Transportation Research Part D 113 (2022) 103502

Contents lists available at ScienceDirect

Transportation Research Part D

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

Low-carbon routing for cold-chain logistics considering the time-dependent effects of traffic congestion

Xiaolong Guo, Wei Zhang, Bingbing Liu *

Anhui Province Key Laboratory of Contemporary Logistics and Supply Chain, International Institute of Finance, School of Management, University of Science and Technology of China, Hefei 230026, China

ARTICLE INFO

Keywords:
Carbon emission
Vehicle routing problem
Cold chain logistics
Time-dependent effect
Traffic congestion

ABSTRACT

To meet the demands of green logistics while considering the time-dependent effects caused by traffic congestion, we establish a time-dependent green vehicle routing problem with time windows model for cold chain logistics. This model aims to minimize the total cost, including the transportation cost, refrigeration cost, carbon emission cost, and labor cost. Vehicles are allowed to wait to avoid a bad traffic environment after completing their services to customers. To solve the model, we develop a two-stage hybrid search algorithm. In the first stage of this algorithm, an adaptive large neighborhood search technique is used to determine the vehicle route, while in the second stage, a shortest-path algorithm is used to determine the departure time of the vehicles from customer's node. Finally, numerical experiments are performed to verify the effectiveness and superiority of our model and the proposed hybrid search algorithm by comparing with the standard instances and large-scale instances.

1. Introduction

Global warming caused by carbon emissions is one of the world's best-known and most complex issues, threatening natural ecosystems and human survival. This threat should be taken seriously and effectively resolved. As a developing economy with a growing influence around the world, China has become the world's largest energy consumer and carbon dioxide emitter. In 2016, China's carbon dioxide emissions reached 5.2 billion tons, accounting for 15.5 % of global emissions (Lin et al., 2020). In the carbon emission statistics, transportation accounts for 14 %, while road carbon emissions account for 75 % of the total carbon emissions of the transportation sector (Dekker et al., 2012). Road transportation is the most important mode of logistics transportation, and although it has only a small loading capacity, it has high carbon emission cost and energy consumption. Therefore, the realization of green and low-carbon in the logistics industry is naturally put on the agenda, which will have a significant impact on China's ecosystem.

Compared with conventional logistics and distribution, cold chain logistics consume more energy and thus emit more carbon dioxide because of the need to maintain a specified low-temperature environment. This trade-off highlights the contradiction between the high energy consumption of cold chain logistics and the low emission goals of green logistics. With the continuous improvement of the economic level in China, the scale of the cold chain logistics market is increasing year by year, and it is expected to reach $80 billion in revenue in 2024 (Dai et al., 2020). This provides a precious opportunity for the development of cold chain logistics enterprises in China but also creates huge challenges.

* Corresponding author.
E-mail addresses: gxl@ustc.edu.cn (X. Guo), zw1231@mail.ustc.edu.cn (W. Zhang), lbb1224@ustc.edu.cn (B. Liu).

https://doi.org/10.1016/j.trd.2022.103502
Received 17 January 2022; Received in revised form 6 October 2022; Accepted 18 October 2022
Available online 29 October 2022

Usually, the customers of cold chain logistics are mainly distributed in cities. Unfortunately, traffic congestion has become a common phenomenon in cities due to road speed limits, traffic control, and accidents. Traffic congestion leads to longer vehicle traveling times and significant increases in vehicle energy consumption and carbon emissions. Therefore, to improve delivery service quality, it is indispensable to consider the time-dependent characteristics of the road network caused by traffic congestion when designing delivery plans and strategies.

The above background shows that cold chain logistics, especially when traffic congestion occurs, lead to high energy consumption and carbon emissions. However, the current related research has certain limitations. Most studies assume that the vehicle departs at the same time from the distribution center each day and leaves for the next node immediately after completing the service at each node. Due to traffic issues, this strategy may cause the vehicle to travel in a congested environment for a long time, resulting in increased costs. Therefore, how to plan the routing of vehicles for cold chain logistics while optimizing the departure time after completing the service at each node is a valuable problem worthy of study.

To address the above limitations and considering the feasibility in reality, this paper transforms energy conservation and emission reduction into green costs and takes the minimum cost as the objective function to study the cold chain logistics distribution problem in a traffic congestion environment. The contributions of this paper are summarized as follows:

(i) This paper introduces the time-dependent green vehicle routing problem with time windows (TDGVRPTW) in cold chain logistics, comprehensively considering traditional factors such as congestion, capacity, and time window, and a mixed integer programming (MIP) model is established to formulate the problem. (ii) In response to congestion, after a van completes the delivery task at the customer node, it is allowed to delay its departure time; that is, the strategy of "post-service waiting" is adopted to avoid bad traffic environments. (iii) To solve the model, a two-stage hybrid search algorithm (TSHSA) is proposed to optimize the vehicle route and the departure time from each node. Numerical experiments verify the superiority of the proposed approach. Compared with the branch and bound algorithm (implemented in the general MIP solver CPLEX), the TSHSA solution is 14.14 % better than the CPLEX solution for small-scale instances, and the TSHSA also performs well in solving large-scale instances that CPLEX cannot handle practically. Moreover, the solving time of TSHSA is shorter than that of CPLEX. In addition, we also compare the TSHSA with three heuristics and find that our algorithm is superior to these heuristic methods in solving time and quality. Specifically, the average solution improvement is 11.92 % for the adaptive large neighborhood search algorithm (ALNS), 15.94 % for the ant colony algorithm (ACO) and 13.1 % for the genetic algorithm (GA). (iv) The approach developed in this paper contributes to the literature on the problem modeling and solving of cold chain logistics by dealing with the time-dependent effect and carbon emission, which are inevitably encountered in the cold chain logistics distribution. Furthermore, the proposed algorithm provides direct decision-making guidance for the efficient operation of cold chain logistics distribution.

The structure of this paper is organized as follows. We review the literature related to this research in Section 2. Section 3 briefly describes the problem, and a TDGVRPTW model for cold chain logistics is constructed in Section 4. In Section 5, we propose a two-stage hybrid algorithm. Section 6 presents a series of numerical studies and result analyses. Finally, we conclude this paper in Section 7.

2. Literature review

The vehicle routing problem (VRP) was first proposed by Dantzig and Ramser (1959) and quickly became a hot topic for scholars. Since then, many achievements in this research topic have been added to the literature over the sixty years. Our proposed cold chain TDGVRPTW is built on classical VRP problems to optimize the total distribution cost while considering additional factors such as the time window, vehicle capacity, and congestion. This study belongs to the field of green logistics, so we briefly review closely related work in this stream of the literature.

2.1. Green routing

In recent years, green (or low-carbon) development operations have been advocated, and the green vehicle routing problem (GVRP) has attracted the attention of scholars. GVRP and traditional VRP both consider economic benefits, but their main difference is that for GVRP, environmental effects must also be taken into account, including vehicle fuel consumption, carbon emissions, and other environmental factors (Erdoğan et al., 2012). Fuel consumption and carbon emissions are affected by load, speed, road slope, traffic congestion, and other factors (Barth et al., 2005; Demir and Bektaş, 2011; Demir and Bektaş, 2014; Suzuki et al., 2016; Yao et al., 2020), so the modeling and optimization of the GVRP are much more complex than those for the traditional VRP. Barth et al. (2005) built the comprehensive modal emission model (CMEM), and Xiao et al. (2012) presented the load-dependent fuel consumption model (LFCM); these two models have been widely used in subsequent research to accurately measure fuel consumption. On this basis, many scholars have studied GVRP variants, such as those with heterogeneous vehicles (Kwon and Choi, 2013; Li et al., 2019), electric vehicles (Zhen et al., 2019), and energy minimization requirements (Ghannadpour et al., 2019). More recently, Yang et al. (2020) proposed a mixed integer programming model to explore the effects of external factors such as road, traffic conditions, and weather on vehicle fuel consumption. Zeng et al. (2020) established a link-based fuel consumption model for an eco-routing navigation system. Then, they designed a Lagrangian-relaxation-based heuristic approach to solve the model.

Note that the pollution routing problem (PRP), first proposed by Bektaş and Laporte (2011), is an important variant of the GVRP. The focus of the PRP is to consider environmental costs in the objective function, and the route and speed need to be determined at the same time. Demir et al. (2012) designed a speed optimization algorithm for the PRP to calculate the best speed on a specific route, thereby minimizing fuel consumption, carbon emissions, and driver costs. Then, PRP was extended to a multiobjective version (Suzuki

et al., 2016; Rauniyar et al., 2019; Raeesi and Zografos, 2019), with objectives including fuel consumption, travel distance, travel time, and vehicle hiring costs. Xiao et al. (2020) presented a new realistic variant, the continuous pollution routing problem (CPRP), which extended the discrete PRP to a continuous case.

However, most of the above literature on green routing assumes that the vehicle speed is constant, which cannot truly reflect industry practice. This paper introduces time-dependent traffic networks and carbon emissions into cold chain logistics, constructing a mixed integer programming model to optimize the total operating costs.

2.2. Time-dependent routing

Due to morning and evening traffic peaks, road speed limits, traffic control, and accidents, any urban traffic road network has time-dependent characteristics, and scholars have gradually begun to study the time-dependent vehicle routing problem (TDVRP), which is an extension of VRP. In TDVRP, the travel time between nodes changes with the departure time of the vehicle in a dynamic road network. Malandraki and Daskin (1992) first proposed the TDVRP, using a mixed integer programming model to formulate the problem. However, their travel time model does not satisfy the "first-in-first-out" (FIFO) property, which can be intuitively expressed as follows: if two vehicles take the same route, the one that departs first will arrive first. Later, Ichoua et al. (2003) solved this problem by modeling a piecewise linear function to represent travel time at different departure times.

On this basis, some scholars have carried out further research on the TDVRP model, considering design elements such as stochasticity in time-dependent vehicle speeds (Çimen and Soysal, 2017), vehicle depreciation cost and fuel consumption cost (Huang et al., 2017), capacitated profitable tour with precedence constraints (Sun et al., 2018), dual objectives (Wang et al., 2019; Xu et al., 2019), and multitrip time-dependent vehicle routing (Pan et al., 2021). Due to the complexity of the time-dependent problem, most studies have not found optimal analytical solutions but rather have proposed heuristic algorithms, such as genetic algorithm and particle swarm optimization (Zhu and Hu, 2018), ant colony algorithm (Liu et al., 2020a) and adaptive large neighborhood search (Zhang et al., 2020; Sun et al., 2020; Pan et al., 2021). It is noteworthy that Franceschetti et al. (2013) extended the PRP, constructed a time-dependent pollution routing problem (TDPRP) model by explicitly considering traffic congestion, and designed a recursive subroutine to optimize the departure time and speed on a fixed route. Then, they (Franceschetti et al. 2017; Franceschetti et al. 2018) continued their previous research by designing an adaptive large neighborhood search algorithm (ALNS) and an accurate shortest path algorithm to solve the TDPRP.

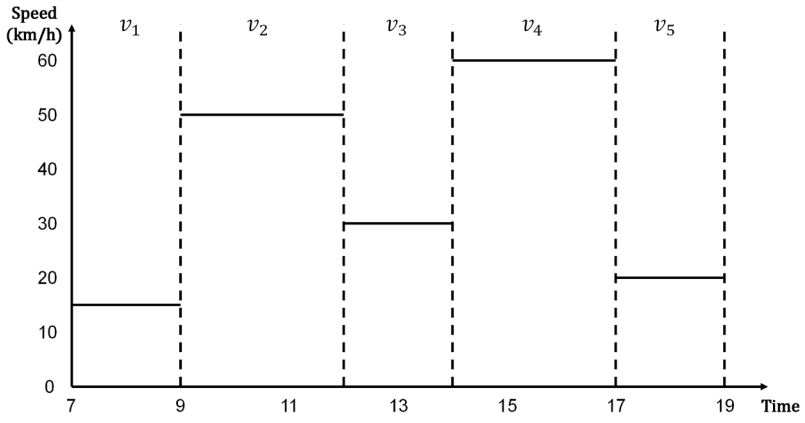
Recently, researchers have developed more interesting TDVRP variants (Yao et al., 2019; Visser et al., 2020; Vu et al., 2020; Florio et al., 2021; Mansourianfar et al., 2022; Zhou et al., 2022). Rincon- Garcia et al. (2020) established and analyzed a TDVRP model that considered time windows, time-dependent travel times and rules on driving hours. Then, they designed a large neighborhood search algorithm to obtain the optimal solution. Gmira et al. (2021) constructed a TDVRPTW model, in which travel speeds are associated with road segments, and designed a tabu search heuristic. Fan et al. (2021) studied a multidepot TDVRP that aims to minimize the total generalized costs, including fixed costs, penalty costs and fuel costs, and developed a hybrid genetic algorithm with variable neighborhood search. Allahyari et al. (2021) presented a hybrid algorithm to solve TDVRP with uncertain demands, which has combined the greedy randomized adaptive search procedure and the iterated local search. Fontaine (2022) considered the load-dependent travel speed and proposed the vehicle routing problem with load-dependent travel times.

