A Memetic Algorithm for Vehicle Routing Problem with Fresh Food and Heterogeneous Customer Behavior

Yankai Zhang, Kaiqi Zhao, Na Liu, Shiyi Xu, Shiwei Liang, Wenxuan Shan*

School of Transportation Science and Engineering, Beihang University, Beijing, China

*shanwenxuan@buaa.edu.cn

Abstract. The online community group-buying of fresh products is becoming welcomed by the public. Providing customer fresh products with satisfied quality is concerned by community group-buying distributor, as residents have diverse requirements for delivery times and freshness, often overlooked in existing research. This paper proposes a delivery route optimization model tailored for online community group-buying, addressing the heterogeneous behavior of customers, multiple delivery time windows, and the freshness decline of various fresh products. We introduce a memetic algorithm for this non-linear programming problem, incorporating a Split algorithm to handle multi-commodity delivery and time-varying arc costs. A dynamic allocation scheme for delivery time is developed for optimal fixed sequences, allowing vehicles to wait within a reasonable timeframe. Experimental results demonstrate that our method achieves average reductions of 82.01% and 90.92% in comparison to classical algorithms for instances with 50 and 100 nodes, respectively. Additionally, the proposed algorithm with dynamic allocation shows an average objective function decrease of 0.44% compared to without dynamic allocation. The allowance for waiting during delivery enhances refrigeration conditions, offering significant managerial insights into maintaining product freshness.

Keywords: Vehicle routing problem, Perishability, Online community group-buying, Split delivery

1 Introduction

Online community group-buying has emerged as a primary method for acquiring daily fresh products. According to a report by the National Bureau of Statistics of China, the community group-buying market size is projected to reach 1301.74 billion RMB by 2024 [1][2]. The increasing demand and diverse requirements necessitate that online community group-buying distributors meticulously design their delivery services to meet the unique characteristics of fresh products and the evolving needs of residents.

The rapid freshness decline of perishable goods often results in residents receiving less fresh items, thereby diminishing their satisfaction. Consequently, designing a delivery solution that transports perishable goods within specified timeframes to enhance resident satisfaction is crucial for community group-buying. Residents exhibit diverse

G. Zhao et al. (eds.), *Proceedings of the 2024 7th International Symposium on Traffic Transportation and Civil Architecture (ISTTCA 2024)*, Advances in Engineering Research 241,
https://doi.org/10.2991/978-94-6463-514-0_59

needs, including preferences for product types, pickup times, and freshness levels. For instance, office workers typically retrieve purchases between 5 PM and 9 PM after work and exhibit less concern for freshness, while senior citizens prefer pickups between 10 AM and 2 PM and have higher freshness requirements. These considerations highlight the importance of tailored delivery solutions to accommodate the varied needs and preferences of community members in community group-buying endeavors.

Since pick-up time windows vary among heterogeneous customers, multiple time windows are considered in our study. Previous studies have addressed this aspect in various ways. Wang et al. (2018) [3] established a multi-objective VRP optimization model with mixed time windows and perishability. Zheng (2020) [4] developed a VRP model with multiple fuzzy time windows to minimize travel distance and distribution costs. Belhaiza et al. (2019) [5] proposed three multi-start data-driven evolutionary heuristics.

The freshness decline of fresh products is another critical consideration, as it directly affects customer satisfaction and incurs potential costs. Li and Li (2022) [6] described deterioration cost as a linear function of delivery time. Leng et al. (2020) [7] and others (Chen, 2021; Chen, 2023; Zhang, 2023; Yang, 2023) [8][9][10][11] modeled freshness decline over delivery time as a negative exponential function. Yang and Tao (2023) [11] linked freshness to customer satisfaction, dividing freshness decline into three stages. Deng et al. (2021) [12] considered temperature's effect on deterioration as different decline rates.

Traditional community group-buying approaches have investigated the vehicle routing problem with multiple time windows and addressed issues concerning freshness decline. We introduce a memetic algorithm to tackle the nonlinear programming challenges posed by these factors. This algorithm allows delivery vehicles to wait proactively after serving a community node to optimize delivery times.

2 Problem Description

In contrast to the door-to-door delivery typical of conventional e-commerce platforms, products in community group-buying are initially delivered to the group-buying leader's location. Customers then pick up their products from the leader at their convenience. The time gap between the delivery to leader and the subsequent pick-up by customers can lead to a further decline in quality of fresh products. Customers not only care about loss of freshness but also expect to collect their products within a convenient time interval, referred to in this study as pick-up time windows. If fresh products are delivered to the group-buying leader after a customer's pick-up time window, the customer may be unable to collect them that day, significantly increasing penalty costs.

Customers can typically be categorized into several types based on their features. Each type of customer has different pick-up time windows and requirements for freshness. Delivering products too early may result in prolonged storage at the group-buying leader's location, while delivering too late will miss customers' pick-up time windows. In community group-buying, customers demand multiple types of fresh products, but the capacity of the delivery vehicle and the types of products it can carry are limited.

Multiple delivery services may be required with different products delivered each time.

This study aims to determine the types of fresh products loaded on each vehicle and to optimize the routes of each vehicle to minimize the total cost, which includes transportation and penalty costs for breaches in freshness or pick-up time windows.

