being adopted depends on its performance in all previous iteration groups, to achieve the purpose of self-adaptation. Each operator has a corresponding weight π_i, which is used to calculate the probability of its adoption in the next iteration. i represents the corresponding operator number. Through roulette, operator i is selected according to the probability $\pi_i / \sum_{j=1}^{m} \pi_j$. m is the total number of operators removed or inserted, which is set as $m = 3$ for both removal operators and insertion operators. At the beginning of the iteration, the weight of each operator is set to 1. During the iteration, the weight is updated according to the following rules: If the new solution is better than the optimal solution, the operator weight increases by σ_1; If the new solution is better than the current solution, the operator weight increases by σ_2; If the new solution is inferior to the current solution, but is accepted by the simulated annealing criterion, the operator weight increases by σ_3.

To avoid the search process falling into the local optimal solution, when $loss(x') > loss(x)$, the judgment criterion of the simulated annealing algorithm is adopted, and the acceptance probability of the new solution x' is set as $e^{-(loss(x')-loss(x))/T}$. After the reconstruction of a solution, the score of the operator is updated. Equation (14) shows how to calculate the operator weight at the $(j+1)^{th}$ iteration for operator i. l_{ij} represents the number of times operator i is used at the j^{th} iterations. γ represents the update speed of operator weight. The closer the value of γ is to 1, the faster the weight update of the operator; The

ALGORITHM 3 ALNS algorithm for the meal instant delivery routing problem

Initialize: s: initial solution/current solution; α: cooling rate; T_{start}: initial temperature; π_i: weights of operators; ω_i: operator scores.

$s_{best} \leftarrow s$, $T \leftarrow T_{start}$, $iter \leftarrow 1$;

Repeat:

$s_{iter} \leftarrow s$;

$q_{iter} \leftarrow$ The number of orders to be removed;

$ro \leftarrow$ Select a removal operator by roulette based on the current weights of removal operators;

$io \leftarrow$ Select an insertion operator by roulette based on the current weights of insertion operators;

$s' \leftarrow io(ro(q_{iter}, s))$ Use ro to remove q_{iter} orders from s and use io to generate the new solution s';

If $loss(s') < loss(s_{best})$ **Then**:

$s_{best} \leftarrow s \leftarrow s'$, scores of operators used are increased by σ_1;

Else if $loss(s') < loss(s)$ **Then**:

$s \leftarrow s'$, scores of operators used are increased by σ_2;

Else if $loss(s') > loss(s)$ but s' is accepted by simulated annealing criterion **Then**:

$s \leftarrow s'$, scores of operators used are increased by σ_3;

End if

$T \leftarrow T * \alpha$;

$iter \leftarrow 1 + iter$;

Update π_i using the adaptive weight adjustment procedure;

IF $iter\%y == 0$ **Then**:

$T \leftarrow T_{start}$, reset π_i;

End if

Until Reach the maximum number of iterations;

Output s_{best}.

ALGORITHM 4 The generic structure of the removal operator

Initialize: s: current solution;

Repeat:

$K_m, posR, posC \leftarrow$ select the positions of an order's restaurant node and customer node in the path K_m from the current solution s with each operator's specific rules;

$s \leftarrow$ remove nodes at $posR$ and $posC$ in the path K_m in s;

record the removed order n;

Until q_{iter} orders have been removed;

Output: partially destroyed solution s, a set of orders that have been moved out;

closer the value of γ is to 0, the slower the weight of the operator updates.

$$\pi_{i,j+1} = \begin{cases} \pi_{ij}, l_{ij} = 0 \\ \pi_{ij}(1-\gamma) + \gamma \dfrac{\omega_i}{l_{ij}}, l_{ij} \neq 0 \end{cases} \tag{14}$$

ALGORITHM 5 The generic structure of the insertion operator

Initialize: partially destroyed solution s, a set of orders that have been moved out;

Repeat:

$K_m, posR, posC \leftarrow$ select two positions in the path K_m from the partially destroyed solution s with each operator's specific rules;

$s \leftarrow$ insert one order's restaurant node at $posR$ and customer node at $posC$ in the path K_m in s;

Until all the orders have been inserted into the solution;

Output a new solution s'.

5.4 | Removal and insertion operators

Three removal operators and three insertion operators are used. The removal operator is a function that partially removes q_{iter} orders from the current solution s. The insertion operator is to insert the orders removed by the removal operator into the solution according to certain rules, and get s' as the new solution. The generic description of the removal operator and insertion operator is shown in Algorithms 4 and 5, respectively.

The selected operators are explained below:

Random removal operator: Select and remove q_{iter} orders from s randomly. This operator is mainly used to increase the diversity of solutions.

Greedy removal operator: Select and remove one order from the solution which increases the loss most. Repeat q_{iter} times until q_{iter} orders have been removed.

Shaw removal operator: This operator can remove similar orders in a path and increase the diversity of solutions [25]. First, choose an order in a courier's path randomly and then calculate the similarity of the orders in the same path. Then, select an order that is most similar to the order chosen before and remove it from the path. Repeat q_{iter} times until q_{iter} orders have been removed.

Random insertion operator: Randomly select the path and randomly select the position in the path to insert one order from the q_{iter} orders that have been removed before and repeat q_{iter} times until all the removed orders have been inserted.