The above literature has laid a good foundation for TDVRP research. This paper differs from the above literature in the following two aspects. (i) To avoid harsh traffic environments, vehicles are allowed to wait after completing the delivery service, and the vehicle can choose to depart from the distribution center freely at any time. The experimental results prove that this strategy is effective. (ii) Most previous studies use heuristic algorithms to optimize the departure time from each node, whereas this paper designs an exact shortest path optimization algorithm (the second stage of the TSHSA). After determining the distribution path, the optimal departure time from each node can be accurately determined.

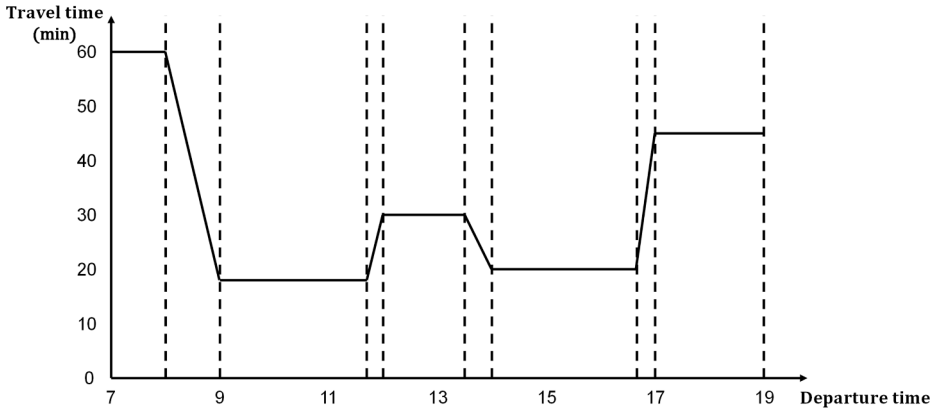
2.3. Cold chain routing

The VRP model studied in this paper mainly applies to cold chain logistics. Because cold chain logistics require temperature control to maintain the quality of the product, more fuel will be consumed for thermal insulation during transportation, and more carbon emissions will be generated (Stellingwerf et al., 2018). Therefore, most of the subsequent literature introduces a carbon emission cost into cold chain logistics to realize low-carbon logistics (Babagolzadeh et al., 2020; Liu et al., 2020b; Chen et al., 2021; Li and Zhou, 2021). In terms of the research objective and scope, the research of Meneghetti and Ceschia (2019) may be the most relevant to our study. They established a refrigerated route problem (RRP) model with the optimization objective of minimizing fuel consumption during transportation. The model takes into account speed variations due to traffic congestion and the decreasing load along the route as successive customers are visited. Chen et al. (2019) considered the costs of quality deterioration and carbon emissions and established a novel routing optimization model for cold chain distribution. Wang and Leng (2020) established a dual-objective model to optimize the total cost and total travel time, after which they designed a multiobjective hyperheuristic optimization framework to solve the problem.

The above cold chain logistics literature mainly discusses the economy, ignoring the impact of traffic congestion, and rarely involves the green benefits of vehicle energy conservation and emission reduction. Therefore, this paper develops the mathematical model and solution algorithm of the TDGVRPTW to provide a decision-making reference for related logistics enterprises.



(a) Time-dependent speed.



(b) Travel time profiles for nodes 15 km apart.

Fig. 1. Speed and travel time functions.

3. Problem description

This section introduces and analyzes the TDGVRPTW in cold chain logistics. A logistics firm has a homogeneous fleet of vehicles to distribute fresh agricultural products to customers scattered all over a city. The goal is to keep total operating costs as low as possible for up to K vehicles. We define a complete graph $G = \{N, E\}$ where $N = \{0, 1, \cdots, n\}$ is the set of nodes, 0 stands for the distribution center, $N^* = N \backslash \{0\}$ is the set of customer nodes, and $E = \{(i,j) : i,j \in N \text{ and } i \neq j\}$ is the set of arcs connecting nodes. The distance between i and j is denoted by d_{ij}. Each vehicle has the same maximum capacity Q.

Each customer $i \in N^*$ is assigned one and only one vehicle to complete the delivery task, and the customer has a nonnegative demand q_i. Each customer $i \in N^*$ has a service time s_i (i.e., the duration required to provide service at the node) and a hard time window $[l_i, u_i]$. Specifically, if a van arrives at customer node i before time window l_i, the service cannot be started immediately; delivery of the service must wait until time l_i. This is called "preservice waiting". After a van completes its delivery task at node i, its departure time at this node can be postponed to avoid harsh traffic environments. This is called "postservice waiting". The remainder of this section will introduce the travel time model and the total cost of the TDGVRPTW model, including transportation cost (C_{TR}), refrigeration cost (C_{RE}), carbon emissions cost (C_{EM}), and labor cost (C_{LA}). Please see Appendix A for all the parameters and decision variables used.

3.1. Modeling travel time

Since the delivery service always occurs during the day, we collected the average vehicle speed from 7 a.m. to 7p.m. in the main streets of a large city in China (sample data are provided in Appendix B). Specifically, following the model proposed by Zhu and Hu (2018), a vehicle has a lowest speed during the morning and evening peaks and the travel speed can be conversed to a stepwise function regarding the departure time. We choose the average speed between 7 a.m. and 9 a.m. (about 15 km/h) to represent the

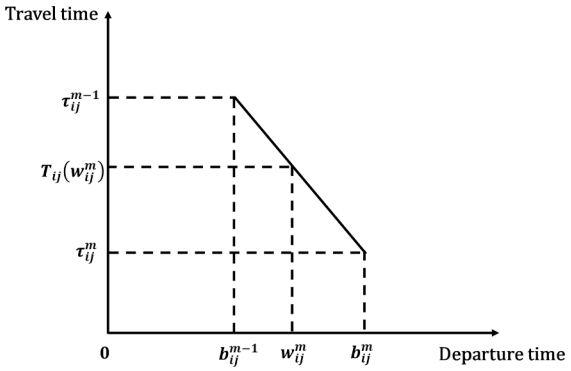


Fig. 2. Travel time as a function of departure time.

vehicle speed in the morning peak, and the average speed between 5p.m. to 7p.m. (about 20 km/h) to represent the evening peak. Since the average speed in our collected speed is much lower than that in Zhu and Hu (2018), we keep their high speed (60 km/h) in our function. Moreover, our data show that the average speed during 2p.m. to 5.p.m. is higher than that during 9 a.m. to 11 a.m. Therefore, we further distinguish these two sections with setting a lower speed (50 km/h) for the period from 9 a.m. to 11 a.m. Furthermore, many cities have noon peaks which also affect the travel speed, although this phenomenon is not shown well in the collected data due to limited datasize. We add such a section (with a average vehicle speed 30 km/h) in the function to investigate its effect, which will provide us more realistic results without affecting the model and algorithm proposed in this paper. Thus, the vehicle speed at different times as a stepwise function is shown in Fig. 1(a). As an example, assume that the vehicle needs to travel from one node to another node 15 km away; the corresponding travel time function is shown in Fig. 1(b), and it varies with different departure times. It should be noted that for each arc (i,j) with a different travel distance d_{ij}, the corresponding travel time function is also different.

According to the different slopes of the travel time function, we define the period as an index set $M = \{m|m = 1, 2, \cdots\}$. For instance, as shown in Fig. 1(b), the travel time function is divided into nine periods. Following the methods proposed by Ichoua et al. (2003) and Pan et al. (2021), the travel time for any departure time $b_{ij}^{m-1} \le w_{ij}^{m} \le b_{ij}^{m}$ can be obtained as follows:

$$T_{ij}\left(w_{ij}^{m}\right) = \left(\frac{\tau_{ij}^{m} - \tau_{ij}^{m-1}}{b_{ij}^{m} - b_{ij}^{m-1}}\right)\left(w_{ij}^{m} - b_{ij}^{m-1}\right) + \tau_{ij}^{m-1}. \tag{1}$$

Formula (1) is illustrated in Fig. 2, in this formula, $b_{ij}^{m-1}\left(b_{ij}^{m}\right)$ is the breakpoint, τ_{ij}^{m-1} (τ_{ij}^{m}) is the travel time when the departure time is b_{ij}^{m-1} (b_{ij}^{m}), w_{ij}^{m} represents the departure time within period $m \in M$ from node $i \in N$ to traverse arc $(i,j) \in E$, and $T_{ij}\left(w_{ij}^{m}\right)$ represents the travel time when the departure time is w_{ij}^{m}.

Because the travel time $T_{ij}\left(w_{ij}^{m}\right)$ may span multiple periods, we use $t_{ij}^{\eta}\left(w_{ij}^{m}\right), \eta \in M$ to denote the travel time of the vehicle in the η-th period at the departure time w_{ij}^{m}. Therefore,

$$T_{ij}\left(w_{ij}^{m}\right) = \sum_{\eta \in M} t_{ij}^{\eta}\left(w_{ij}^{m}\right). \tag{2}$$

Similarly, when the departure time of the vehicle is w_{ij}^{m}, the travel time of the vehicle in each time period can be calculated as:

$$t_{ij}^{\eta}\left(w_{ij}^{m}\right) = \left(\frac{\tau_{ij}^{\eta,m} - \tau_{ij}^{\eta,m-1}}{b_{ij}^{m} - b_{ij}^{m-1}}\right)\left(w_{ij}^{m} - b_{ij}^{m-1}\right) + \tau_{ij}^{\eta,m-1}. \tag{3}$$

In formula (3), $\tau_{ij}^{\eta,m-1}$ $(\tau_{ij}^{\eta,m})$ is the travel time in the η-th period when the departure time is b_{ij}^{m-1} (b_{ij}^{m}).

3.2. Transportation cost

The transportation cost consists of the fuel consumption in the process of vehicle operation. We use the CMEM of Barth et al. (2005) and Scora et al. (2006) to calculate fuel consumption. The instantaneous fuel rate is given as follows:

$$FR = A(\mu + f)v + B + Cv^{3}. \tag{4}$$

Here, A is the CMEM weight module constant, B is the CMEM engine module constant, C is the CMEM speed module constant, v denotes the vehicle instantaneous speed, μ denotes the vehicle weight, and f is the vehicle load.

Therefore, the total transportation cost is given as follows:

$$C_{TR} = \varphi_f \sum_{(i,j)\in E} \left(A(\mu + f_{ij}) x_{ij} d_{ij} + \sum_{m\in M} BT_{ij}\left(w_{ij}^m\right) z_{ij}^m + \sum_{m\in M}\sum_{\eta\in M} Ct_{ij}^{\eta}\left(w_{ij}^m\right) v_m^3 z_{ij}^m \right), \tag{5}$$

In (5), φ_f represents the unit fuel consumption cost, and x_{ij} is a binary variable such that when a vehicle traverses from node i to j, $x_{ij} = 1$, and otherwise $x_{ij} = 0$. Here, z_{ij}^m is also a binary variable: $z_{ij}^m = 1$ if the vehicle traverses arc $(i,j) \in E$ and departs node i during the m-th period, and otherwise, $z_{ij}^m = 0$. Finally, f_{ij} represents the load of the vehicle on arc $(i,j) \in E$.

3.3. Refrigeration cost

Unlike ordinary VRP, cold chain VRP includes the cost of refrigeration during the distribution process. When a vehicle is providing service at a customer node, the door needs to be opened. Due to air convection, the temperature in the vehicle compartment will rise, resulting in higher refrigeration costs. Consequently, the refrigeration cost in the distribution process is divided into refrigeration cost during transportation and refrigeration cost during service. The total refrigeration cost can be expressed as:

$$C_{RE} = \varphi_f \left(\alpha_{r1} \left(\sum_{i\in N^*} h_i - \sum_{j\in N^*}\sum_{m\in M} w_{0j}^m z_{0j}^m - \sum_{i\in N^*} s_i \right) + \alpha_{r2} \sum_{i\in N^*} s_i \right). \tag{6}$$

In formula (6), h_i represents the time to return to the distribution center on the route that has node $i \in N^*$ as the last visited customer node, and w_{0j}^m is the departure time of the vehicle in the m-th period from the distribution center to traverse arc $(0,j) \in E$. Parameters α_{r1} and α_{r2} represent the fuel consumption of refrigeration equipment per unit time during transportation and unloading, respectively, and s_i is the service time at node i.