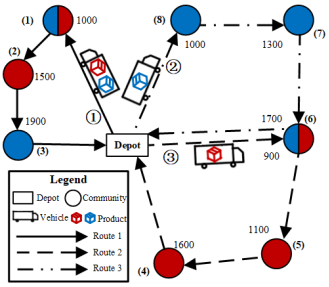


Fig. 1. Delivery Routes

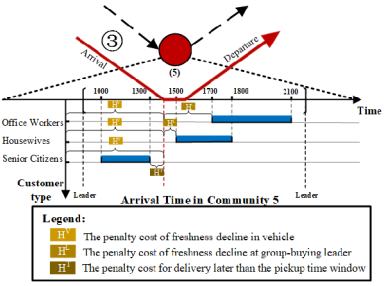


Fig. 2. The penalty cost at community 5

Figure 1 illustrates an example of online community group-buying involving two types of fresh products and three types of customers. Among the eight communities, Communities 1 and 6 require both types of fresh products. Community 1 is served only once by Route 1, while Community 6 must be served by Routes 2 and 3, each delivering only one type of product. Figure 2 demonstrates penalty cost at community 5. The horizontal axis is time, and vertical axis represents windows for retirees, housewives, and office workers. The pick-up time windows are 10:00 to 13:00, 15:00 to 18:00, and 17:00 to 21:00, respectively. For housewives and office workers, fresh products are delivered earlier than their pick-up time windows, resulting in freshness decline both within the vehicle and at the leader's location. For retirees, the fresh products are delivered later than their pick-up time window. They can receive fresh products once delivered, but the delayed delivery will incur higher return costs.

3 Mathematical Model

3.1 Assumptions

(1) The same type of fresh products in each community is served exclusively by a single vehicle.

(2) The transportation process is considered uniform, with no additional time consumption throughout the entire delivery process.

(3) The freshness of the delivered fresh products significantly influences residents' decisions to return the products, with other factors being disregarded.

3.2 Penalty Cost During Delivery

Fresh product delivery in online community group-buying entails additional considerations beyond basic transportation costs, which is divided into three components.

The perishable nature of fresh products leads to damage costs due to freshness decline during delivery. These costs are estimated using the first-order reaction dynamics model, as described by Hsu et al. (2007) [13]. Freshness decline correlates with both transportation time and storage duration at leader's location, which is defined as follows.

$$F_p^1(t) = e^{-\alpha_p t} \tag{1}$$

$$F_p^2(t) = e^{-\beta_p t} \tag{2}$$

$F_p^1(t)$ and $F_p^2(t)$ denote the freshness decline for product p after transportation time t and storage duration t, respectively. The coefficients α_p and β_p are the respective freshness decline rates. Penalty costs due to freshness decline are calculated as follows:

$$H_{lp}^{\mathrm{V}}(\tau_{ik}^{\mathrm{S}}) = \omega(100\% - F_p^1(\tau_{ik}^{\mathrm{S}})) \tag{3}$$

$$H_{lp}^{\mathrm{L}}(\tau_{ik}^{\mathrm{S}}) = \omega(100\% - F_p^2(\kappa_{il}^{\mathrm{E}} - \tau_{ik}^{\mathrm{S}})) \tag{4}$$

where $H_{lp}^{\mathrm{V}}(\tau_{ik}^{\mathrm{S}})$ and $H_{lp}^{\mathrm{L}}(\tau_{ik}^{\mathrm{S}})$ indicate the penalty costs during delivery and storage, respectively. τ_{ik}^{S} is unloading start time of vehicle k at community i, which ω is the penalty cost related to freshness decline.

Late deliveries decrease customer satisfaction, leading to return costs. The penalty cost for late delivery is defined as:

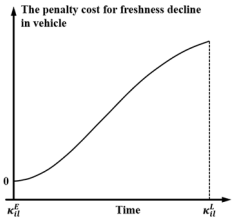
$$H_{lp}^{\mathrm{T}}(\tau_{ik}^{\mathrm{S}}) = a(\tau_{ik}^{\mathrm{S}} - \kappa_{il}^{\mathrm{L}}) \tag{5}$$

where $H_{lp}^{\mathrm{T}}(\tau_{ik}^{\mathrm{S}})$ represents the penalty cost for late delivery. If delivery occurs after the latest pickup time κ_{il}^{L}, the delay time is $\tau_{ik}^{\mathrm{S}} - \kappa_{il}^{\mathrm{L}}$, with a being the penalty cost coefficient for delays.

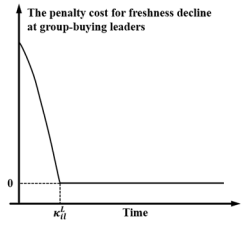
$H_{lp}^{\mathrm{V}}(\tau_{ik}^{\mathrm{S}})$, $H_{lp}^{L}(\tau_{ik}^{\mathrm{S}})$ and $H_{lp}^{\mathrm{T}}(\tau_{ik}^{\mathrm{S}})$ are referred to as Type I, Type II, and Type III penalty costs, respectively. Total penalty cost, $H_{lp}(\tau_{ik}^{\mathrm{S}})$, can be computed as follows.

$$H_{lp}(\tau_{ik}^{\mathrm{S}}) = \begin{cases} \omega(200\% - F_p^1(\tau_{ik}^{\mathrm{S}}) - F_p^2(\kappa_{il}^{\mathrm{E}} - \tau_{ik}^{\mathrm{S}})), & \tau_{ik}^{\mathrm{S}} < \kappa_{il}^{\mathrm{E}} \\ \omega(100\% - F_p^1(\tau_{ik}^{\mathrm{S}})), & \kappa_{il}^{\mathrm{E}} < \tau_{ik}^{\mathrm{S}} < \kappa_{il}^{\mathrm{L}} \\ \omega(100\% - F_p^1(\tau_{ik}^{\mathrm{S}})) + a(\tau_{ik}^{\mathrm{S}} - \kappa_{il}^{\mathrm{L}}), & \tau_{ik}^{\mathrm{S}} > \kappa_{il}^{\mathrm{L}} \end{cases} \tag{6}$$

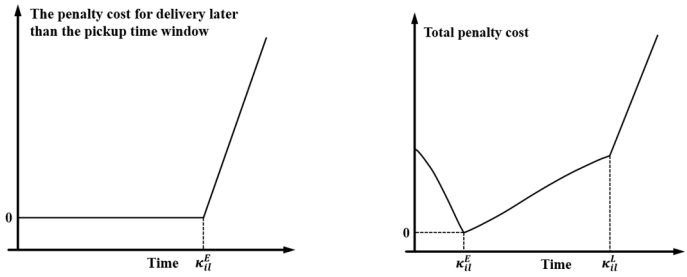
Figures 3 illustrate the function curves of Type I, Type II, and Type III penalty costs, along with the total penalty cost.