Greedy insertion operator: Similar to the process of constructing the initial solution, select one of the removed orders and insert it into the path that minimizes the change of the total loss. Repeat the above process until all the removed orders are inserted into the path.

Max regret insertion operator: Use regret value $\Delta loss(K_{mn_1}) - \Delta loss(K_{mn_2})$, which represents the difference between inserting the order n into the position with the lowest cost and the location with the lowest cost in the path K_m. Insert all the removed orders into the path according to the principle of maximizing regret value.

6 | COMPUTATIONAL RESULTS

In this paper, the ALNS algorithm is implemented in Java, and it runs on the Windows computer with Ryzen 5 2500U CPU

TABLE 3 Meal order information

Restaurant node location	Customer node location	Confirming time	Pick-up time	Commitment time
(121.479636,39.072151)	(121.505954,39.087651)	11:55:42	12:14:30	12:35:00
(121.479636,39.072151)	(121.506874,39.088031)	11:52:41	12:08:59	12:45:00
(121.483951,39.064602)	(121.479314,39.065451)	11:19:41	11:37:28	12:03:27

2.0GHz and 8G RAM. Test instances of different sizes are constructed based on the real-life data of the Fengniao instant delivery platform (https://fengniao.ele.me/). First, a certain number of long-distance orders across different delivery regions are generated and added to the normal real-life orders to construct new test instances. Then, the cross-regional long-distance orders are split according to the proposed order transfer strategy. Finally, the routes of couriers are optimized to serve the normal orders and new spilt orders within each region.

6.1 | Test instance construction

Test instances are constructed based on real delivery platform data, which can reflect the spatio-temporal distribution of service demands and couriers' routing behaviours. The spatial position of each node is described by latitude and longitude. The time when one order is created by the customer is defined as the confirming time, the time when one restaurant prepares the meal well for couriers to take away is defined as pickup time, and commitment time is the latest time without any incurring overtime loss. Take three meal orders as an example, as shown in Table 3.

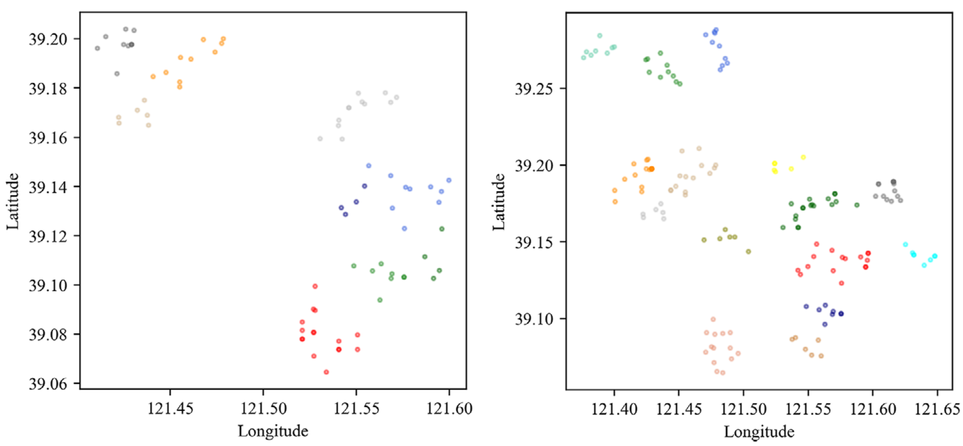
6.1.1 | Order combination results

Two different test instances are constructed based on the real platform order delivery data, which contain 40 and 80 normal orders, respectively. The original regions are divided into a series of delivery regions based on the locations of restaurants and customers included in the orders of the test instances based on the DBSCAN algorithm. The 80 points of test instance with 40 normal orders are clustered to generate 9 clusters marked by different colours, and each cluster can be regarded as a delivery region, as shown in Figure 7a. The 160 points of the test instance with 80 normal orders are clustered to generate 15 clusters, as shown in Figure 7b.

6.1.2 | Generate cross-regional long-distance orders

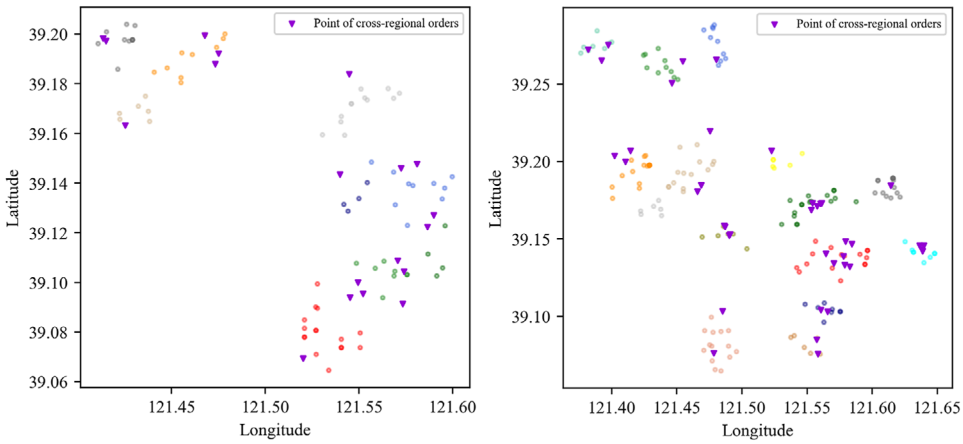
The customer nodes and restaurant nodes of cross-regional long-distance orders are generated randomly based on the clusters of normal orders. Firstly, a large number of candidate nodes are randomly generated in each region, and then the candidate nodes in different regions are randomly combined to generate candidate cross-regional long-distance orders. Then, the candidate orders with the distance between the restaurant node and the customer node within 5 to 10 km are selected, accounting for 20% of the total orders. The lower bound 5km is the distance threshold of the order splitting strategy. The upper bound 10km is estimated from the maximum value between adjacent regions for these two test instances, which is set to ensure that the location of customers and restaurants for the generated cross-regional long-distance orders exist in two adjacent regions, rather than crossing multiple regions.