3.4. Carbon emission cost

The cost of carbon emissions refers to the carbon dioxide generated by consuming fuel during the cold chain distribution process. In the cold chain, the fuel consumption of vehicles mainly comes from two aspects: fuel consumption for driving the vehicle and additional fuel consumption for maintaining the target temperature of the refrigeration equipment. The relationship between vehicle carbon emissions and fuel consumption is usually linear. In this paper, referring to the method of Toro (2017), we assume that $1L$ of gasoline produces $2.3kg$ of carbon emissions. Letting $\partial = 2.3kg/L$, the total carbon emissions cost is.

$$C_{EM} = \varphi_c \partial \frac{C_{TR} + C_{RE}}{\varphi_f}. \tag{7}$$

In formula (7), φ_c is the unit carbon emissions cost, and ∂ is the carbon emissions produced per unit of fuel consumed.

3.5. Labor cost

The labor cost refers to the wages paid by logistics companies to drivers. In reality, the driver's wage consists of two parts: a fixed base salary and a bonus for the quantity of goods delivered. The total labor cost is given as follows:

$$C_{LA} = \varphi_d \sum_{i\in N^*} x_{i0} + \varphi_q \sum_{i\in N^*} q_i. \tag{8}$$

In formula (8), φ_d represents a fixed base salary for each driver, x_{i0} is a binary variable (when the vehicle traverses from customer i to depot 0, $x_{i0} = 1$; otherwise, $x_{i0} = 0$),φ_q is the delivery bonus for the driver per unit of quality goods, and q_i is the demand at customer $i \in N^*$.

4. Model formulation

In this section, we formulate the mathematical model of the cold chain TDGVRPTW. The objective is to minimize the total cost by finding a set of driving routes and the departure time of the vehicles at each node. The decision variables are as follows:

x_{ij}: Binary variable, $x_{ij} = 1$ if arc $(i,j) \in E$ is traversed, 0 otherwise;

z_{ij}^m: Binary variable, $z_{ij}^m = 1$ if the vehicle traverses arc $(i,j) \in E$ and departs node i during the m-th period, 0 otherwise;

w_{ij}^m: Departure time in the m-th period from node $i \in N$ to traverse arc $(i,j) \in E$;

f_{ij}: Load of vehicle on arc $(i,j) \in E$;

r_i: Starting service time at node $i \in N^*$;

h_i: Time to return to the distribution center on the route that has a node $i \in N^*$ as the last visited customer node.

Now, we formulate a TDGVRPTW nonlinear integer programming model for cold chain logistics:

$$\text{minimize} \quad C_{TR} + C_{RE} + C_{EM} + C_{LA} \tag{9}$$

subject to.

$$\sum_{j\in N^*} x_{0j} \le K \tag{10}$$

$$\sum_{i\in N} x_{ij} = 1 \ \forall j \in N^* \tag{11}$$

$$\sum_{j\in N} x_{ij} = 1 \ \forall i \in N^* \tag{12}$$

$$\sum_{m\in M} z_{ij}^m = x_{ij} \ \forall (i,j) \in E \tag{13}$$

$$\sum_{j\in N} f_{ji} - \sum_{j\in N} f_{ij} = q_j \ \forall i \in N^* \tag{14}$$

$$q_j x_{ij} \le f_{ij} \le x_{ij}(Q - q_i) \ \forall (i,j) \in E \tag{15}$$

$$l_i \le r_i \le u_i \ \forall i \in N^* \tag{16}$$

$$r_i + s_i \le \sum_{m\in M} w_{ij}^m \ \forall i \in N, \forall j \in N \tag{17}$$

$$z_{ij}^m b_{ij}^{m-1} \le w_{ij}^m \le z_{ij}^m b_{ij}^m \ \forall (i,j) \in E, \forall m \in M \tag{18}$$

$$\sum_{m\in M}\sum_{i\in N} z_{ij}^m \left(w_{ij}^m + T_{ij}\left(w_{ij}^m\right)\right) \le r_j \ \forall j \in N^* \tag{19}$$

$$\sum_{m\in M} z_{i0}^m \left(w_{i0}^m + T_{i0}\left(w_{i0}^m\right)\right) \le h_i \ \forall i \in N^* \tag{20}$$

$$x_{ij} \in \{0,1\},\ z_{ij}^m \in \{0,1\},\ f_{ij} \ge 0,\ r_i \ge 0$$

$$w_{ij}^m \ge 0,\ h_i \ge 0 \tag{21}$$

Objective function (9) minimizes the total cost. Constraint (10) means that all vehicles depart from the distribution center, and the number of vehicles used cannot exceed K. Constraints (11) and (12) mean that all customer nodes are served exactly once, an in-out balance constraint. Constraint (13) represents the limiting relationship between variables x_{ij} and z_{ij}^m. Constraints (14) and (15) mean that the needs of each customer are met, and vehicle capacity constraints are respected. Constraint (16) requires customer service to be performed within the time window. Constraint (17) pertains to the relationship between the customer starting service time, service time, and departure time. When the left side of the equation is equal to the right side, the vehicle departs directly from the customer node after completing the service. When the left side of the equation is smaller than the right side, it means that after completing the service, the vehicle waits for a period of time and then departs from the customer node. Constraint (18) determines the boundary conditions for departure time. Constraint (19) enforces the time cohesion of vehicles traveling between different nodes. Constraint (20) calculates the time for the vehicle to return to the distribution center. Constraint (21) captures the value constraints of the variables. The following proposition guarantees that the above nonlinear model can be transformed into a linear integer programming model.

Proposition 1. *If the travel speed is a stepwise function, then the corresponding travel time function and constraints can be linearized.*

Proof. With constraint (18), we have $z_{ij}^m w_{ij}^m = w_{ij}^m,\ m \in M, (i,j) \in E$. Similarly, with constraint (15), we have $f_{ij}x_{ij} = f_{ij},\ (i,j) \in E$. Therefore, combining formula (1) and formula (3), C_{TR} in objective function (9) can be replaced by formula (22), and constraints (19) and (20) can be replaced by constraints (23) and (24), respectively.

$$\begin{aligned} C_{TR}^{'} = \varphi_f \sum_{(i,j)\in E} \Bigg(& A\left(\mu x_{ij} + f_{ij}\right) d_{ij} + \sum_{m\in M} B \left(\left(\frac{\tau_{ij}^m - \tau_{ij}^{m-1}}{b_{ij}^m - b_{ij}^{m-1}} \right) \left(w_{ij}^m - z_{ij}^m b_{ij}^{m-1} \right) + z_{ij}^m \tau_{ij}^{m-1} \right) \\ & + \sum_{m\in M}\sum_{\eta\in M} C \left(\left(\frac{\tau_{ij}^{\eta,m} - \tau_{ij}^{\eta,m-1}}{b_{ij}^m - b_{ij}^{m-1}} \right) \left(w_{ij}^m - z_{ij}^m b_{ij}^{m-1} \right) + z_{ij}^m \tau_{ij}^{\eta,m-1} \right) v_m^3 \Bigg) \end{aligned} \tag{22}$$

$$\sum_{m\in M}\sum_{i\in N} \left(w_{ij}^m \left(1 + \frac{\tau_{ij}^m - \tau_{ij}^{m-1}}{b_{ij}^m - b_{ij}^{m-1}} \right) + z_{ij}^m \left(\tau_{ij}^{m-1} - b_{ij}^{m-1} \frac{\tau_{ij}^m - \tau_{ij}^{m-1}}{b_{ij}^m - b_{ij}^{m-1}} \right) \right) \le r_j \ \ \forall j \in N^* \tag{23}$$

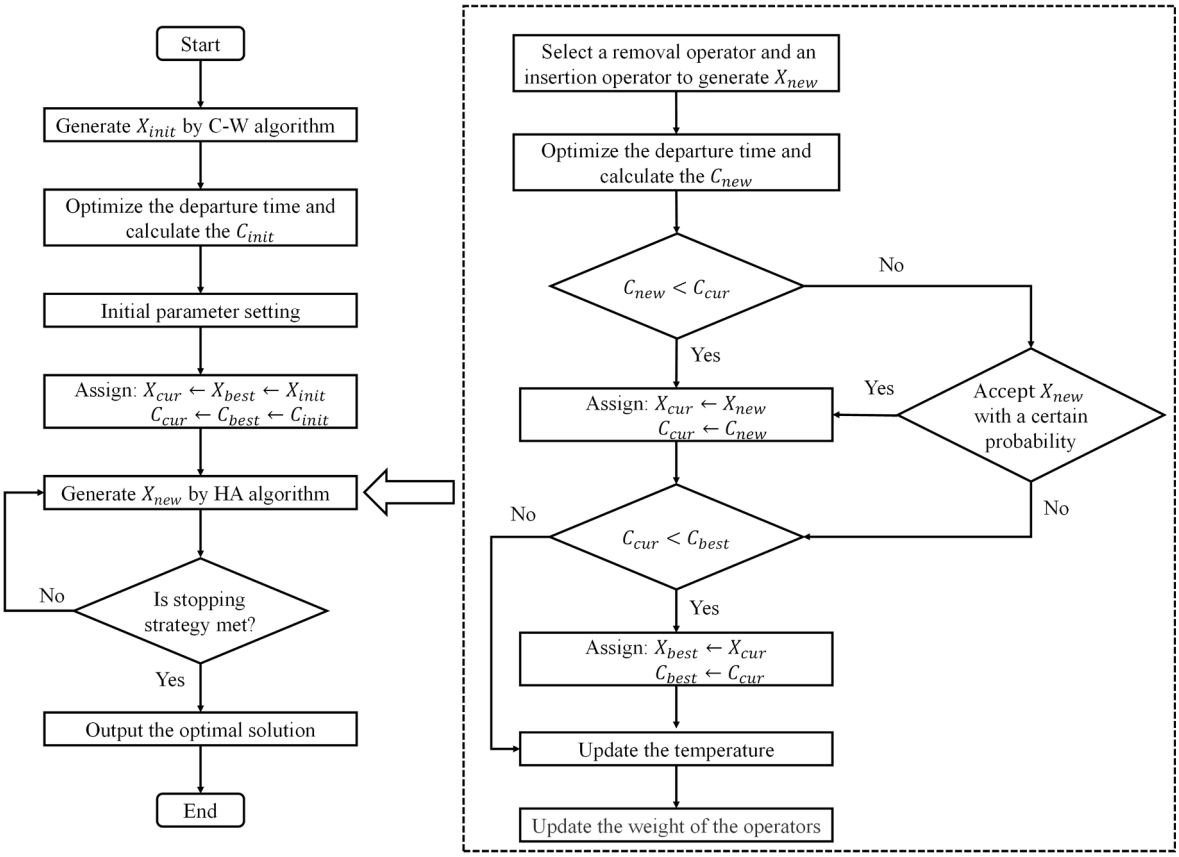


Fig. 3. Flowchart of TSHSA.

$$\sum_{m\in M}\left(w_{i0}^{m}\left(1+\frac{\tau_{i0}^{m}-\tau_{i0}^{m-1}}{b_{i0}^{m}-b_{i0}^{m-1}}\right)+z_{i0}^{m}\left(\tau_{i0}^{m-1}-b_{i0}^{m-1}\frac{\tau_{i0}^{m}-\tau_{i0}^{m-1}}{b_{i0}^{m}-b_{i0}^{m-1}}\right)\right)\leq h_i \quad \forall i\in N^{*} \tag{24}$$

According to Proposition 1, we can obtain the following linear integer programming model for the TDGVRPTW.

$$\text{minimize} \quad C_{TR}^{'}+C_{RE}+C_{EM}+C_{LA}$$

$$\text{subject to} \quad (10)-(18),(21),(23),(24)$$

5. Algorithm

To solve the above model, a two-stage hybrid search algorithm (TSHSA) is designed in this section. The general framework of TSHSA is summarized as follows: In the first stage of the algorithm, we develop an adaptive large neighborhood search (ALNS) algorithm to determine the routes of the vehicles used. Then, in the second stage of the algorithm, we design a shortest-path algorithm to optimize the departure time of the vehicle at each node. The flowchart of the proposed algorithm is given in Fig. 3.

5.1. ALNS overview

The ALNS algorithm adopted in our solving approach was first proposed by Røpke (2006) and Pisinger (2007) to solve the vehicle routing problem. It evolved from the large neighborhood search (LNS) heuristic algorithm put forward by Shaw (1998). Different from the LNS, where only one removal and insertion operator are used, the ALNS algorithm adopts a variety of removal and insertion operators that can act on the solution, and the neighborhood of the resulting solution is therefore rich and diverse. The algorithm framework and operator settings of this paper mainly refer to the methods of Demir et al., 2012 and Sun et al. (2020). In this section, we introduce the ALNS algorithm framework in detail, including the initial solution, removal operators, insertion operators, adaptive weight and score adjustment, acceptance criteria, and stopping strategy.