(a) Type I penalty cost



(b) Type II penalty cost



(c) Type III penalty cost **(d) Total penalty cost**

Fig. 3. The function curves of penalty costs

3.3 Mathematical Model

Table 1. Notations in the proposed model

Type	Notation	Description
Sets	G	$G = \{V, A\}$, a complete undirected network, V denotes the set of nodes and A represents the set of arcs in the network
	N	Set of communities, $N = \{1, \cdots, \lvert N \rvert\}$, $\lvert N \rvert$ represents the number of communities
	V	Set of nodes including the depot and communities, $V = \{0,1, \cdots, \lvert N \rvert + 1\}$, where 0 and $\lvert N \rvert + 1$ represent the depot for vehicles depart and return, respectively, the remaining nodes represent communities
	C	Set of customers, $C = \{1, \cdots, \lvert L \rvert\}$, $\lvert L \rvert$ represents the number of customer types
	P	Set of fresh products, $P = \{1, \cdots, \lvert P \rvert\}$, $\lvert P \rvert$ represents the number of fresh product types
	A	Set of arcs, $A = \{(i,j) \mid \forall i,j \in V\}$
	K	Set of vehicles, $K = \{1, \cdots, \lvert K \rvert\}$, $\lvert K \rvert$ represents the maximum number of vehicles
Parameters	e_i	Earliest service start time of the group-buying leader at community i, $i \in N$
	l_i	Latest service start time of the group-buying leader at community i, $i \in N$
	s_i	Service time at community i, where $i \in N$
	q_{ilp}	Demand of customer l for product p at community i, $i \in N$, $l \in C$, $p \in P$
	c_{ij}	Travel cost along arc (i,j), where $(i,j) \in A$
	ρ_k	Maximum fresh product types that vehicle k can load, $k \in K$
	t_{ij}	Travel time from community i to j, $i,j \in V: i \neq j$
	κ_{il}^{E}	Earliest pickup time for customer l at community i, $i \in N$, $l \in C$
	κ_{il}^{L}	Latest pickup time for customer l at community i, $i \in N$, $l \in C$

	ω	Penalty cost coefficient between freshness decline and incurred costs
	α_p	Freshness decline coefficient of product p in vehicle, $p \in P$
	β_p	Freshness decline coefficient of product p at group-buying leader, where $p \in P$
	γ	Penalty cost coefficient between the part exceeding the time window of group-buying leaders and incurred costs
	δ	Penalty cost coefficient between exceeding vehicle capacity and incurred costs
	Q	Maximum capacity of vehicles
	D	Maximum running time of vehicles on a delivery route
	M	A sufficiently large positive constant
Decision variables	w_{pk}	Load capacity of vehicle k for product p when departing from the depot, $k \in K,\ p \in P$
	x_{ijk}	$x_{ijk} = 1$if vehicle k traverses arc (i,j), otherwise $x_{ijk} = 0,\ (i,j) \in A,\ k \in K$
	τ_{ik}^{S}	Service start time of vehicle k at community $i,\ i \in N,\ k \in K$
	v_{ik}	Total load of vehicle k when arriving at community $i,\ i \in N,\ k \in K$
	L_{ipk}	$L_{ipk} = 1$ if vehicle k loads product p for community i, otherwise $L_{ipk} = 0,\ i \in N,\ k \in K,\ p \in P$

[P]

$$\min\left(\sum_{k\in K}\sum_{i\in V}\sum_{j\in V} c_{ij}x_{ijk} + \sum_{k\in K}\sum_{i\in N}\sum_{l\in C}\sum_{p\in P} H_{lp}\left(\tau_{ik}^{S}\right)q_{ilp}\right) \tag{7}$$

$$\sum_{j\in V} x_{0jk} = 1 \quad \forall k \in K \tag{8}$$

$$\sum_{j\in V} x_{j,|N|+1,k} = 1 \quad \forall k \in K \tag{9}$$

$$\sum_{j\in V} x_{jik} = \sum_{j\in V} x_{ijk} \quad \forall i \in N, \forall k \in K \tag{10}$$

$$\sum_{k\in K}\sum_{j\in V} x_{jik} = \sum_{k\in K}\sum_{j\in V} x_{ijk} \geq 1 \quad \forall i \in N \tag{11}$$

$$L_{ipk} \leq \sum_{j\in V} x_{ijk} \quad \forall i \in N, \forall p \in P, \forall k \in K \tag{12}$$

$$\sum_{p\in P} \min\left\{1, \sum_{i\in V} L_{ipk}\right\} \leq \rho_k \quad \forall k \in K \tag{13}$$

$$\min\left\{1, \sum_{l\in C} q_{ilp}\right\} \leq \sum_{k\in K} L_{ipk} \quad \forall i \in N, \forall p \in P \tag{14}$$

$$\sum_{j\in V} x_{ijk} \leq \sum_{p\in P} L_{ipk} \quad \forall k \in K, \forall i \in N \tag{15}$$