Ten cross-regional orders are added for the test instance with 40 normal orders and 20 cross-regional orders are added for



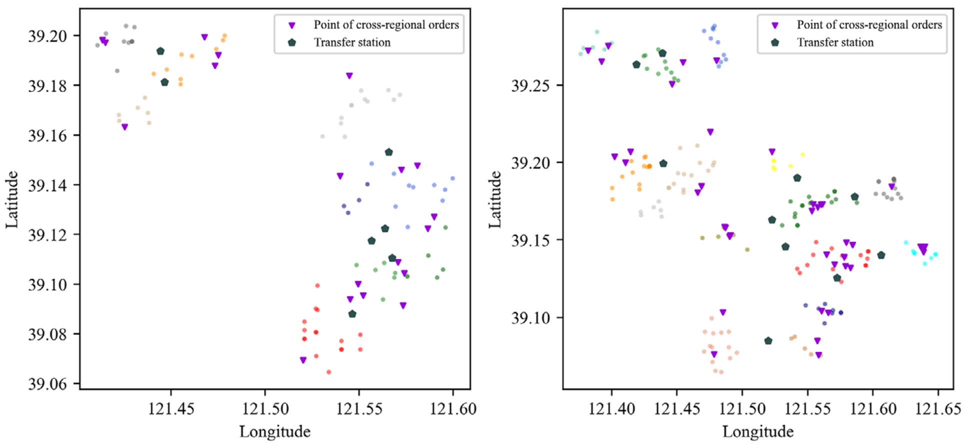
(a) 40 orders (b) 80 orders

FIGURE 7 Order combination results of normal orders for different test instances



(a) 50 orders (b) 100 orders

FIGURE 8 Test instances with cross-regional orders



(a) 50 orders,7 transfer station (b) 100 orders,10 transfer station

FIGURE 9 Location of transfer stations for different test instances

the test instance with 80 normal orders. Then, these two test instances consist of 50 and 100 orders, respectively, as shown in Figure 8.

6.2 | Locations of transfer stations and order splitting results

The location of transfer stations is determined by the cross-regional long-distance orders. If there are multiple orders across two delivery regions, the location of the transfer station is calculated according to Equations (1) and (2). The distance threshold value is set as 5 km. 7 and 10 transfer stations need to be set for these two test instances, and their locations are as shown in Figure 9.

After confirming the location of transfer stations, the cross-regional orders are split into two parts through the proposed order splitting strategy. The splitting results of these two test instances are shown in Table 4. Tables 5 and 6 show the number of orders and couriers in each cluster for the instance with 50 orders and 100 orders, respectively. The number of couriers in each cluster is determined by Algorithm 2. The distributions of the confirming time, pick-up time, and commitment time for the orders of two test instances are visualized in Figures 10 and 11, respectively, which are concentrated in the peak period.

6.3 | Meal delivery routing optimization

6.3.1 | Parameter setting

The parameters of the ALNS algorithm have a great impact on its performance. For common parameters, we determine them according to the scale of the problem. We set the initial temperature as $T_{start} = \frac{0.05Loss(s_{init})}{ln2}$, which means, in the first iteration, the probability that a solution that is 5% weaker than the initial solution is accepted is 50%. The cooling rate is set to $\alpha = 0.9975$ The parameters for calculating overtime loss $Loss(t_c)$ are set as $L_c = 0$, $\Delta t_1 = 5$ min, $\Delta\ t_2 = 10$ min, $\Delta t_3 = 15$ min, $\rho_1 = 1$, $\rho_2 = 10$, $\rho_3 = 100$.

TABLE 4 Order splitting results

Test instance	Total number of cross-regional orders	Average distance of cross-regional orders (km)	Total number of transfer stations	Total number of orders after splitting
50 orders	10	5.95	7	60
100 orders	20	5.93	10	120

TABLE 5 Number of orders and couriers in each cluster for the instance with 50 orders

Cluster number	Number of orders after splitting	Number of front/latter part orders after splitting	Number of couriers
1	7	2	2
2	8	3	2
3	4	1	2
4	7	1	2
5	10	6	3
6	3	1	1
7	8	3	3
8	3	1	1
9	10	2	3

TABLE 6 Number of orders and couriers in each cluster for the instance with 100 orders

Cluster number	Number of orders after splitting	Number of front/latter part orders after splitting	Number of couriers
1	8	1	2
2	11	3	3
3	10	3	3
4	3	2	1
5	7	5	2
6	14	0	4
7	6	2	2
8	6	7	2
9	15	2	4
10	9	2	2
11	5	1	1
12	7	4	3
13	4	1	1
14	8	4	2
15	7	3	3