5.1.1. ALNS algorithm framework

The main process of the ALNS algorithm is as follows: (i) Set the value of parameters and input the initial solution. (ii) Use the roulette-wheel method to select a removal operator and an insertion operator in this iteration. (iii) The selected operators are used to remove the customer nodes and then reinsert them to obtain a new solution. (iv) Judge the quality of the newly generated solution and update the score of the selected operator. If the new solution worsens the current solution, we use the simulated annealing method to choose whether to accept the solution. (v) After every certain number (set as 100 in our algorithm) of iterations, the weight is updated according to the score of the operator, and the score is initialized. (vi) The algorithmic program stops when the number of iterations or the number of successive unpromoted solutions reach the limitation set for the algorithm.

5.1.2. Construction of the initial solution

The search algorithm begins with an initial solution, and a high-quality initial solution helps the algorithm quickly find the optimal solution to the problem. Cordeau et al. (2002) analyze some classical algorithms for initial solution generation under four evaluation index systems (including accuracy, speed, simplicity, and flexibility). The results show that the C-W algorithm (Clarke and Wright, 1964; Liu et al., 2019) has obvious advantages in generating the initial solution quickly, and since it does so using a simple method, we use the C-W algorithm to generate the initial solution.

The saving value in the traditional C-W algorithm is obtained by calculating the distance. To better adapt the model, the savings value of this paper is obtained by calculating the total cost (that is, the objective function, including the sum of transportation cost, refrigeration cost, carbon emission cost and labor cost). The basic idea of the C-W algorithm is as follows: First, connect each customer node to the distribution center to form distribution routes, each of which contains only one customer node, and calculate the total cost. Then, calculate the cost saved by merging two single-customer routes by connecting the two customer nodes. The route with the largest saving value is chosen first, and the process is repeated until the saving value is zero.

If there are other customers within the delivery range of the distribution center, and if all the constraints allow, the customers can be added to the route in sequence according to the size of the saved cost until the vehicles are fully loaded. In this way, a delivery route is formed. Then, the design of the next delivery route is carried out by the same rule. If the node j is inserted after the node i, the saving cost is calculated as follows:

$$Sav(i,j) = \left(c_{0i}^{1} + c_{i0}^{1}\right) + \left(c_{0j}^{2} + c_{j0}^{2}\right) - \left(c_{0i}^{3} + c_{ij}^{3} + c_{j0}^{3}\right),$$

where c_{ij}^{r} represents the cost from node i to node j in the route r.

5.1.3. Removal operators

In this paper, the ALNS algorithm contains eight removal operators, which are used to remove q customer nodes from the current solution, and the number of removed nodes q is controlled by parameter κ. The aim of the operator is to make drastic changes to the current solution. ALNS uses roulette-wheel selection in each iteration to select different operators to reach appropriate neighborhoods, namely, the search space.

Random removal (RR)

The RR operator randomly removes q customer nodes from the current solution. This operator often produces poor solutions (subsequently removed from consideration), but it helps the diversity of the neighborhood search and enables the algorithm to jump out of a local optimal solution.

Route removal (RoR)

Among all routes, a route is randomly chosen, and the RoR operator removes all nodes on that route except for the distribution center node.

Worst removal (WR)

This operator removes the nodes with high insertion costs. The calculation of the insertion cost is as follows:

$$C_i = D(R) - D_{-i}(R),$$

where C_i is the insertion cost of customer node i, R is the route to serve customer i, $D(R)$ is the total cost of route R, and $D_{-i}(R)$ is the total cost of route R after removing node i. This formula totals the current path cost of customer node i minus the path cost after assuming that the path removes node i. The WR operator arranges the customer node set in descending order of its insertion cost and then selects the customer nodes to remove in turn.

Worst time removal (WTR)

This operator removes the nodes with the longest waiting times. The calculation of waiting time is as follows:

$$wait_m = \max\{l_i - arr_m, 0\}.$$

Here, $wait_i$ is the waiting time of customer node i, l_i is the lower time window of node i, and arr_i is the time when the vehicle arrives at i. The WTR operator arranges customer nodes in descending order of their waiting time and then selects customer nodes to remove in turn.

Shaw removal (SR)

The core idea of this operator is to remove customer nodes that are closely related in some sense so that they can easily be reinserted into the partial solution to obtain a better solution. The SR operator first randomly selects a node i and then calculates the similarity between other customer nodes and i. The similarity is calculated using the following rule:

$$L(i,j) = \alpha d_{ij} + \beta|l_i - l_j| + \gamma\pi_{ij} + \delta|q_i - q_j|.$$

Here, $L(i,j)$ is the similarity between node i and node j, d_{ij} is the distance between node i and node j, l_i is the lower time window of node i, and q_i is the customer demand at node i. If nodes i and j are on the same path, $\pi_{ij} = -1$; otherwise, $\pi_{ij} = 1$. Parameters α, β, γ, and δ are normalized weights, where $\alpha = 0.5$, $\beta = 0.25$, $\gamma = 0.15$, and $\delta = 0.1$. This operator arranges the customer node set in ascending order according to its similarity with node i and then selects the customer nodes to remove in turn.

Proximity-based removal (PR)

Compared with the SR operator, this operator emphasizes the similarity of the distance between nodes. The PR operator can be regarded as a particular instance of the SR operator by setting $\alpha = 1, \beta = \gamma = \delta = 0$.

Time-based removal (TR)

Similarly, this operator emphasizes the similarity of the customer time window. The TR operator can be regarded as a particular instance of the SR operator by setting $\beta = 1, \alpha = \gamma = \delta = 0$.

Demand-based removal (DR)

This operator emphasizes the similarity of the customer needs between nodes. The DR operator can be regarded as a particular instance of the SR operator by setting $\delta = 1, \alpha = \beta = \gamma = 0$.

5.1.4. Insertion operators

In this paper, the ALNS algorithm contains three insertion operators. The insertion operator is performed after the execution of the removal operator is completed, and the node in the removed list is reinserted into the current solution to generate a new path.

Best time insertion (BTI)

For each subpath in the reconstruction solution, under the time window and load constraints, first find all feasible insertion positions of customer node j, and calculate the waiting time at customer node j after insertion. Then, select the position with the shortest waiting time for insertion.

Greedy insertion (GI)

Adopting the greedy algorithm idea, this operator selects the customer node and route with the smallest insertion cost in each iteration. The calculation rule of the insertion cost is as follows:

$$C_i(R) = D_{+i}(R) - D(R).$$

Here, $C_i(R)$ is the minimum insertion cost of path R after inserting node i, $D(R)$ is the total cost of the original path R, and $D_{+i}(R)$ is the total cost of path R after inserting node i. If the feasible solution cannot be found by inserting node i into path R, set $C_i(R) = \infty$. We use Ω to represent the set of all paths and define $C_i = min\{C_i(R)\}, R \in \Omega$, where C_i is the global minimum insertion cost of node i and its corresponding insertion position.

Regret insertion (RI)

The regret insertion operator is based on the greedy insertion operator and selects customer nodes for processing based on the regret value. It considers not only the best path but also the second-best path. Let $C_{i,1}$ represent the total cost change caused by inserting node i into the optimal position, and let $C_{i,2}$ represent the total cost change caused by inserting node i into the suboptimal position. We select the node with the largest regret value and insert it into the optimal position. For this operator, we define the regret value of node i as.

Table 1
ALNS parameter setting.

Notation	Description	Tuned Value
ξ	Cooling rate	0.999
S	Operator call probability update cycle	100
ψ	Roulette-wheel parameter	0.1
ω_1	Worse than the current solution	1
ω_2	Better than the current solution	3
ω_3	Better than the global optimal solution	5
ς	Destroy rate for removal operators	0.1 ~ 0.2

$$C_i^* = C_{i,2} - C_{i,1}.$$

5.1.5. Adaptive weight adjustment

When the ALNS algorithm is initialized, all removal operators and insertion operators have the same weight of 1/8 and 1/3, respectively. We divide the entire search process into many fragments, and every 100 iterations constitute a fragment. At the end of each fragment, we update the weight of the operators according to their score. The call probability update rule of the operator y is as follows:

$$P_y^{t+1} = (1-\psi)P_y^t + \frac{\psi\sigma_y}{\sum y}.$$

After every 100 iterations, the weight is updated according to the score of the operator, where ψ represents the roulette-wheel parameters, $\sum y$ represents the number of times the operator y is used in this fragment, and σ_y represents the score of the operator y in this fragment. For the new solution generated by calling an operator, the score of the operator will increase by ω_1, ω_2, or ω_3. The specific explanation is as follows: (i) ω_1: the new solution obtained worsens the current solution. (ii) ω_2: the new solution obtained improves the current solution. (iii) ω_3: a new global optimal solution is obtained.

5.1.6. Acceptance criteria and stopping strategy

This paper uses the simulated annealing method as the criterion for accepting a new solution from a certain iteration. We use X_{best} to denote the historical optimal solution, X_{cur} the current solution, and X_{new} the new solution after the removal and insertion operators are updated. The costs of solutions X_{best}, X_{cur} and X_{new} are recorded as C_{best}, C_{cur} and C_{new}, respectively. If $C_{new} \le C_{cur}$, then unconditionally accept X_{new}. If $C_{new} > C_{cur}$, then accept X_{new} with probability $e^{-(C_{new}-C_{cur})/T_{cur}}$, where T_{cur} is the current temperature, $T_{cur} = T_{init}\xi^{\varrho}$,$T_{init}$ is the initial temperature, ϱ is the number of iterations, and ξ is the cooling rate (a constant). The initial temperature is reset in each iteration, and the specific calculation is as follows:

$$T_{init} = -\rho\frac{C(X_{cur})}{ln0.5}.$$

Therefore, the setting of the initial temperature means that if the new solution is ρ worse than the current solution, the new solution still has a 50 % probability of being accepted. The parameter ρ is the initial temperature control parameter.

The stopping strategy of the ALNS algorithm in this paper is that the iteration count reaches 10,000, or 3,000 consecutive potential solutions do not produce any further improvement. In addition, the parameter values are important for the performance of the heuristic algorithm. After a large number of experiments and referring to the relevant literature (Demir et al., 2012; Zhang et al., 2020; Sun et al., 2020; Pan et al., 2021), we chose the parameter settings shown in Table 1.

5.2. Departure time optimization

In this section, we introduce the effective time window limit and transform the departure time optimization problem into a shortest path problem. It should be noted that the shortest path optimization algorithm is embedded as a subroutine into the ALNS. Whenever the ALNS algorithm generates a path, the shortest path optimization algorithm is used to optimize the departure time at each node on the path. Therefore, this section uses a path as an example to illustrate the process of departure time optimization.

5.2.1. Effective time window limits

Consider a given path: $(0,1,\cdots,n,n+1)$, where nodes 0 and $n+1$ represent distribution centers, and $1,2,\ldots,n$ represent customer nodes, and each node has a hard time window $[l_i,u_i]$ for service. For $i = 0,1,\ldots,n,n+1$, we define U_i as the effective upper time window restriction. Thus, the effective time window limits are given as follows:

$$U_{n+1} = u_{n+1},$$

$$w_{n,n+1} = U_{n+1} - T_{n,n+1}\left(w_{n,n+1}\right)$$

$$U_i = min\left\{u_i, U_{i+1} - T_{i,i+1}\left(w_{i,i+1}\right) - s_i\right\} i = n, n-1, \cdots, 1, 0. \tag{25}$$

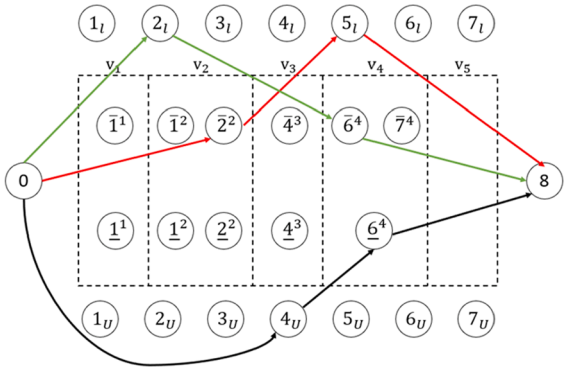


Fig. 4. Several typical feasible paths in the shortest path network.

$$w_{i,i+1} = U_{i+1} - T_{i,i+1}\left(w_{i,i+1}\right) i = n, n-1, \cdots., 1, 0.$$

In formula (25), $T_{i,i+1}\left(w_{i,i+1}\right)$ represents the traveling time between the two nodes i and $i+1$ when the departure time is $w_{i,i+1}$, $w_{i,i+1}$ represents departure time from node i to traverse arc (i,j) and s_i represents the service time at node i.

Obviously, $U_i \leq u_i$ for $i = 0, 1, \cdots., n-1, n$. Otherwise, if the arrival time of the vehicle at node i between U_i and u_i, the time window is not violated at node i, but it must be violated in the subsequent trip. In the following analysis, we use the effective time window limits $[l_i, U_i]$.