$$w_{pk} = \sum_{i\in N}\sum_{l\in C} q_{ilp}L_{ipk} \quad \forall p \in P, \forall k \in K \tag{16}$$

$$\sum_{p\in P} w_{pk} \leq Q \quad \forall k \in K \tag{17}$$

$$v_{ik} - \sum_{l\in C}\sum_{p\in P} q_{ilp}L_{ipk} + M\left(1 - x_{ijk}\right) \geq v_{jk} \quad \forall i \in N, \forall j \in N, \forall k \in K \tag{18}$$

$$v_{|N|+1,k} = 0 \quad \forall k \in K \tag{19}$$

$$\tau_{ik}^{\mathrm{S}} + s_i + t_{ij} - M\left(1 - x_{ijk}\right) \le \tau_{jk}^{\mathrm{S}} \quad \forall i \in N, \forall j \in V, \forall k \in K \tag{20}$$

$$\tau_{ik}^{\mathrm{S}} \le l_i + M\left(1 - \sum_{j \in V} x_{ijk}\right) \quad \forall i \in N, \forall k \in K \tag{21}$$

$$\tau_{ik}^{S} \ge e_i - M\left(1 - \sum_{j \in V} x_{ijk}\right) \quad \forall i \in N, \forall k \in K \tag{22}$$

$$\sum_{k \in K} \sum_{i \in V} x_{0ik} \le |K| \quad \forall k \in K \tag{23}$$

$$H_{lp}(\tau_{ik}^{\mathrm{S}}) = \begin{cases} \omega(200\% - F_p^1(\tau_{ik}^{\mathrm{S}}) - F_p^2(\kappa_{il}^{\mathrm{E}} - \tau_{ik}^{\mathrm{S}})), & \tau_{ik}^{\mathrm{S}} < \kappa_{il}^{\mathrm{E}} \\ \omega(100\% - F_p^1(\tau_{ik}^{\mathrm{S}})), & \kappa_{il}^{\mathrm{E}} < \tau_{ik}^{\mathrm{S}} < \kappa_{il}^{\mathrm{L}} \\ \omega(100\% - F_p^1(\tau_{ik}^{\mathrm{S}})) + a(\tau_{ik}^{\mathrm{S}} - \kappa_{il}^{\mathrm{L}}), & \tau_{ik}^{\mathrm{S}} > \kappa_{il}^{\mathrm{L}} \end{cases} \tag{24}$$

The notations in the proposed model is shown in Table 1. The model is mathematically represented using the arc-flow model [P]. The objective function aims to minimize the total cost, which comprises both basic travel costs and total penalty costs. Constraints ensure that each vehicle departs from and returns to the depot (Constraints 8-9). Each vehicle must arrive at and leave a community after service is completed (Constraint 10), and each community must be served by at least one vehicle (Constraint 11). Vehicles serve a community only if they pass through it (Constraint 12). Constraints on product types include limiting the number of product types each vehicle can load (Constraint 13), ensuring product delivery when there is demand in a community (Constraint 14), and requiring vehicles to deliver at least one type of product if they pass through a community (Constraint 15). The total load must match the demand (Constraint 16), within vehicle capacity limits (Constraint 17). The model calculates vehicle loads at each community (Constraint 18) and mandates complete unloading before returning to the depot (Constraint 19). Arrival times are calculated (Constraint 20), and time windows are enforced (Constraints 21-22). The number of vehicles is capped (Constraint 23), and the penalty cost function is included (Constraint 24).

4 Algorithm

4.1 Overview

The classical algorithm is introduced by Christian Prins (2004)14. Our proposed memetic algorithm incorporates three enhancements into the foundational framework of the traditional approach. First, it considers three types of penalty costs. Second, the split algorithm network is constructed following the addition of cloned points, with the formulation of arc costs in the split network provided. Third, the dynamic allocation process is conducted to optimize delivery times. Algorithm 1 illustrates the entire process of the tailored memetic algorithm, with individual components detailed in the following sections.

Algorithm 1. VRPTWMTMC

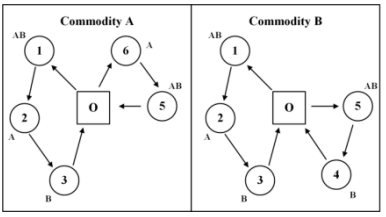
1: Initialize population according to the greedy strategy
2: Split each solution (route allocation procedure)
3: **while** number of iterations without improvement $< I_d$, and time $< T_{max}$ **do**

4: Select parent solutions P_1 and P_2 using elite selection
5: Generate offspring C from P_1 and P_2 (crossover)
6: Split C (route allocation procedure)
7: Educate C (local search procedure)
8: Allocation C (delivery time optimization procedure)
9: **if** C infeasible **then**
Assign C into infeasible subpopulation,
Repair with probability P_{rep}
10: **if** C feasible **then**
Insert C into feasible subpopulation
11: **if** reaching maximum subpopulation **then**
Select survivors
12: **if** best solution not improved for I_d iterations, **then**
Diversify population
13: **end while**
14: Return best feasible solution

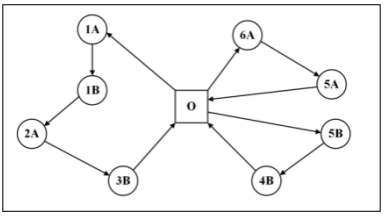
4.2 Solution Representation

A chromosome is a series of unseparated nodes, each containing information about a community and the demand for a specific product within that community. Given the order of nodes in a chromosome, the corresponding optimal solution can always be obtained using the Split algorithm.