For hyper-parameters, we adjust them through a 100 nodes test instance by referring to the setting of parameters in [16, 26, 27]. Important parameters used in the experiment are shown in Table 7. Parameter n_{max} is the maximum number of iterations. The larger n_{max} is, the more likely it is to find the optimal

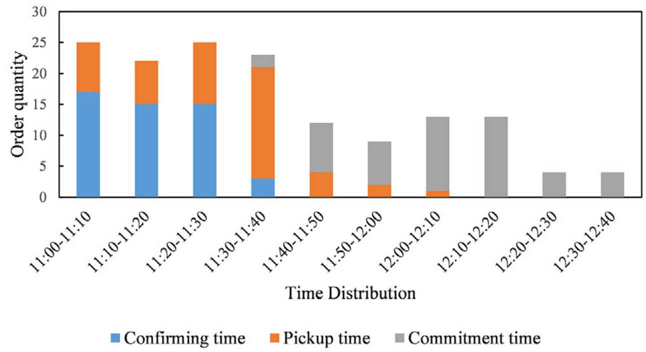


FIGURE 10 Order time distribution for the instance with 50 orders

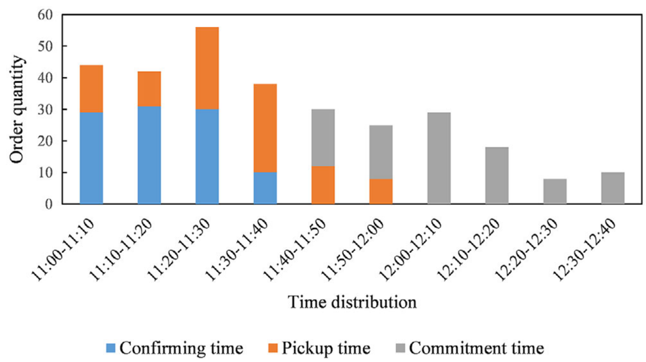


FIGURE 11 Order time distribution for the instance with 100 orders

TABLE 7 Parameter setting for ALNS algorithm

Symbol	Role	Value
δ_l	The lower limit for δ	0.15
δ_u	The upper limit for δ	0.25
φ_1	First shaw parameter	0.5
φ_2	Second shaw parameter	0.25
φ_3	Third shaw parameter	0.15
φ_4	Fourth shaw parameter	0.25
ω	Initial temperature control parameter	0.05
α	Cooling rate	0.99975
σ_1	Operator score increment case 1	1.5
σ_2	Operator score increment case 2	1.2
σ_3	Operator score increment case 3	0.8
γ	Operator weight change parameter	0.5
n_{max}	Maximum number of iterations	30,000

TABLE 8 Impacts of parameter n_{max} on the performance of the ALNS algorithm

n_{max}	5000	10,000	15,000	20,000	25,000	30,000	35,000	40,000
Best loss (min)	675.20	654.60	664.55	666.01	668.96	659.38	661.56	670.38
Average loss (min)	684.71	677.94	678.21	675.13	674.54	671.51	676.34	675.33
Average time cost (s)	2.34	4.48	6.67	8.73	11.16	12.46	15.53	17.38

TABLE 9 Impacts of parameter δ on the performance of the ALNS algorithm

		δ_u				
		0.15	**0.20**	**0.25**	**0.30**	**0.35**
δ_l	0.05	601.78	601.91	605.85	599.78	605.35
	0.1	607.22	607.18	605.12	606.78	603.83
	0.15	—	601.93	**596.75**	598.38	610.22
	0.2	—	—	604.68	607.07	604.48

TABLE 10 Numerical experimental results

Test instance	Loss (min)	Total time cost (min)	Total overtime cost (min)
50 orders	1081.17	1066.80	14.37
100 orders	2154.28	2099.40	54.88

solution, but if n_{max} is too large, it will affect the efficiency of the algorithm. Therefore, we test the impacts of different values of n_{max} on the performance of the ALNS algorithm. The best loss, average loss, and average time spent in the experiment are used to measure the effect of parameters. As shown in Table 8, when $n_{max} = 30{,}000$, the best loss and average loss are better than when $n_{max}<30{,}000$, and when $n_{max} = 35{,}000$, there is no obvious solution improvement, and thus, $n_{max} = 30{,}000$ is selected.

Parameter δ represents the proportion of the orders removed to the total number of orders when using the removal operator once, which has a great impact on the iterative process and speed of the algorithm and will be randomly selected within a range $[\delta_l, \delta_u]$. The larger the δ, the higher the proportion of solution removal in each iteration. Too low δ will slow the neighbourhood search process and affect the quality of the solution, but too high δ will slow the solution process and also affect the quality of the solution. As shown in Table 9, the loss of the optimal solutions with different ranges for δ in the iterative process are compared after ten experiments and we choose [0.15,0.25] as the final range.

6.3.2 | Optimization results

The optimization results for these two test instances are shown in Table 10. The proportion of overtime loss is quite low, indicating that the overtime delivery of orders is less frequent for the test instances. To some extent, it reflects the rationality of the order allocation strategy based on the transfer station, which can widen the delivery scopes without significantly delaying the order delivery time.