5.2.2. Shortest path formulation

To address the problem of departure time optimization for each node on the path, we first give the following two theoretical results.

For a given path $(0, 1, \cdots, n, n+1)$, the travel speed is a stepwise function $V = \{v_1, v_2, \cdots, v_p\}$. According to this speed profile, we divide the time zone into an indexed set $P = \{1, 2, \cdots, p\}$. In the optimal solution, we must find a subsequence $\{g_1, g_2, \cdots, g_{p-1}\}$ with $0 \leq g_1 \leq \cdots \leq g_{p-1} \leq n+1$, where g_k for $k = \{1, 2, \cdots p-1\}$ is the last node visited by the vehicle in period k. Thus, node 0 to node g_1 are traveled exactly at the same speed v_1 in the first period, and node g_k+1 to node g_{k+1} are traveled proceeds exactly at the same speed v_{k+1} in the $(k+1)$-th period.

Proposition 2. *All of the postservice waiting (if any) occurs at $g_k \in \{0, 1, 2, \cdots, n+1\}$, which is the last customer node for the vehicle to visit in the k-th period.*

Proposition 3. *If the vehicle has finished the delivery task at node g_k, one has the following six situations.*

(i) The postservice waiting lasts for some time, and there exists a $\delta \in \{g_k+1, g_k+2, \cdots n, n+1\}$, such that the vehicle departs exactly at the node δ at the end of a certain time period.
(ii) The postservice waiting lasts for some time, and there exists a $\delta \in \{g_k+1, g_k+2, \cdots n, n+1\}$, such that the vehicle arrives exactly at the node δ at the end of a certain time period.
(iii) *The postservice waiting lasts for some time, and there exists a $\delta \in \{g_k+1, g_k+2, \cdots n, n+1\}$, such that the vehicle arrives at the node δ strictly at its effective upper time window limit U_δ.*
(iv) *The postservice waiting lasts for some time, and there exists a $\delta \in \{g_k+1, g_k+2, \cdots n, n+1\}$, such that the vehicle arrives at the node δ strictly at its effective lower time window limit l_δ.*
(v) The postservice waiting continues until the end of the k-th period.
(vi) The vehicle departs directly after the service at node g_k, and thus the postservice waiting does not occur.

The proof of these two propositions can be found in Appendix C. With the help of Propositions 2 and 3, we can recast the departure time optimization problem as a shortest path problem to accurately find the optimal departure time of the vehicle at all nodes. We will illustrate this method in detail in the next section.

5.2.3. Numerical illustrations

From the above description in Sections 5.2.1 and 5.2.2, we can define the following six types of nodes in the shortest path problem: 0, $\bar{i}^k$, $i\_^k$, i_U, i_l, and $n+1$. Here, node 0 represents starting from the distribution center, and node $n+1$ represents returning to the distribution center. Node $\bar{i}^k$ represents that the vehicle departs exactly at the end of the k-th period (corresponding to Proposition 3 (*i*) and Proposition 3 (*v*)). Node $i\_^k$ represents that the vehicle arrives at node i exactly at the end of the k-th period (corresponding to Proposition 3 (*ii*)). Node i_U represents that the vehicle arrives at node i exactly within the effective upper time window (corresponding to Proposition 3 (*iii*)). Node i_l represents that the vehicle starts service at node i at the effective lower time window (corresponding to

Table 2
Model parameter value.

Symbol	Value	Dimension
A	8.46×10^{-6}	$l/kg \bullet km$
B	4	l/h
C	1.41×10^{-5}	$l \bullet h^2/km^3$
φ_f	7.5	RMB/l
φ_c	0.05	RMB/kg
α_{r1}	2	l/h
α_{r2}	2.5	l/h
φ_d	150	RMB
φ_q	0.05	RMB/kg

Table 3
Results on Solomon's benchmark VRPTW instances.

	ALNS		CPLEX			ACO			GA		
Instances	Obj	CPU(s)	Obj	CPU(s)	Imp(%)	Obj	CPU(s)	Imp(%)	Obj	CPU(s)	Imp(%)
C102-10D	57.1	1.7	57.1	90.7	0.00	64.3	7.4	11.2	57.1	9.2	0.00
C104-10D	56.2	1.6	56.2	372.9	0.00	64.7	7.6	13.14	56.2	8.1	0.00
C202-10D	144.6	1.7	144.6	69.4	0.00	178.3	7.4	18.9	144.6	8.3	0.00
C208-10D	146.0	1.7	146.0	73.3	0.00	170.7	7.7	14.47	146	8.3	0.00
R103-10D	229.4	1.9	229.4	151.3	0.00	269.8	7.8	14.97	229.4	8.6	0.00
R106-10D	219.6	1.8	219.6	215.9	0.00	270.6	7.7	18.85	219.6	9	0.00
R201-10D	248.9	1.7	248.9	82.1	0.00	253.9	7.9	1.97	248.9	8.5	0.00
R205-10D	209.6	1.7	209.6	349.8	0.00	241	7.5	13.03	209.6	8.3	0.00
RC102-10D	169.3	1.8	169.3	277.8	0.00	175.4	7.7	3.48	169.3	8.4	0.00
RC106-10D	177.6	1.7	177.6	154.7	0.00	191.2	7.6	7.11	177.6	8.1	0.00
RC203-10D	165.9	1.7	165.9	122.5	0.00	193.8	7.6	14.4	165.9	8.2	0.00
RC208-10D	137.5	1.7	137.5	137.6	0.00	144.9	7.5	5.11	137.5	8.9	0.00
C102-25D	190.3	4.9	190.3	7200	0.00	226.1	21.5	15.83	190.3	24.7	0.00
C104-25D	186.9	5.2	186.9	7200	0.00	216.1	21.2	13.51	186.9	25.5	0.00
C202-25D	214.7	5.6	214.7	1497.8	0.00	260.2	22	17.49	214.7	23.9	0.00
C208-25D	214.5	5.9	214.5	1683.2	0.00	231.4	22.7	7.3	214.5	22.7	0.00
R103-25D	454.6	6.3	514.3	7200	11.61	588.9	22.6	22.81	454.6	27.4	0.00
R106-25D	465.4	5.5	468.0	7200	0.56	569.9	21.3	18.34	465.4	26.9	0.00
R201-25D	463.3	5.6	463.3	2487.5	0.00	539.8	21.7	14.17	463.3	25.7	0.00
R205-25D	393.0	5.7	393.0	7200	0.00	443.1	22.6	11.31	401.5	24.5	2.12
RC102-25D	351.8	5.8	352.0	7200	0.06	430.8	21.8	18.34	351.8	24.6	0.00
RC106-25D	345.5	6.5	345.5	7200	0.00	360.3	21.9	4.11	345.5	24.3	0.00
RC203-25D	326.9	6.1	331.4	7200	1.36	416.5	22.5	21.51	326.9	25.6	0.00
RC208-25D	269.1	6.4	296.3	7200	9.18	342.7	22.2	21.48	269.1	24.8	0.00
C102-50D	361.4	19.5	361.4	7200	0.00	466.7	77.3	22.56	361.4	113.8	0.00
C104-50D	358.0	24.0	—	7200	—	435.2	76.1	17.74	358	119.9	0.00
C202-50D	360.2	23.1	360.2	7200	0.00	451.8	79.5	20.27	360.2	106.1	0.00
C208-50D	352.1	26.1	475.1	7200	25.89	429.7	78.9	18.06	352.1	104.3	0.00
R103-50D	772.9	26.6	—	7200	—	950.3	73.7	18.67	783.9	133	1.40
R106-50D	795.2	35.1	979.3	7200	18.80	987	73.2	19.43	795.2	130.7	0.00
R201-50D	791.9	20.9	841.8	7200	5.93	981	78.3	19.28	800.9	121.3	1.12
R205-50D	693.4	32.3	—	7200	—	828.7	77.8	16.33	704.3	119.5	1.55
RC102-50D	823.9	22.7	900.3	7200	8.49	1015.5	72.2	18.87	869.8	128.9	5.28
RC106-50D	723.2	23.3	—	7200	—	922.5	72.1	21.6	761.8	127.4	5.07
RC203-50D	556.4	23.8	—	7200	—	648.5	79.1	14.2	556.4	126.6	0.00
RC208-50D	480.1	24.4	—	7200	—	588.5	78.1	18.42	490.8	136.3	2.18
C102-100D	827.3	99.3	848	7200	2.44	1046.5	56	20.95	855.6	687.5	3.31
C104-100D	822.9	103.1	—	7200	—	1057.1	349.6	22.15	925.9	696.4	11.12
C202-100D	589.1	94.6	—	7200	—	764.4	352.9	22.93	598.5	605.8	1.57
C208-100D	585.8	108.1	—	7200	—	710.9	352.2	17.6	608.3	648.1	3.7
R103-100D	1213.6	110.4	—	7200	—	1563.2	349.5	22.36	1326.9	621.3	8.54
R106-100D	1249.6	98.5	—	7200	—	1636.6	349.9	23.65	1367.6	636.6	8.63
R201-100D	1178.1	103.1	—	7200	—	1346.8	352.6	12.53	1212.6	627.9	2.85
R205-100D	959.2	105.6	—	7200	—	1256	351.7	23.63	994.5	649.2	3.55
RC102-100D	1468.3	97.5	—	7200	—	1633.6	349	10.12	1592.8	645.8	7.82
RC106-100D	1409.8	106.8	—	7200	—	1692.3	344.5	16.69	1600.3	651.3	11.9
RC203-100D	941.7	101.4	—	7200	—	1269.6	340.5	25.83	957.9	699.6	1.69
RC208-100D	785.7	107.3	—	7200	—	1050.7	338.9	25.22	825	668.4	4.76

—No feasible solution found within 7200 s.

Table 4
Results comparison of different algorithms for 10-node instances.

	TSHSA		CPLEX			ALNS			ACO			GA		
Instances	TC	CPU(s)	TC	CPU(s)	Imp(%)	TC	CPU(s)	Imp(%)	TC	CPU(s)	Imp(%)	TC	CPU(s)	Imp(%)
UK1-10D	1219.01	3.59	1269.45	7200	3.97	1353.08	3.47	10.97	1369.26	7.69	10.97	1369.26	9.63	10.97
UK2-10D	1290.26	4.55	1502.35	7200	14.12	1570.49	4.02	17.84	1570.49	6.24	17.84	1570.49	10.51	17.84
UK3-10D	1214.40	3.95	1392.91	7200	12.82	1479.05	3.69	17.89	1479.05	7.27	17.89	1479.05	10.44	17.89
UK4-10D	1092.33	4.03	1293.16	7200	15.53	1402.83	3.79	22.13	1402.83	6.83	22.13	1402.83	9.93	22.13
UK5-10D	1065.15	4.85	1243.51	7200	14.34	1337.83	4.15	20.38	1337.83	6.59	20.38	1337.83	9.32	20.38
Average		4.19		7200	12.16		3.82	17.84		6.92	17.84		9.97	17.84

—No feasible solution found within 7200 s.

Table 5
Results comparison of different algorithms for 20-node instances.

	TSHSA		CPLEX			ALNS			ACO			GA		
Instances	TC	CPU(s)	TC	CPU(s)	Imp(%)	TC	CPU(s)	Imp(%)	TC	CPU(s)	Imp(%)	TC	CPU(s)	Imp(%)
UK1-20D	2201.65	10.51	2848.40	7200	22.71	2359.03	9.54	7.63	2391.11	23.96	7.92	2387.71	28.86	7.79
UK2-20D	2207.16	12.28	2585.22	7200	14.62	2431.25	10.94	9.22	2435.7	24.46	9.38	2441.7	28.12	9.61
UK3-20D	1624.17	11.96	1815.41	7200	10.53	1850.28	8.82	12.22	1869.29	23.93	13.11	1862.12	29.01	12.78
UK4-20D	2126.61	15.18	2493.70	7200	14.72	2405.12	11.59	11.58	2463.8	24.85	13.69	2459.88	27.89	13.55
UK5-20D	1935.45	11.65	2361.30	7200	18.03	2246.24	9.46	13.84	2302.92	22.74	15.96	2246.24	29.18	13.84
Average		12.32		7200	16.12		10.07	10.9		23.99	12.01		28.61	11.51

—No feasible solution found within 7200 s.

Table 6
Results comparison of different algorithms for 50-node instances.