An example is shown in Figure xx, where the solution representation is divided into four steps. In Figure 4, all the routes for delivering each commodity are indicated. If a community is regarded as multiple nodes based on commodity types, the multiple simultaneous delivery route solutions of Figure 4 (a) can be combined and transformed into Figure 4 (b). Figure 4 (c) represents the corresponding delivery pattern. The final solution representation with separators is shown in Figure 4 (d).



(a) Delivery Routes



(b) Delivery solution

Cus	1	2	3	4	5	6
Pat	{A,B}	{A}	{B}	{B}	{A},{B}	{B}

(c) Delivery Pattern

1A 1B 2A 3B 6A 5A 5B 4B

(d) Solution Representation

Fig. 4. The example of solution representation

4.3 Split Algorithm

The split algorithm requires the sequential order of all nodes and determines the costs associated with any segment of $\mathrm{arc}(i,j), i,j \in \mathrm{V}$. The cost associated with $\mathrm{arc}(i,j), i,j \in \mathrm{V}$ is formulated as follows:

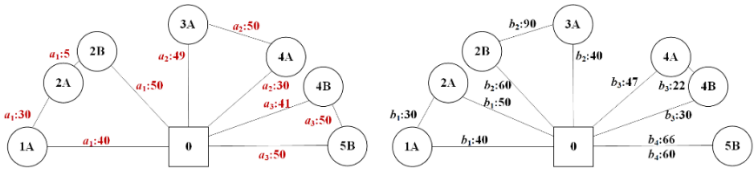
$$f_{ij} = c_{0,i+1} + c_{j,0} + \sum_{h=i+1}^{j-1} c_{h,h+1} + \sum_{h=i+1}^{j} \sum_{l \in C} \sum_{p \in P} H_{lp}(\tau_{hk}) q_{hlp} + \sum_{h=i+1}^{j} \gamma \max(\tau_{hk} - l_h, 0) + \delta \max(\sum_{p \in P} w_{pk} - Q, 0), k \in K \quad (25)$$

$$\tau_{ik}^{\mathrm{S}} + s_i + t_{ij} - M(1 - x_{ijk}) \leq \tau_{jk}^{\mathrm{S}}, k \in K (25) \quad (26)$$

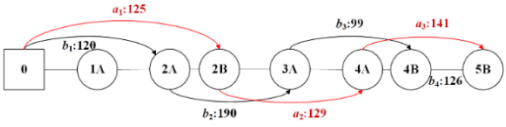
where f_{ij} represents the total cost of a delivery route visiting communities from i to j. $c_{0,i+1}$ is the fundamental delivery cost from the depot to the first community i, and $c_{j,0}$ is the cost from the last community j back to the depot. $c_{h,h+1}$ is the basic cost from community h to community $h+1$. Therefore, $\sum_{h=i}^{j-1} c_{h,h+1}$ represents the total delivery cost for a vehicle to deliver between communities.

Additionally, $\mathrm{arc}(i,j), i,j \in \mathrm{V}$ includes the penalty cost associated with serving different customers in different communities. $H_{lp}(\tau_{hk}) q_{hlp}$ denotes the penalty cost for customer l for product p at community h. Consequently, the total penalty cost can be expressed as $\sum_{h=i}^{j} \sum_{l \in C} \sum_{p \in P} H_{lp}(\tau_{hk}) q_{hlp}$. If $\mathrm{arc}(i,j), i,j \in \mathrm{V}$ does not satisfy the time window constraint or the capacity constraint, a corresponding penalty cost $\sum_{h=i+1}^{j} \gamma \max(\tau_{hk} - l_h, 0)$ or $\delta \max(\sum_{p \in P} w_{pk} - Q, 0)$ is incurred, respectively.

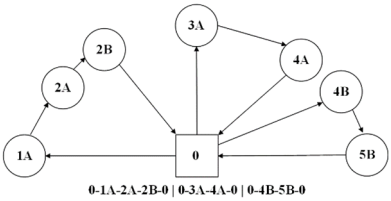
Figure 5 illustrates an example of the split algorithm. Figure 5 (a) depicts solutions with 3 and 4 delivery routes, respectively. These two solutions can be transformed into the directed acyclic graph shown in Figure 5 (b). In this graph, each arc cost is calculated by summing all the costs along the arc. For a fixed sequence order of nodes, the corresponding optimal solution with separators can be obtained using Bellman's algorithm (Prins, C., 2004). In this example, the optimal solution is shown in Figure 5 (c).



(a) Split network



(b) Directed acyclic graph



(c) Vehicle routing solution after split

Fig. 5. The example of Split algorithm

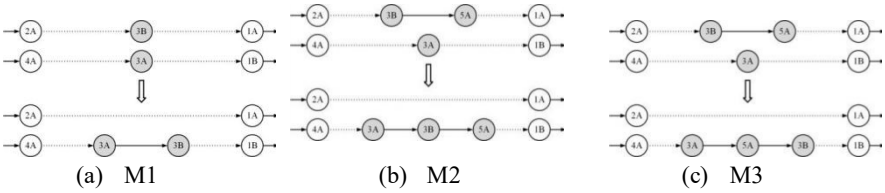
4.4 Crossover

Two parents are randomly selected from the population, and Ox crossover (Prins, C., 2004) is used in our algorithm. Ox crossover is advantageous as it can inherit favorable access sequences from both parents and rearrange the sequence of community nodes.

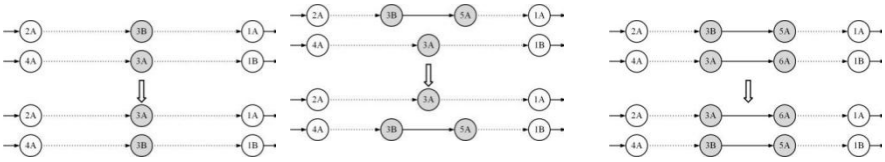
4.5 Education

After the crossover process, a local search method shown in Figure 6 is employed in place of mutation operators to improve solutions. Each operation traverses all possible pairs (u,v) within the routes. Let x and y denote the next nodes of u and v. T(u) is the route of node u.