The convergence processes of the ALNS algorithm for two instances are shown in Figure 12a and b, respectively. It can be obtained by adding the loss values of each iteration in the iterative process of each region. It can be seen that the solution has been significantly improved in the first 10,000 iterations. After 10,000 iterations, the solution continues to be improved slightly with the iterations, and the solution tends to converge after 25,000 iterations. The total running time of 50 orders is 7.214 s, and that of 100 orders is 20.332 s. They can be obtained by adding the running time of the iterative process for each region.

The optimized meal delivery routes of couriers for two instances are shown in Figures 13a and b, respectively. It can be seen that couriers only serve the customers within the specific region they belong to. The meals of cross-regional long-distance orders are transferred at the intermediate points which are the transfer stations. The cross-regional long-distance orders are split into normal orders, which are delivered by the couriers in the corresponding regions.

7 | ECONOMIC BENEFIT ANALYSIS

In this section, the rationality of the platform's investment in the transfer stations is explained by analysing the possibility of profit from setting up a transfer station. The additional costs brought by the establishment of transfer stations are mainly reflected in: (1) the construction, maintenance and operation of transfer stations; (2) the extra delivery cost of the orders delivered through transfer stations. The benefits of setting up transfer stations include: (1) providing customers with low-cost cross-regional delivery service, and stimulating more potential orders; (2) improving the service efficiency of couriers working in their familiar region. To simplify the problem, some assumptions are made for the mathematical model of economic benefit analysis as follows:

1. For a delivery region, the numbers of orders using special delivery and transfer station-based delivery are both proportional to the number of normal orders.
2. After the implementation of transfer station-based delivery, some customers who originally use special delivery continue to use special delivery, and others turn to transfer station-based delivery.