	TSHSA		CPLEX			ALNS			ACO			GA		
Instances	TC	CPU(s)	TC	CPU(s)	Imp(%)	TC	CPU(s)	Imp(%)	TC	CPU(s)	Imp(%)	TC	CPU(s)	Imp(%)
UK1-50D	4139.63	88.64	—	7200	—	4547.76	84.79	10.41	4684.84	176.08	11.64	4653.82	226.32	11.05
UK2-50D	4211.85	95.37	—	7200	—	4621.71	86.84	8.87	4807.8	173.16	12.4	4635.21	245.19	9.13
UK3-50D	4184.18	89.25	—	7200	—	4748.36	79.78	11.88	5064.05	172.8	17.37	4812.21	241.54	13.05
UK4-50D	4881.80	104.38	—	7200	—	5208.44	96.49	6.27	5726.92	157.74	14.76	5268.42	230.38	7.34
UK5-50D	4262.39	112.85	—	7200	—	4778.08	98.04	10.79	5098.34	160.84	16.4	4798.36	227.57	11.17
Average		98.10		7200			89.19	9.64		168.12	14.51		234.2	10.35

—No feasible solution found within 7200 s.

Table 7
Results comparison of different algorithms for 75-node instances.

	TSHSA		CPLEX			ALNS			ACO			GA		
Instances	TC	CPU(s)	TC	CPU(s)	Imp(%)	TC	CPU(s)	Imp(%)	TC	CPU(s)	Imp(%)	TC	CPU(s)	Imp(%)
UK1-75D	6779.31	293.61	—	7200	—	7173.93	277.26	6.6	8213.88	359.46	17.47	7411.19	467.78	8.53
UK2-75D	6065.19	259.72	—	7200	—	6708.48	217.07	9.59	7021.56	347.89	13.62	6994.35	441.31	13.28
UK3-75D	5940.95	214.11	—	7200	—	6492.85	194.02	8.50	7311.03	370.39	18.74	6615.41	497.34	10.2
UK4-75D	5796.32	282.85	—	7200	—	6524.69	246.19	11.16	7136.3	322.68	18.78	6706.5	463.51	13.57
UK5-75D	6144.06	316.96	—	7200	—	6637.55	291.71	7.43	7429.19	326.31	17.3	6805.77	474.18	9.72
Average		273.45		7200			245.25	8.66		345.35	17.18		468.82	11.06

—No feasible solution found within 7200 s.

Table 8
Results comparison of different algorithms for 100-node instances.

	TSHSA		CPLEX			ALNS			ACO			GA		
Instances	TC	CPU(s)	TC	CPU(s)	Imp(%)	TC	CPU(s)	Imp(%)	TC	CPU(s)	Imp(%)	TC	CPU(s)	Imp(%)
UK1-100D	8388.02	644.63	—	7200	—	9140.27	607.66	8.34	10193.58	742.24	17.71	9841.95	1014.37	14.77
UK2-100D	7803.83	664.28	—	7200	—	8736.82	618.05	10.68	9980.99	755.96	21.81	9289.77	1122.16	15.99
UK3-100D	7656.99	630.76	—	7200	—	8487.39	596.53	9.78	9208.85	733.52	16.85	8697.88	1152.86	11.97
UK4-100D	7957.36	768.42	—	7200	—	8790.59	720.25	9.48	9622.56	797.47	17.31	9343.87	1024.64	14.84
UK5-100D	7855.34	794.06	—	7200	—	8701.33	695.11	9.72	9476.47	841.03	17.11	9372.64	1204.57	16.19
Average		700.43		7200			647.52	9.6		774.04	18.16		1103.72	14.75

—No feasible solution found within 7200 s.

Proposition 3 (*iv*)). Therefore, the above six kinds of nodes constitute a network, and we can optimize the departure time of each node by finding the shortest path from node 0 to node $n+1$. Appendix D shows the calculations of arc length between any two nodes.

To better describe the solution process of our model, we assume there are a distribution center and seven customer nodes. Nodes 0 and 8 indicate distribution centers, and nodes 1 to 7 indicate customers. Fig. 4 shows several typical feasible paths in the shortest path network (each colored path represents a feasible solution).

Therefore, we can find the shortest path from node 0 to node $n+1$, which is the minimum total cost of this path.

6. Numerical experiments

6.1. Parameter setting

To test the validity of the above TDGVRPTW model and TSHSA, we use the instances in the PRPLIB database (Demir, 2012) for simulation experiments. The content of the instances includes the distance between nodes, customer demands, service times, and time windows.

According to the practice of city distribution and sets the vehicle speed as shown in Fig. 1(a), this paper sets the earliest time for the vehicle to depart from the distribution center at 7:00 am and the latest time to return to the distribution center at 7:00 pm. However, the working time of the distribution center in the PRPLIB database is only 9 h, so for this database, to meet the testing requirement, we made appropriate improvements, multiplying the time window, service time, and distance between each node by 12/9. Table 2 shows the vehicle-related parameter values in the model.

The program was run using MATLAB R2020a and CPLEX 12.6. All experiments were run on personal computers with an AMD Ryzen 7 4800U CPU and 1.80 GHz memory.

6.2. Results on the VRPTW instances

In this part, to verify the quality of the TSHSA, we present the results on Solomon's benchmark VRPTW instances, which contain three types of instances: C, R, and RC. The customer coordinates of type C instances are centralized distribution, type R instances are randomly distributed, and type RC instances are mixed distribution (a small number of customer coordinates are according to a centralized distribution, and most customer coordinates are random). All 48 instances, including 10, 25, 50 and 100 customer nodes, are solved directly using the branch and bound algorithm (executed in CPLEX solver), ant colony algorithm (ACO), genetic algorithm (GA) and adaptive large neighborhood search (the first stage of the hybrid algorithm, ALNS). For example, C101-10D represents a C101 instance with 10 customer nodes. The maximum running time of CPLEX is set to 7200 s. In Table 3, we let Obj^{ALNS} (Obj^{ACO}, Obj^{GA}, Obj^{CPLEX}) represent the objective value produced by ALNS (ACO, GA, CPLEX), and *Imp*% represents the solution improvement of ALNS and other methods, which is calculated as $Imp = \left(Obj^{other} - Obj^{ALNS}\right) \Big/ Obj^{other}$. The parameters of ACO and GA are defined as follows.

ACO: The number of ants is set as 50, the maximum number of iterations is 200, and the maximum number of iterations that no longer produce better results is 60. The pheromone parameter is set as 1, the heuristic parameter as 3 and pheromone volatility as 0.15.

GA: The population size is set as 100, the maximum number of iterations is 200, and the maximum number of iterations that no longer produce better results is 60. Set the crossover probability as 0.9 and the mutation probability as 0.1.

As shown in Table 3, CPLEX provides a solution for each of 31 instances, but within the given time limit, no feasible solution is reported for the other 17 instances, and CPLEX performs very poorly in solution speed. In contrast, the ALNS algorithm can give feasible solutions for each instance in a very short time. In ten of these instances, the quality of the ALNS solution is better than that of CPLEX, with an average improvement of 8.43 %.

Compared with ACO, our ALNS algorithm can obtain better solutions in less time, with an average improvement of 16.49 %. In small-scale cases (the number of customer nodes is 10 or 25), the GA obtains the same excellent solution as the ALNS. However, in large-scale cases (the number of customer nodes is 50 or 100), the quality of the solution obtained by GA is worse than the solution obtained by ALNS. When the number of customer nodes increases to 100, the average improvement of the solution obtained by ALNS is 5.79 % better than the solution obtained by GA. In addition, the GA has a poor solution speed and is the worst of the three heuristics (ALNS, ACO, GA).

In summary, these results demonstrate the effectiveness of our proposed algorithm in solving traditional VRPTW problems.

6.3. Algorithm performance comparison

In this part, we test the effectiveness of the TSHSA for solving the model of this paper. The results are compared with the results obtained by CPLEX and traditional heuristics (ALNS, ACO, GA) without combining the departure time optimization. The maximum running time of CPLEX is set to 7200 s. We compare five types of PRP instances, calculating each instance ten times, and the result is represented by its average. In Tables 4-8, UK1-10D represents a UK1 instance with 10 customer nodes, *TC* represents the objective value (total cost), and *Imp*% represents the quality improvement between the solution obtained by TSHSA and the solution obtained by other methods, which is calculated as $Imp = \left(TC^{other} - TC^{\mathrm{TSHSA}}\right)/TC^{other}$.

For solution quality, when the vehicle route and departure time of each node should be optimized simultaneously, CPLEX can solve the problem of 10 customers and 20 customers within the specified time limit. However, due to the complexity of the problem, CPLEX

Table 9
Comparison of results with different objectives.

Instances	Z_2 TC	TTD	TTT	TCE	Z_3 TC	TTD	TTT	TCE
UK1-50D	1.157	0.953	1.099	1.371	1.081	0.978	0.913	1.212
UK1-75D	1.104	0.971	1.083	1.200	1.068	1.016	0.937	1.088
UK1-100D7	1.108	0.962	1.102	1.250	1.064	0.992	0.945	1.128
Average	1.123	0.962	1.094	1.274	1.071	0.995	0.932	1.142

Table 10
Optimal vehicle route and optimal departure time of instance UK1-50D.

Number	Vehicle route	Departure time
1	0–13–14–50–26–37–48–7–44–41-0	9:00:00–10:37:38–11:21:16–11:47:45–12:24:52–13:12:01–13:50:34–14:29:45–15:14:31–15:53:09–16:59:36
2	0–30–33–23–5–32–1–9–28–0	9:00:00–10:07:14–10:40:06–11:46:10–13:19:02–13:55:13–14:41:28–15:17:29–16:12:12–16:46:39
3	0–43–22–21–35–45–36–17–25–0	9:00:00–9:30:12–10:16:47–11:59:59–13:06:53–14:08:25–14:34:49–15:17:55–16:01:13–16:52:42
4	0–19–16–24–27–38–29–34-0	9:00:00–10:31:28–10:58:32–11:41:29–13:17:10–14:14:15–15:03:20–15:26:23–16:15:46
5	0–40–10–11–18–39–12–20–3–49-0	9:00:00–9:35:36–10:06:18–10:48:36–11:20:58–13:07:21–14:30:52–15:09:21–15:38:31–16:15:51–16:36:05
6	0–42–47–15–4–6–2–31–46–8-0	9:00:00–9:51:18–11:37:23–12:23:27–13:05:58–13:52:50–14:38:20–15:4:46–15:36:46–15:56:25–17:53:35

can only find feasible solutions, not optimal solutions. Furthermore, when the number of customers rises to 50, the solver cannot find a feasible solution within the time limit. In contrast, the traditional heuristics (ALNS, ACO, GA) only optimize the vehicle path and immediately set off to the next node after the service ends. Although a feasible solution can be found, the quality of the solution is poor. In contrast, by comparing *Imp*, we find that the quality of the solution obtained by TSHSA is higher. Compared with CPLEX, the objective value is reduced by an average of 14.14 % (see Imp in Tables 4-5), and compared with traditional heuristics, the objective value is reduced by an average of 11.92 % for ALNS, 15.94 % for ACO and 13.1 % for GA (see Imp in Tables 4-8).

For solving efficiency, the CPLEX solver can only solve the instances when there are at most 20 customer nodes. With the increase in customer nodes, a feasible solution cannot be obtained within 7200 s. The TSHSA and other heuristics are excellent in terms of solving speed. For large-scale instances with 100 customer nodes, the optimal results can be obtained in only 700.43 s (TSHSA), 647.52 s (ALNS), 774.04 s (ACO) and 1103.72 s (GA). We note that because of the lack of the departure time optimization step, ALNS is slightly faster than TSHSA, but the difference is not significant. In summary, our simulation results provide strong evidence that the TSHSA proposed in this paper can solve the TDGVRPTW model very well.

6.4. Analysis of experimental results

In this section, we verify the effectiveness of the TSHSA under different optimization objectives. Two objectives that the decision maker pays attention to are also proposed, namely, minimizing the total travel distance (Z_2) and minimizing the total travel time (Z_3). The TSHSA is used to carry out simulation experiments on the calculation instances of different customer sizes, calculating each instance ten times, and the result is represented by its average. The average is compared with the simulation result in this paper with the total cost (Z_1) minimization as the optimization objective.