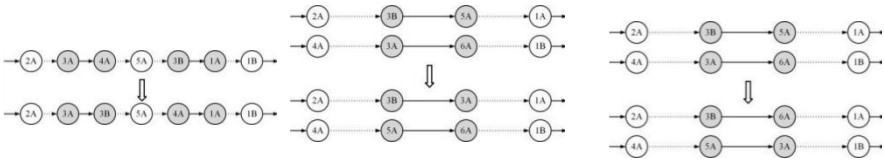
M1. If u is a client node, remove u then insert it after v,
M2. If u and x are clients, remove them then insert (u, x) after v,
M3. If u and x are clients, remove them then insert (x, u) after v,
M4. If u and v are clients, swap u and v,
M5. If u, x and v are clients, swap (u, x) and v,
M6. If (u, x) and (v, y) are clients, swap (u, x) and (v, y),
M7. If $T(u) = T(v)$, replace (u, x) and (v, y) by (u, v) and (x, y),
M8. If $T(u) \neq T(v)$, replace (u, x) and (v, y) by (u, v) and (x, y),
M9. If $T(u) \neq T(v)$, replace (u, x) and (v, y) by (u, y) and (x, v).



(a) M1 (b) M2 (c) M3



(d) M4 (e) M5 (f) M6



(g) M7 (h) M8 (i) M9

Fig. 6. Education operators

4.6 Dynamic Allocation of Delivery Time

For a sequence of fixed-order nodes optimized by education, the delivery time of each community node is uniquely determined for continuous delivery by vehicles. However, poor refrigeration equipment at some group-buying leaders leads to greater freshness decline during storage. If there is a large gap between the customer pickup time and the delivery time, a more optimal delivery solution involves vehicles pausing the delivery, waiting until it is closer to the customer pickup time, and then unloading, ensuring a better storage environment while waiting for pickup. Therefore, we propose a dynamic allocation of delivery time to enable proactive waiting by vehicles.

During the dynamic allocation process, the delivery time of each community node can be shifted backward within a reasonable range to allow for vehicle waiting. The wait cannot exceed the latest time window at the group-buying leader of this node, ensuring time window compliance. The allowable adjustment also cannot exceed the minimum of the interval between delivery time and the latest time window at the group-buying leader for all subsequent nodes in the route. This satisfies the time window constraints for the subsequent nodes. After determining the adjustable time range, a greedy algorithm optimizes the delivery time to minimize the penalty cost of each node.

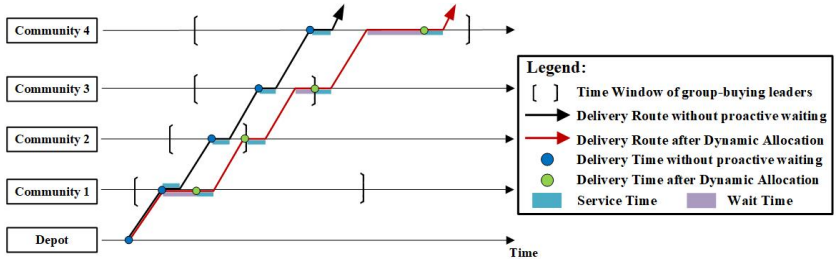


Fig. 7. The example of delivery time dynamic allocation

As shown in Figure 7, the black and red lines represent solutions before and after dynamic allocation, respectively. For community node 1, the range of allowed adjustment is small because the time window at the head of node 2 is narrow and needs to satisfy the time window constraints at node 2. For node 2, it has been forced to be delayed to the latest pickup time of the node, and no adjustment can be made. For node

3, to satisfy the current node's head of the node's time window, it can only be allowed to be adjusted up to the lower bound of the time window. For node 4, since there is no other node and the time window range of this node is larger, the adjustment range is larger. Each node takes the optimal delivery time within the allowed adjustment range to minimize the penalty cost of the node.

5 Experiment Results

5.1 Instance Generation

The instances used for the experiments are generated based on Solomon benchmark instances, encompassing 50 and 100 community nodes. These instances include types R (Random), C (Clustered), and RC (Random-Clustered). For each type, we generate two combinations based on two product types (2 and 4), resulting in a total of 54 instances for both the 50-node and 100-node scenarios.

Our proposed algorithm is programmed in Java language and subjected to a comparative experiment using CPLEX 12.8. All experiments are conducted on a laptop featuring an AMD Ryzen 7 5800H with Radeon Graphics, 3.20 GHz 8-core 16-thread Processor, and Windows 10 operating system. All computational times in experiments are reported in CPU seconds. The overall execution time limit for each run of the algorithm is set to 21600s.

5.2 Instance Results

After generating the instances, each type is solved using three different methods: a classical memetic algorithm, our proposed method without dynamic allocation, and our proposed method with dynamic allocation. The results for the two combinations of each instance type are averaged. The comparison of the results for the instances with 50 and 100 nodes, obtained from the three solving methods, is presented in the table below.