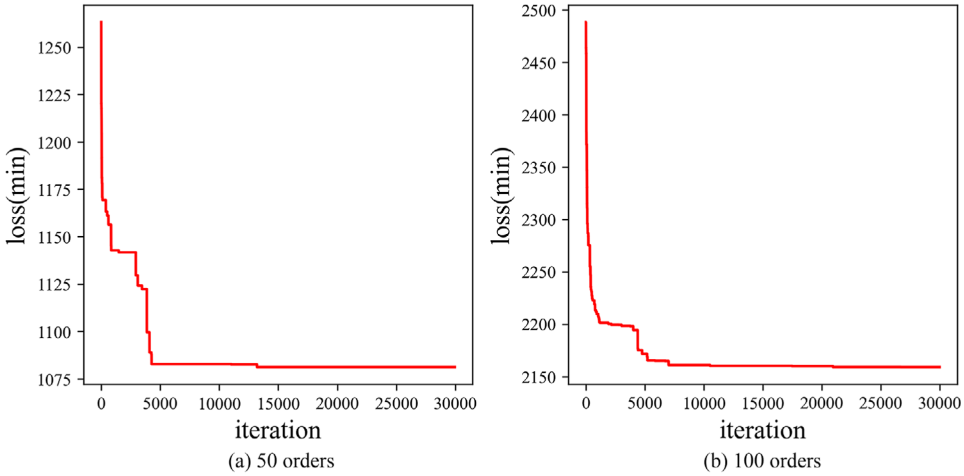


FIGURE 12 Convergence process of the ALNS algorithm for different test instances

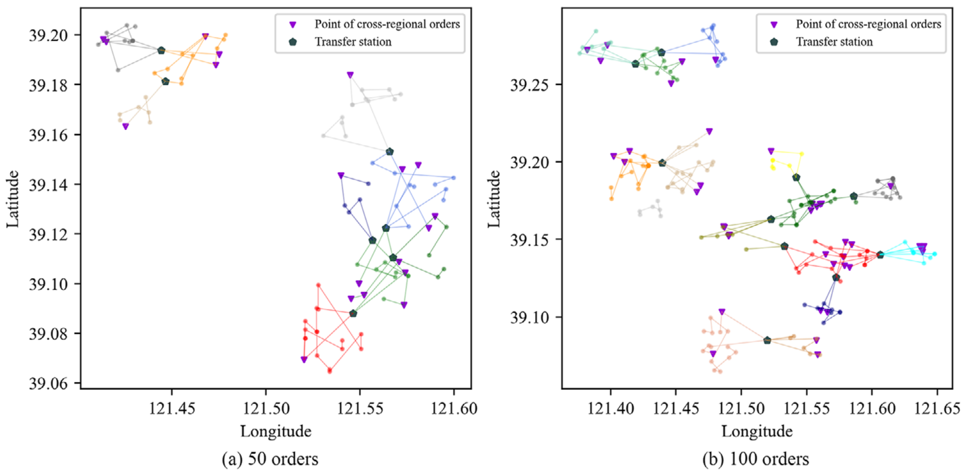


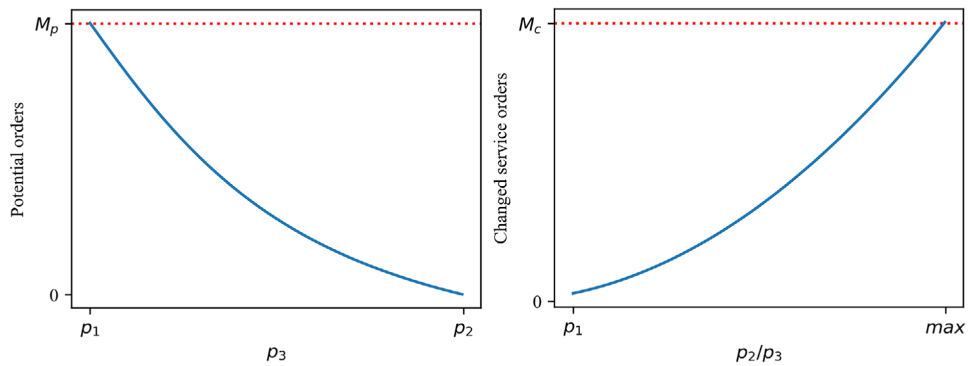
FIGURE 13 Optimized meal delivery routes for different test instances

3. The quantity of orders is affected by the service prices of the different delivery modes.
4. The total cost of the O2O platform is fixed, and the average profits of different kinds of orders are in direct proportion to its service price.

For the traditional delivery service strategy, cross-regional long-distance orders can only be delivered by special delivery. After adopting the order allocation strategy based on transfer stations, the cross-regional long-distance orders can be served through either special delivery or transfer station-based delivery. The time-sensitive customers can continue to use the special delivery with a higher price, while the price-sensitive customers can choose the delivery service based on the transfer station with a lower price. The transfer station-based delivery service would stimulate the potential demand of price-sensitive customers due to its lower price.

X represents the number of normal orders. Y_1 represents the number of special delivery orders under traditional delivery mode. Y_2 represents the number of special delivery orders under the delivery mode based on transfer stations. Z represents the number of transfer station-based delivery orders under the delivery mode based on transfer stations. p_1, p_2, p_3 represent the average service prices of normal order delivery, special delivery, and transfer station-based delivery respectively, satisfying $p_1 < p_3 < p_2$. M_p is the maximum number of potential cross-regional long-distance orders. M_c is the maximum number of changed service orders, which represents the number of orders that use the transfer station-based delivery instead of special delivery after adopting the delivery mode based on transfer stations. S_1 is the total profit of the traditional delivery mode, as shown in Equation (15). S_2 is the total profit of the delivery mode based on transfer stations, as shown in Equation (16). I is the total increased profit margin after adopting the delivery mode based on transfer stations, as shown in Equation (17).

$$S_1 = p_1 * X + p_2 * Y_1 \tag{15}$$



(a) Potential orders (b) Changed service orders

FIGURE 14 Relationship between the order quantity and the average price of delivery services

$$S_2 = p_1 * X + p_2 * Y_2 + p_3 * Z \tag{16}$$

$$I = \frac{S_2 - S_1}{S_1} \tag{17}$$

Two types of customers are considered for the profit analysis. The first category is potential customers, who are price-sensitive. They need cross-regional long-distance orders, but special delivery is too expensive for them. The upper limit of these potential cross-regional long-distance orders M_p is affected by p_3, as shown in Figure 14a. The lower the price p_3, the more new orders would be stimulated. The second category is customers who change their services. They originally use the special delivery service. After the implementation of the delivery model based on the transfer station, they choose the new delivery service. The upper limit of these changed service orders M_c is affected by the ratio of p_2 to p_3, as shown in Figure 14b. The larger the ratio of p_2 to p_3, the more customers would change the delivery service mode.

Since the value of X can reflect the order quantity in a region, X is used as the basic unit of order quantity. To simulate an assumed specific situation, the related parameters are set as $M_p = 0.2X$, $M_c = 0.05X$, $4p_1 \le p_2 \le 6p_1$, $p_1 \le p_3 \le 4p_1$, $Y_1 = 0.1X$. The values of Z and Y_2 are calculated based on these parameters. The numerical diagram of Z and Y_2 varying with the average profit of delivery services are shown in Figures 15a and b, respectively. The parameters can be set as different values according to realistic situations. The purpose of this example is to show the specific impact of the transfer station on the platform profitability in a specific situation.