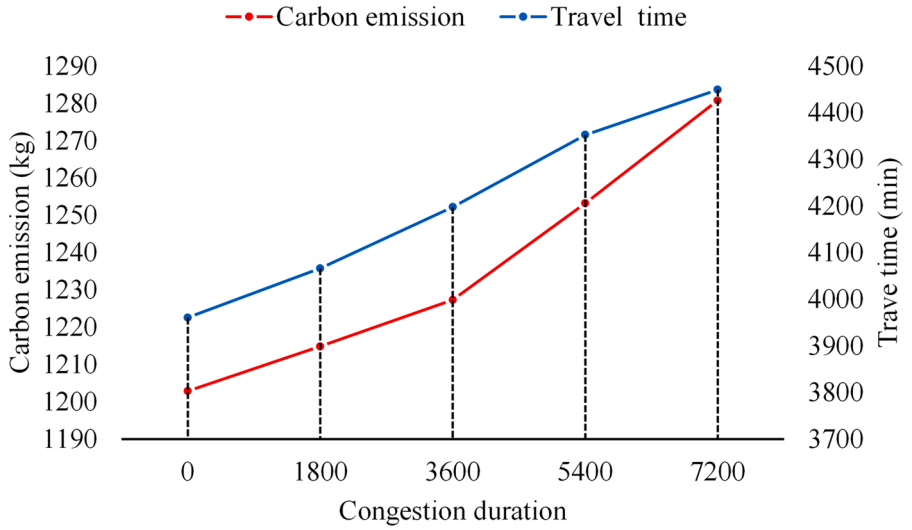
Table 9 shows four metrics: total cost (*TC*), total travel distance (*TTD*) and total travel time (*TTT*), and total carbon emissions (*TCE*). To facilitate comparison, all values are standardized with the results obtained for objective Z_1. Table 10 shows the optimal vehicle travel route and the optimal departure time at each node of instance UK1-50D when Z_1 is taken as the optimization objective.

We can draw the following conclusions: (i) Taking Z_2 and Z_3 as the optimization objectives, although a single objective can be optimized, doing so will greatly increase the cost of other objectives. In this paper, the total cost minimization as the optimization goal can achieve a better balance effect over all objectives. (ii) Compared with objective Z_1, the carbon emissions produced by objectives Z_2 and Z_3 are 27.4 % and 14.2 % higher, respectively. Therefore, carbon emissions can be significantly reduced by considering carbon emission factors in the objective function. (iii) Except for vehicle 6, all vehicles completely avoid the morning and evening peaks (7:00 am-9:00 am; 5:00 pm-7:00 pm) of urban traffic congestion, which shows that this algorithm can reasonably avoid the traffic congestion period according to customer needs, thereby shortening traveling times and effectively reducing total costs.

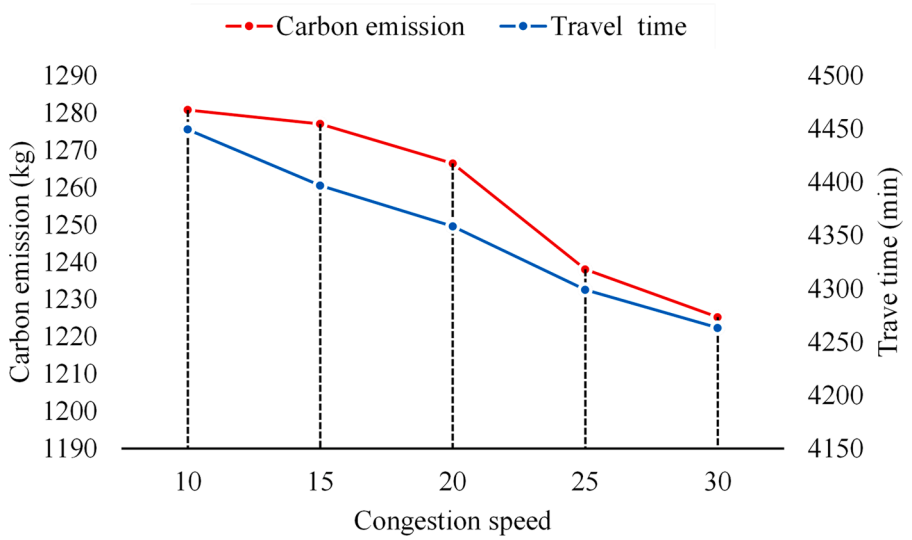
6.5. Analysis of congestion duration

In this part, a sensitivity analysis is conducted by varying the traffic congestion. Referring to Fig. 1, we regard the period when the speed is equal to or less than 30 km/h as the congestion period (7:00 am-9:00 am; 12:00 pm-2:00 pm; 5:00 pm-7:00 pm). We assume that the congestion duration varies from 0 to 7200 s, and the congestion speed (i.e., driving speed during congestion) varies from 10 km/h to 30 km/h. The UK1-100D instance is solved ten times under different congestion durations and speeds, and the result is represented by its average. The results are summarized in Fig. 5.

Fig. 5 (a) shows that carbon emissions and total travel time are positively correlated with the duration of congestion, and Fig. 5 (b) shows that they negatively correlate with congestion speed. As the duration of congestion increases or the congestion speed decreases,



(a) Simulation results under different congestion durations.



(b) Simulation results under different congestion speeds.

Fig. 5. Comparison of results with varying traffic congestion parameters.

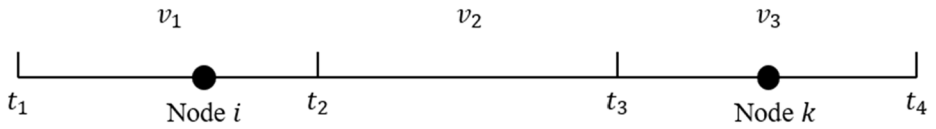


Fig. 6. Example of three time periods.

carbon emissions and the total travel time will also increase to varying degrees. Intuitively, this happens because the long congestion duration or low congestion speed will increase the proportion of low-speed travel in the whole travel cycle. This makes the fuel consumption of the vehicle increase, as do the corresponding carbon emissions.

7. Conclusion

In this paper, we combine GVRP and TDVRP to study an urban cold chain logistics distribution problem, which simultaneously

considers time-dependent traffic congestion, vehicle load, and time window constraints. To alleviate bad traffic environments, we propose a strategy (i.e., postservice waiting) in which vehicles can postpone the departure time after completing the services at each node. The model and algorithm proposed in this paper can effectively reduce carbon emissions and traffic congestion, and hence has vitally important practical value for the operation of cold chain logistics distribution.

We first modeled the stepwise travel speed function and the corresponding piecewise linear travel time function according to the actual traffic situation in the city. After that, we construct a mixed integer programming model to optimize the total distribution cost, including transportation cost, refrigeration cost, carbon emissions cost, and labor cost. These optimization costs interact and may even conflict with each other. In addition, we also need to optimize the delivery route and the departure time of each node at the same time, which greatly increases the complexity of the problem, and the traditional metaheuristic algorithm cannot solve it accurately and quickly. Therefore, we design a two-stage hybrid search algorithm (TSHSA). In the first stage of this algorithm, we develop an adaptive large neighborhood search to determine the vehicle route. In the second stage, we transform the problem of optimizing the departure time into the shortest path problem, which can accurately optimize the optimal departure time at each node. A large number of instances show that the performance of our algorithm is excellent.

By further analyzing our numerical results, the following conclusions can be obtained. First, when we solve Solomon's benchmark VRPTW instances, the results show that our ALNS algorithm is superior to CPLEX, ACO and GA regarding both solving time and quality. Second, the TSHSA we proposed can reasonably determine the vehicle route and the departure time at each node to effectively reduce the total cost. Compared with the general ILP solver CPLEX, the TSHSA is 14.14 % better than the CPLEX solution in solving small-scale instances, and the TSHSA can solve large-scale instances that CPLEX cannot. Moreover, the TSHSA solution time is shorter than that of CPLEX. Compared with other heuristics, our TSHSA is better in solving time and quality (11.92 % for ALNS, 15.94 % for ACO and 13.1 % for GA). Third, by comparing different optimization objectives, the results show that carbon emissions can be significantly reduced by considering carbon emission factors in the objective function. Fourth, we conducted a sensitivity analysis by varying the parameters of traffic congestion. The results reveal that the longer the congestion duration or the lower the congestion speed, the higher the carbon emissions and the longer the traveling time.

In future work, several meaningful directions are worth studying. First, the congestion situation in the real situation may be very complex and dynamic in time, such as breakdowns of vehicles, accidents and bad weather events. In addition, the situation of congestion will be affected by geographical space. Due to road construction, unreasonable road planning and other reasons, the vehicle speed may be associated with road segments in the road network. We believe that these congestion situations are of great research value. Second, from the perspective of model formulation, heterogeneous vehicle fleets, multiple distribution depots, and multi-objective optimization in uncertain environments could also be an attractive research direction. Finally, according to the characteristics of the model, it can be considered to propose more targeted removal and insertion operators in the ALNS algorithm, and the parameters of the ALNS algorithm can be accurately adjusted in the subsequent instance analysis, which may further improve the algorithm performance. Although it is very difficult to deal with these problems, we believe it will be of great significance to the operations of firms.

Declaration of Competing Interest

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

Data availability

The data that has been used is confidential.

Acknowledgments

The authors are grateful to the editor and four anonymous referees for their valuable comments, constructive suggestions, and encouragement. The quality of this paper was improved substantially and significantly as a result of their precious feedback. This research is supported by the National Key R&D Program of China (No. 2018YFB1601401); the National Natural Science Foundation of China (Nos. 71991464/71991460, 72271225, 71921001, 72091215/72091210); Anhui Provincial Natural Science Foundation (No. 2208085J06); the USTC Research Funds of the Double First-Class Initiative (No. YD2040002017); and the Fundamental Research Funds for the Central Universities (No. WK2040000037).

Appendix A. Notation in this paper

	Notation	Interpretation
Sets	N	Set of nodes $N = \{0, 1, \cdots, n\}$
	N^*	Set of customer nodes $N^* = N \backslash \{0\}$
	E	Set of arcs composed of all nodes

(*continued on next page*)

(*continued*)

	Notation	Interpretation
Parameters	M	Set of periods $M = \{1, 2, \cdots, m\}$
	K	Maximum number of vehicles
	Q	Vehicle capacity
	μ	Vehicle weight
	d_{ij}	Distance between arcs $(i,j) \in E$
	s_i	Service time at node $i \in N^*$
	q_i	Demands at node $i \in N^*$
	l_i	Time window's starting time at node $i \in N$
	u_i	Time window's ending time at node $i \in N$
	v_m	Speed of the vehicle in the m-th period
	A	CMEM weight module
	B	CMEM engine module
	C	CMEM speed module
	φ_f	Unit fuel consumption cost
	φ_c	Unit carbon emissions cost
	φ_d	Fixed base salary for each driver
	φ_q	Bonus for the driver to deliver unit quality goods
	α_{r1}	Fuel consumption of refrigeration equipment per unit time during transportation
	α_{r2}	Fuel consumption of refrigeration equipment per unit time during unloading
Decision variables	x_{ij}	Binary variable, $x_{ij} = 1$ if vehicle traverses arc $(i,j) \in E$, 0 otherwise
	z_{ij}^m	Binary variable, $z_{ij}^m = 1$ if vehicle traverses arc $(i,j) \in E$ and depart node i during m-th period, 0 otherwise
	w_{ij}^m	Departure time in the m-th period from node $i \in N$ to traverse arc $(i,j) \in E$;
	f_{ij}	Load of vehicle on arc $(i,j) \in A$
	r_i	Starting service time of vehicle at node $i \in N^*$
	h_i	Time to return to the distribution center on the route that has a node $i \in N^*$ as the last visited customer node

Appendix B. Sample travel data

The traveling data is collected from a path from the University of Science and Technology of China (96 Jinzhai Road) to Anhui Agricultural University (130 Changjiang Road), between which the travel distance is 3.5 km. Details are as follows:

Departure time	Required time (minute)	Speed (km/h)
7:00	11	19.09
7:30	12	17.50
8:00	14	15.00
8:30	14	15.00
9:00	12	17.50
9:30	11	19.09
10:00	11	19.09
10:30	10	21.00
11:00	11	19.09
11:30	10	21.00
12:00	11	19.09
12:30	11	19.09
13:00	11	19.09
13:30	11	19.09
14:00	9	23.33
14:30	9	23.33
15:00	8	26.25
15:30	8	26.25
16:00	9	23.33
16:30	8	26.25
17:00	9	23.33
17:30	10	21.00
18:00	12	17.50
18:30	12	17.50
19:00	11	19.09

Appendix C. Proofs

Proof of Proposition 2

Without loss of generality, we assume that the model has such an optimal routing scheme, where all postservice waiting occurs in

the last visited customer node in each period. Take the first period as an example. If the nodes at $(0, 1, \ldots, g_1 - 1)$ include postservice waiting, we can transfer all the postservice waiting that is required to node g_1. This is relatively intuitive because node 0 to node g_1 travel proceeds at the same speed, so postponing the postservice waiting to node g_1 does not increase the distribution cost. This completes the proof.