Table 2. Experimental results of the instances with 50 nodes

Name	classical memetic algorithm					our method						
	cpu	In.	Obj.	ADF	Vin.	cpu.	In.	Obj.	ObjDA.	ADF.	Vin.	gap.
C201	10830	313	13073	6.36	58	7253	142	4771	4700	24.35	62	-1.48%
C202	9985	309	5233	6.31	34	7232	197	3939	3842	9.67	153	-2.46%
C203	10254	314	24591	5.87	63	6641	395	3362	3282	4.41	235	-2.39%
C204	10815	324	17613	5.12	108	5252	474	2229	2178	2.73	132	-2.29%
C205	10828	313	16809	6.12	25	7209	234	4271	4216	13.13	134	-1.30%
C206	10830	327	3927	5.73	134	7209	238	3908	3857	11.18	97	-1.30%
C207	9023	314	10309	6.29	34	6981	345	3894	3871	6.00	266	-0.60%
C208	10839	316	25496	6.25	41	7225	206	4315	4284	12.52	114	-0.72%
R201	10815	319	86457	5.60	110	5360	398	1887	1851	3.79	235	-1.92%
R202	10820	328	25351	4.73	266	5303	434	1631	1621	3.25	303	-0.61%
R203	9290	337	44062	3.85	98	5075	474	1439	1470	2.68	281	2.11%

R204	10819	354	54475	3.64	280	4473	479	1172	1165	2.73	239	-0.57%
R205	10844	213	66208	8.70	121	5279	363	1804	1819	5.40	185	0.84%
R206	10796	312	32132	5.65	160	4886	429	1471	1460	3.17	251	-0.77%
R207	8609	328	21188	4.36	125	4822	500	1284	1258	2.42	338	-2.01%
R208	9340	348	14348	3.63	146	4711	478	1130	1131	2.46	459	0.10%
R209	10828	278	43669	7.97	85	5358	427	1672	1673	3.28	248	0.02%
R210	10832	304	3787	7.22	47	5365	353	1623	1593	5.63	179	-1.90%
R211	10850	287	29032	7.62	173	5063	488	1455	1477	2.49	364	1.54%
RC201	10838	311	84061	10.92	34	6105	361	2241	2236	9.48	342	-0.22%
RC202	8954	341	38567	7.38	31	6057	370	2008	2051	7.86	224	2.15%
RC203	10215	374	31291	5.68	11	6043	329	1710	1734	11.16	107	1.45%
RC204	9246	376	83928	5.10	31	5801	375	1369	1387	7.60	158	1.33%
RC205	10830	278	67019	11.90	165	5519	388	2051	2028	7.16	286	-1.11%
RC206	10373	302	16744	11.95	150	6223	399	1975	1995	6.99	265	1.04%
RC207	9029	292	3818	14.16	44	6106	356	1836	1926	8.46	113	4.87%
RC208	9807	310	27001	11.37	148	5409	346	1659	1655	9.75	132	-0.25%

Legend: In: Iteration number; ADF: Average discriminate of fitness; Vin: Valid iteration number; ObjDA: objective function with dynamic allocation

Table 3. Experimental results of the instances with 100 nodes

Name	classical memetic algorithm					our method						
	cpu	In.	Obj.	ADF	Vin.	cpu.	In.	Obj.	ObjDA.	ADF.	Vin.	gap.
C201	21990	155	331062	74.81	23	14487	33	8973	8892	227.38	27	-0.90%
C202	21955	168	384974	134.42	13	14602	84	7408	7328	81.72	72	-1.08%
C203	21951	55	84285	160.19	29	14517	112	5595	5494	51.98	107	-1.81%
C204	22022	65	321276	127.26	29	14470	226	4359	4383	29.59	67	0.54%
C205	21814	80	16985	138.49	23	14460	38	8236	8172	167.66	35	-0.78%
C206	21775	102	76543	127.94	21	14532	37	8042	8045	174.28	32	0.03%
C207	21853	144	32754	133.32	24	14727	47	7551	7475	114.58	46	-1.01%
C208	21845	75	46202	117.47	32	14551	69	7846	7792	104.35	68	-0.69%
R201	21910	62	165112	84.55	37	14423	100	3542	3569	49.52	63	0.74%
R202	21736	71	140236	75.90	20	14561	118	3038	3043	48.49	78	0.16%
R203	21702	87	45971	62.25	28	14574	123	2814	2766	39.22	51	-1.72%
R204	21834	188	70843	41.11	98	14440	173	2216	2204	20.65	140	-0.57%
R205	21888	75	107167	89.80	58	14520	122	3145	3092	33.90	91	-1.69%
R206	21902	90	91154	78.91	28	14458	113	2859	2855	44.25	90	-0.14%
R207	21772	65	35663	70.40	19	14454	161	2551	2500	29.42	153	-2.03%
R208	21939	225	37198	40.56	110	14428	258	2104	2147	17.15	207	2.03%
R209	22210	66	70783	110.95	27	14457	184	3030	2970	21.05	170	-1.98%
R210	22198	69	40531	101.13	24	14482	206	2939	2940	20.24	199	0.04%
R211	22292	62	39510	112.12	7	14436	243	2801	2800	24.53	193	-0.01%
RC201	22157	64	179106	170.12	37	14504	98	4162	4120	75.73	89	-0.99%
RC202	22126	132	107478	126.61	47	14492	119	3742	3780	54.92	115	0.99%
RC203	21959	105	54465	99.23	65	14486	130	3186	3176	55.97	100	-0.30%
RC204	21813	105	67927	89.24	69	14454	150	2703	2648	47.14	140	-2.04%

RC205	22032	41	139594	217.92	4	14431	107	4134	4124	74.32	87	-0.24%
RC206	21924	87	82628	178.92	47	14528	112	3671	3630	58.34	87	-1.10%
RC207	22143	73	27871	208.50	9	14435	129	3662	3613	48.96	111	-1.35%
RC208	22066	53	6634	255.74	47	14455	158	3214	3164	43.28	115	-1.56%

Legend: In: Iteration number; ADF: Average discriminate of fitness; Vin: Valid iteration number; ObjDA: objective function with dynamic allocation

The proposed Split algorithm in our method enhances solution time and increases the ratio of valid iterations compared to the classical memetic algorithm. The average computation times for the classical memetic algorithm on instances with 50 and 100 nodes are 10,242 seconds and 21,956 seconds, respectively. Our algorithm solves instances with 50 and 100 nodes in 5,850 seconds and 14,495 seconds, respectively, reducing computation time by 36.81%.