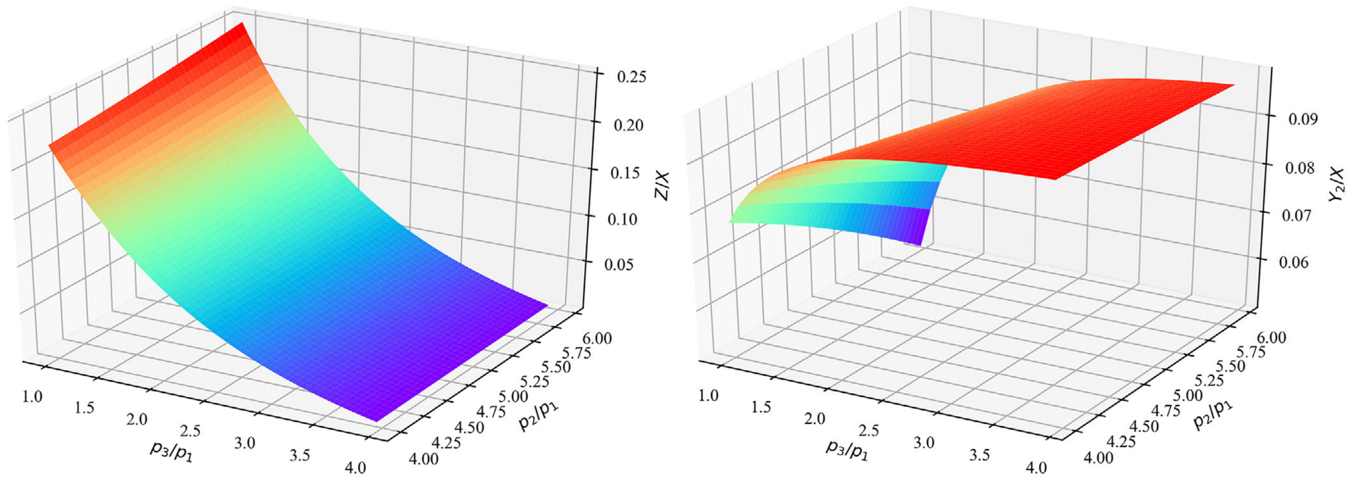
The numerical diagram of I varying with p_2 and p_3 is shown in Figure 16, reflecting the variation of the total increased profit margin after adopting the delivery mode based on transfer stations with different delivery pricing schemes. The warmer the colour, the bigger the value, and the colder the colour, the smaller the value. In this example, the platform has a wide adjustment range of pricing schemes to ensure that the transfer station can be significantly profitable. With the best pricing scheme, the maximum value of the total increased profit margin in this region is about 12%. To gain more profits, the platform will not choose the pricing schemes that reduce the profit margin or the effect is not obvious but will choose the pricing strategy that increases the profit margin the most. In practice, the improvement of profit margin is affected by many factors and the service providers need to investigate the demands of customers and design reasonable prices for different delivery services. This model only considers the different prices of different service modes and the change in order quantity caused by different prices. The above analysis shows that setting up a transfer station may improve the profits of the platform with a high possibility. For different regions, the platform can choose whether to establish transfer stations to improve profits according to the order demands of customers and the construction cost of transfer stations.

8 | CONCLUSIONS

This study has investigated the meal delivery routing problem for the O2O instant logistics services. We propose a three-stage research framework consisting of order combination, splitting, and meal delivery routing. Normal orders are combined by the DBSCAN algorithm based on the locations of restaurants and customers. The orders for each cluster are served by specific couriers with higher familiarity, contributing to improving the service efficiency of delivery systems. To overcome the limitation of delivery distance for instant logistics systems, a novel order splitting strategy based on transfer stations is proposed to handle cross-regional long-distance orders.

We construct a mixed-integer programming model considering the overtime loss, and develop a heuristic algorithm based on ALNS. Test instances of different sizes are constructed based on the real-life order data of instant delivery systems. The experiment results demonstrate that the ALNS algorithm converges quickly, and the order allocation strategy based on the transfer stations can expand the delivery scopes of couriers effectively, and satisfy the diverse customer demands.

In the future, considering the time-varying orders, we will design a spatio-temporal clustering algorithm for order combination and develop a modelling framework for the dynamic meal delivery routing. More strict constraints will



(a) Number of Z (b) Number of Y_2

FIGURE 15 Variation of the order quantity

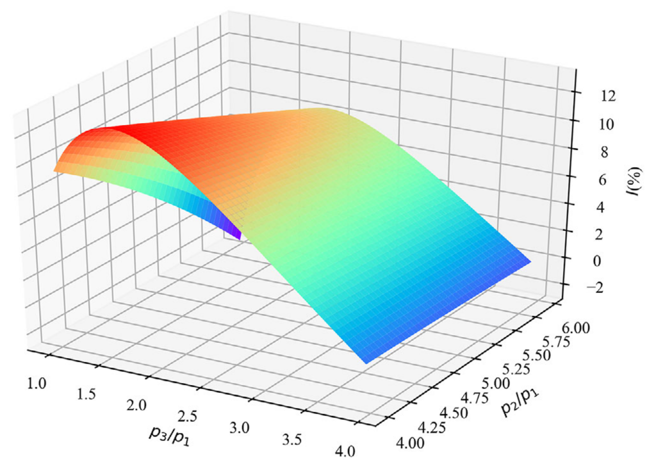


FIGURE 16 Variation of the total increased profit margin

be added to the meal delivery problem, such as the workload balance of couriers, heterogeneous vehicle capacity, etc. Furthermore, more online order consolidation and splitting strategies will be considered for the instant logistics service.

ACKNOWLEDGEMENTS

This work was supported by the Research Innovation Fund for College Students of Beijing University of Posts and Telecommunications, the National Natural Science Foundation of China (No.72001029), and the Research Start-up Fund for Introduction of Talents of Beijing University of Posts and Telecommunications.

CONFLICT OF INTEREST

The authors have declared no conflict of interest.