From Table 2 and 3, our method shows a significant improvement in the percentage of valid iterations relative to the total number of iterations. Specifically, our method increases this percentage by 79.39% and 107.82% for instances with 50 and 100 nodes, respectively, compared to the classical memetic algorithm. The objective function value obtained by our method is consistently lower than that obtained by the classical memetic algorithm. This improvement is attributed to our method's consideration of time windows for different customer types at each community node, which reduces penalty costs for both early and late deliveries of fresh products. For instances with 50 and 100 nodes, our algorithm achieves average gaps of -82.01% and -90.92%, respectively, compared to the classical memetic algorithm.

Additionally, our algorithm achieves an average gap of -0.44% compared to our proposed method without dynamic allocation. This is because the dynamic allocation process in our algorithm optimizes the delivery time of each community node, catering to the requirements for multiple types of customers. The optimization of the delivery time also allows fresh products to be stored in refrigerated vehicles longer, reducing the storage time at group-buying leaders and thereby minimizing freshness decline.

6 Conclusion

This paper investigates the multi-commodity delivery problem in community group-buying considering heterogeneous customer behaviors. Delivering fresh products to a leader introduces a time gap between delivery and customer retrieval, which will result a potential freshness decline and returns.

We develop a vehicle routing problem with multiple time windows and multiple commodities. To solve this model, we propose a memetic algorithm based on a split algorithm, which considers time-varying arc costs. Since waiting time has an impact on the penalty cost, we incorporate dynamic allocation for delivery time, optimizing delivery schedules under fixed sequences by determining the optimized waiting time. We validate the algorithm using 108 test instances derived from Solomon's benchmarks. Experimental results demonstrate that our algorithm outperforms the classical memetic algorithm in terms of solution time, objective function value, number of iterations, and average discrimination. Specifically, it achieves average gaps of -82.01% and -90.92%

relative to the classical memetic algorithm for instances with 50 and 100 nodes, respectively, and an improvement of 0.44% with the proposed dynamic allocation.

Our proposed algorithm is limited by the independent execution of dynamic delivery time allocation and route optimization, which are not currently integrated. Future enhancements could address this coupling, enabling our model and algorithm to be applicable across various vehicle routing optimization scenarios with time windows.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (72201017); This work was also supported by The 17th College Students' Innovation and Entrepreneurship Training Program.

References

1. National Bureau of Statistics, 2023. Statistical Bulletin on National Economic and Social Development of the People's Republic of China in 2023. Accessed 22nd May, 2024, https://www.stats.gov.cn/sj/zxfb/202402/t20240228_1947915.html.
2. Online Economic and Social Affairs, 2024. Fresh Agricultural Products and Community Group Purchasing Market Data Report of China in 2023. Accessed 22nd May, 2024, https://www.100ec.cn/zt/2023sxsqscbg/.
3. Wang, X. P., Wang, M., Ruan, J. H., Li, Y., 2018. Multi-objective optimization for delivering perishable products with mixed time windows. Advances in Production Engineering & Management, 13(3): 321-332.
4. Zheng, J., 2020. A vehicle routing problem model with multiple fuzzy windows based on time-varying traffic flow. IEEE access, 8: 39439-39444.
5. Belhaiza, S., M'Hallah, R., Ben Brahim, G., Laporte, G., 2019. Three multi-start data-driven evolutionary heuristics for the vehicle routing problem with multiple time windows. Journal of Heuristics, 25(3): 485-515.
6. Li, N., Li, G., 2022. Hybrid partheno-genetic algorithm for multi-depot perishable food delivery problem with mixed time windows. Annals of Operations Research: 1-32.
7. Leng, L., Zhang, J., Zhang, C., Zhao, Y., Wang, W., Li, G., 2020. A novel bi-objective model of cold chain logistics considering location-routing decision and environmental effects. Plos one, 15(4): e0230867.
8. Chen, J., Liao, W., Yu, C., 2021. Route optimization for cold chain logistics of front warehouses based on traffic congestion and carbon emission. Computers & Industrial Engineering, 161: 107663.
9. Chen, W., Zhang, D., Van Woensel, T., Xu, G., Guo, J., 2023. Green vehicle routing using mixed fleets for cold chain distribution. Expert Systems with Applications, 233: 120979.
10. Zhang, G., Dai, L., Yin, X., Leng, L., Chen, H., 2023. Optimization of multipath cold-chain logistics network. Soft Computing, 27(23): 18041-18059.
11. Yang, F., Tao, F., 2023. A Bi-Objective Optimization VRP Model for Cold Chain Logistics: Enhancing Cost Efficiency and Customer Satisfaction. IEEE Access, 11: 127043-127056.
12. Deng, H., Wang, M., Hu, Y., Ouyang, J., Li, B., 2021. An Improved Distribution Cost Model Considering Various Temperatures and Random Demands: A Case Study of Harbin Cold-Chain Logistics. IEEE Access, 9: 105521-105531.

13. Hsu, C. I., Hung, S. F., Li, H. C., 2007. Vehicle routing problem with time-windows for perishable food delivery. Journal of food engineering, 80(2): 465-475.
14. Prins, C., 2004. A simple and effective evolutionary algorithm for the vehicle routing problem. Computers & operations research, 31(12): 1985-2002.