DATA AVAILABILITY STATEMENT

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

ORCID

Senyan Yang https://orcid.org/0000-0001-7524-784X

REFERENCES

1. Chen, Q., Zhang, Y., Chen, L.: A study of internet development and enterprise financing in China. Networks Spatial Econ. 21(3), 495–511 (2021)
2. Zhao, X., Lin, W., Cen, S., Zhu, H., Duan, M., Li, W., Zhu, S.: The online-to-offline (O2O) food delivery industry and its recent development in China. Eur. J. Clin. Nutr. 75(2), 232–237 (2021)
3. Choi, Y., Zhang, L., Debbarma, J., Lee, H.: Sustainable management of online to offline delivery apps for consumers' reuse intention: Focused on the meituan apps. Sustainability 13(7), 3593 (2021)
4. Liu, S., He, L., Max Shen, Z.J.: On-time last-mile delivery: Order assignment with travel-time predictors. Manage. Sci. 67(7), 4095–4119 (2020)
5. Fotouhi, H., Mori, N., Miller-Hooks, E., Sokolov, V., Sahasrabudhe, S.: Assessing the effects of limited curbside pickup capacity in meal delivery operations for increased safety during a pandemic. Transp. Res. Rec. 2675(5), 436–452 (2021)
6. Meituan Research Institute and Takeout Professional Committee of China Hotel Association: Development Report of China's Food Delivery Industry in 2019 and the First Half of 2020 (2020). https://mri.meituan.com/research/reportm
7. Reyes, D., Erera, A., Savelsbergh, M., Sahasrabudhe, S., O'Neil, R.: The meal delivery routing problem. Optimization Online, 6571 (2018)
8. Yildiz, B., Savelsbergh, M.: Provably high-quality solutions for the meal delivery routing problem. Transport. Sci. 53(5), 1372–1388 (2019)
9. Wang, Z., Li, T., Yue, C.: Online scheduling for the instant delivery problem in a city based on multiple prediction scenarios. Syst. Eng. Theory Pract. 38(12), 3197–3211 (2018)
10. Liu, Y.: An optimization-driven dynamic vehicle routing algorithm for on-demand meal delivery using drones. Comput. Oper. Res. 111, 1–20 (2019)
11. Ulmer, M.W., Thomas, B.W., Campbell, A.M., Woyak, N.: The restaurant meal delivery problem: Dynamic pickup and delivery with deadlines and random ready times. Transp. Sci. 55(1), 75–100 (2021)
12. Liao, W., Zhang, L., Wei, Z.: Multi-objective green meal delivery routing problem based on a two-stage solution strategy. J. Cleaner Prod. 258, 120627 (2020)
13. Steever, Z., Karwan, M., Murray, C.: Dynamic courier routing for a food delivery service. Comput. Oper. Res. 107, 173–188 (2019)
14. Battarra, M., Cordeau, J.F., Iori, M.: Chapter 6: 'Pickup-and-Delivery Problems for Goods Transportation. Vehicle Routing: Problems, Methods,

and Applications, Second Edition. Society for Industrial and Applied Mathematics, pp. 161–191 (2014)
15. Savelsbergh, M.W., Sol, M.: The general pickup and delivery problem. Transp. Sci. 29(1), 17–29 (1995)
16. Ropke, S., Pisinger, D.: An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. Transp. Sci. 40(4), 455–472 (2006)
17. Keçeci, B., Altıparmak, F., Kara, I.: A mathematical formulation and heuristic approach for the heterogeneous fixed fleet vehicle routing problem with simultaneous pickup and delivery. J. Ind. Manage. Optim. 17(3), 1069 (2021)
18. Wang, S., Yang, L., Yao, Y., Zhang, Q., Shang, P.: Equity-oriented vehicle routing optimization for catering distribution services with timeliness requirements. IET Intell. Transport Syst. 16(2), 163–185 (2022)
19. Aziez, I., Côté, J.F., Coelho, L.C.: Exact algorithms for the multi-pickup and delivery problem with time windows. Eur. J. Oper. Res. 284(3), 906–919 (2020)
20. Kohar, A., Jakhar, S.K.: A capacitated multi pickup online food delivery problem with time windows: A branch-and-cut algorithm. Ann. Oper. Res. 1–22 (2021)
21. Voigt, S., Kuhn, H.: Crowdsourced logistics: The pickup and delivery problem with transshipments and occasional drivers. Networks 79(3), 403–426 (2021)
22. Wolfinger, D.: A large neighborhood search for the pickup and delivery problem with time windows, split loads and transshipments. Comput. Oper. Res. 126, 105110 (2021)
23. Masson, R., Lehuédé, F., Péton, O.: An adaptive large neighborhood search for the pickup and delivery problem with transfers. Transp. Sci. 47(3), 344–355 (2013)
24. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. KDD-96 Proc. 96(34), 226–231 (1996)
25. Shaw, P.: Using constraint programming and local search methods to solve vehicle routing problems. In: *International Conference on Principles and Practice of Constraint Programming*. Springer, Berlin, Heidelberg, pp. 417–431 (1998)
26. Demir, E., Bektaş, T., Laporte, G.: An adaptive large neighborhood search heuristic for the pollution-routing problem. Eur. J. Oper. Res. 223(2), 346–359 (2012)
27. Grangier, P., Gendreau, M., Lehuédé, F., Rousseau, L.M.: An adaptive large neighborhood search for the two-echelon multiple-trip vehicle routing problem with satellite synchronization. Eur. J. Oper. Res. 254(1), 80–91 (2016)

How to cite this article: Li, J., Yang, S., Pan, W., Xu, Z., Wei, B.: Meal delivery routing optimization with order allocation strategy based on transfer stations for instant logistics services. IET Intell. Transp. Syst. 16, 1108–1126 (2022). https://doi.org/10.1049/itr2.12206