Proof of Proposition 3

Through formula (8), we know that the labor cost cost has nothing to do with time, so departure time optimization does not affect the labor cost. Therefore, the total cost discussed in this part only includes transportation cost, refrigeration cost and carbon emission cost. Therefore, the cost between the arc (i, j) can be expressed as:

$$P_{ij} = A_1(\mu + f_{ij})d_{ij} + B_1 T_{ij} + C_1 \sum_{m \in M} t_{ij}^m v_m^3 + D_1 T_{ij} \tag{26}$$

Formula (26) is derived from formulas (5), (6), and (7). The first three segments represent the transportation cost and the corresponding carbon emission cost, and the fourth segment represents the refrigeration cost and the corresponding carbon emission cost. Here, A_1, B_1, C_1 and D_1 are all constants, with $A_1 = A(\varphi_f + \partial\varphi_c)$, $B_1 = B(\varphi_f + \partial\varphi_c)$, $C_1 = C(\varphi_f + \partial\varphi_c)$, and $D_1 = \alpha_{r1}(\varphi_f + \partial\varphi_c)$. T_{ij} represents the total travel time the vehicle travels between two nodes, and t_{ij}^m represents the time the vehicle travels at speed v_m (vehicles may span multiple time periods).

Because the first segment of formula (26) is only related to distance and load, we discuss the optimization of departure time in this part, so the formula can be further simplified, and only the last three segments are considered.

To facilitate the description of the problem, we give an example. As shown in Fig. 6, the vehicle speed is v_1 in the time period (t_1, t_2), v_2 in the time period (t_2, t_3), and v_3 in the time period (t_3, t_4). The vehicle serves customer nodes $(i, \ldots, k)$, where node i is in time period (t_1, t_2) and node k is in time period (t_3, t_4). Therefore, the total cost is calculated as.

$$P = B_1 T + C_1\left(\epsilon_1 v_1^3 + \epsilon_2 v_2^3 + \epsilon_3 v_3^3\right) + D_1 T^*. \tag{27}$$

In formula (27), T represents the total travel time of the vehicle, and ϵ_1, ϵ_2, and ϵ_3 represent the travel times at speeds v_1, v_2, and v_3, respectively. T^* represents the refrigeration time of the vehicle (refrigeration is still required when serving at the customer node, so $T^* > T$).

Without considering the time window limits, we assume to wait after serving at node i. The postservice waiting time is Δt, which is a nonnegative infinitesimal. The total cost changes in the following four situations:

Case 1: If the vehicle is traveling at both time t_2 and time t_3, then the vehicle travel time in time period (t_1, t_2) is reduced by Δt, the vehicle travel time in time period (t_2, t_3) is unchanged, and the vehicle travel time in time period (t_3, t_4) is increased by $\Delta\bar{t}$.

Note that in the first time period, the reduced traveling distance of the vehicle is $v_1 \Delta t$, and in the third time period, the increased traveling distance of the vehicle must also be $v_1 \Delta t$. Therefore, $v_1 \Delta t = v_3 \Delta\bar{t}$, $\Delta\bar{t} = \frac{v_1}{v_3}\Delta t$.

Case 2: If the vehicle is serving the customer at both time t_2 and time t_3, then the vehicle travel time in time period (t_1, t_2) is reduced by Δt, the vehicle travel time in time period (t_2, t_3) is unchanged, and the vehicle travel time in time period (t_3, t_4) is increased by $\frac{v_1}{v_3}\Delta t$.

When case 1 or case 2 holds, the total cost is calculated as.

$$P = B_1\left(T - \Delta t + \frac{v_1}{v_3}\Delta t\right) + C_1\left((\epsilon_1 - \Delta t)v_1^3 + \epsilon_2 v_2^3 + \left(\epsilon_3 + \frac{v_1}{v_3}\Delta t\right)v_3^3\right) + D_1\left(T^* + \frac{v_1}{v_3}\Delta t\right). \tag{28}$$

Case 3: If the vehicle is traveling at time t_2 and serving the customer at time t_3, then the vehicle travel time in time period (t_1, t_2) is reduced by Δt, the vehicle travel time in time period (t_2, t_3) is increased by $\frac{v_1}{v_2}\Delta t$, and the vehicle travel time in time period (t_3, t_4) is unchanged.

When case 3 holds, the total cost is calculated as.

$$P = B_1\left(T - \Delta t + \frac{v_1}{v_2}\Delta t\right) + C_1\left((\epsilon_1 - \Delta t)v_1^3 + \left(\epsilon_2 + \frac{v_1}{v_2}\Delta t\right)v_2^3 + \epsilon_3 v_3^3\right) + D_1\left(T^* + \frac{v_1}{v_2}\Delta t\right). \tag{29}$$

Case 4: If the vehicle is serving the customer at time t_2 and traveling at time t_3, then the vehicle travel time in time period (t_1, t_2) is reduced by Δt, the vehicle travel time in time period (t_2, t_3) is reduced by Δt, and the vehicle travel time in time period (t_3, t_4) is increased by $\frac{v_1+v_2}{v_3}\Delta t$.

When case 4 holds, the total cost is calculated as.

$$P = B_1\left(T - 2\Delta t + \frac{v_1 + v_2}{v_3}\Delta t\right) + C_1\left((\epsilon_1 - \Delta t)v_1^3 + (\epsilon_2 - \Delta t)v_2^3 + \left(\epsilon_3 + \frac{v_1 + v_2}{v_3}\Delta t\right)v_3^3\right) + D_1\left(T^* + \frac{v_1 + v_2}{v_3}\Delta t\right). \tag{30}$$

Note that the following inferences are also tenable for three or more time periods.

Combining formulas (27)-(30), we find that the changes in total cost ΔP and the postservice waiting Δt are linearly related. Therefore, in the optimal solution, there may be the following two situations. One is that the vehicle departs exactly at the end of a certain time period (Just finished serving a customer node at the end of the time period), and the other is that the vehicle arrives at a subsequent node δ at the end of a certain time period (About to start serving a customer node at the end of the time period). Thus, the

proof of proposition 3 (i) and proposition 3 (ii) is completed.

When the time window limit is added, waiting always reduces the total cost if the service is completed at node i. As the waiting time increases until the vehicle departs at this time and arrives at a subsequent node δ strictly at its effective upper time window limit U_δ, then it is the optimal solution. Thus, the proof of proposition 3 (iii) is completed.

When the time window limit is added, if the vehicle departs immediately after completing the service at node i, and the time to arrive at a subsequent node δ is less than its lower time window l_δ, then the prewaiting will occur at node δ (If the vehicle arrives early, it cannot serve in advance and must wait until the lower time window to start the service). Therefore, the optimal solution may have such a situation: instead of making the vehicle wait in the subsequent period, it is better to wait after the service at node i. Thus, the proof of proposition 3 (iv) is completed.

If the postservice waiting at node i always reduces the total cost and waiting does not violate the time window of subsequent customers, then the optimal solution is to wait at node i until the end of the time period. Thus, the proof of proposition 3 (v) is completed.

If the postservice waiting at node i will increase the total cost, then the optimal solution is to depart immediately after completing the service at node i. Thus, the proof of proposition 3 (vi) is completed.

For the above inference, we can further expand. Compared with ordinary customer nodes, if the postservice waiting occurs at the distribution center (that is, the departure time of the vehicle delays at the distribution center), then the refrigeration cost will not be generated during the postservice waiting time at the distribution center. According to the definition of formula (6), the refrigeration cost will only occur during the vehicle delivery process. Therefore, we can regard the distribution center as a special customer node. In formulas (28)-(30), if the postservice waiting Δt occurs at the distribution center, the total cost P is calculated by subtracting $D_1 \Delta t$ (no refrigeration cost). Similarly, the changes in total cost ΔP and the postservice waiting Δt are still linearly related, and proposition 3 still holds at the distribution center. This completes the proof.

Appendix D. The calculation of arc length

The length of arc $(0, i_l)$: Take the minimum of the following three cases. (i) At the initial moment, the vehicle departs from the distribution center and arrives at node i; then, the preservice waiting takes place before the effective lower time window l_i. Intermediate nodes are strictly within the effective time window limits $[l, U)$. (ii) After waiting for a period of time in the distribution center (there is no cost to wait in the distribution center), the vehicle arrives at node i exactly at the effective lower time window l_i. Intermediate nodes are strictly within the effective time window limits $[l, U)$. (iii) At the initial moment, the vehicle departs from the distribution center, and passes through a certain subsection in the middle. Then, the postservice waiting takes place after visiting the last node of this subsection, and the vehicle arrives at node i exactly at the effective lower time window. Intermediate nodes are strictly within the effective time window limits $[l, U)$. The length of arc $\left(0, \bar{i}^k\right)$ is calculated similarly.

The length of arc $(0, i_U)$: Take the minimum of the following two cases. (i) After waiting for a period of time in the distribution center (at no cost), the vehicle arrives at node i exactly at the effective upper time window U_i. Intermediate nodes are strictly within the effective time window limits $[l, U)$. (ii) At the initial moment, the vehicle departs from the distribution center and passes through a certain subsection in the middle. Then, the postservice waiting takes place after visiting the last node of this subsection, and the vehicle arrives at node i exactly at the effective upper time window U_i. Intermediate nodes are strictly within the effective time window limits $[l, U)$. Similarly, we can calculate the length of arc $\left(0, \underline{i}^k\right)$.

The length of arc $(0, n+1)$: At the initial moment, the vehicle departs from the distribution center, and it returns to the distribution center after serving all customers on this route. Intermediate nodes are strictly within the effective time window limits $[l, U)$.

The length of arc (i_l, j_l): Take the minimum of the following two cases. (i) The vehicle starts service at node i at the effective lower time window l_i and arrives at node j; then, the preservice waiting takes place before the effective lower time window l_j. Intermediate nodes are strictly within the effective time window limits $[l, U)$. (ii) The vehicle starts service at node i at the effective lower time window l_i and passes through a certain subsection in the middle. Then, the postservice waiting takes place after visiting the last node of this subsection, and the vehicle arrives at node j exactly at the effective lower time window l_j. Intermediate nodes are strictly within the effective time window limits $[l, U)$. Similarly, we can calculate the lengths of arcs $\left(i_l, \bar{j}^k\right)$, (i_U, j_l), $\left(i_U, \bar{j}^k\right)$, $\left(\bar{i}^k, j_l\right)$, $\left(\bar{i}^k, \bar{j}^k\right)$, $\left(\underline{i}^k, j_l\right)$, and $\left(\underline{i}^k, \bar{j}^k\right)$.

The length of arc (i_l, j_U): The vehicle starts service at node i at the effective lower time window l_i and passes through a certain subsection in the middle. Then, the postservice waiting takes place after visiting the last node of this subsection, and the vehicle arrives at node j exactly at the effective upper time window U_j. Intermediate nodes are strictly within the effective time window limits $[l, U)$. Similarly, we can calculate the lengths of arcs $\left(i_l, \underline{j}^k\right)$, (i_U, j_U), $\left(i_U, \underline{j}^k\right)$, $\left(\bar{i}^k, j_U\right)$, $\left(\bar{i}^k, \underline{j}^k\right)$, $\left(\underline{i}^k, j_U\right)$, and $\left(\underline{i}^k, \underline{j}^k\right)$.

The length of arc $(i_l, n+1)$: The vehicle starts service at node i at the effective lower time window l_i, and it returns to the distribution center after serving all customers on this route. Intermediate nodes are strictly within the effective time window limits $[l, U)$. Similarly, we can calculate the lengths of arcs $(i_U, n+1)$, $\left(\bar{i}^k, n+1\right)$, and $\left(\underline{i}^k, n+1\right)$.

It is worth noting that if travel is impossible between two nodes, we set the length between the two nodes to infinity.

References

Allahyari, S., Yaghoubi, S., Van Woensel, T., 2021. The secure time-dependent vehicle routing problem with uncertain demands. Comput. Oper. Res. 131, 105253.
Babagolzadeh, M., Shrestha, A., Abbasi, B., Zhang, Y., Woodhead, A., Zhang, A., 2020. Sustainable cold supply chain management under demand uncertainty and carbon tax regulation. Trans. Res. Part D: Trans. Environ. 80, 102445.
Barth, M., Younglove, T., Scora, G., 2005. Development of a Heavy-Duty Diesel Modal Emissions and Fuel Consumption Model. Institute of Transportation Studies University of California, Berkely.
Bektaş, T., Laporte, G., 2011. The Pollution-Routing Problem. Trans. Res. Part B: Methodol. 45 (8), 1232–1250.
Chen, J., Liao, W., Yu, C., 2021. Route optimization for cold chain logistics of front warehouses based on traffic congestion and carbon emission. Comput. Ind. Eng. 161, 107663.
Chen, J., Gui, P., Ding, T., Na, S., Zhou, Y., 2019. Optimization of transportation routing problem for fresh food by improved ant colony algorithm based on tabu search. Sustainability 11 (23), 6584.
Çimen, M., Soysal, M., 2017. Time-dependent green vehicle routing problem with stochastic vehicle speeds: an approximate dynamic programming algorithm. Trans. Res. Part D: Trans. Environ. 54, 82–98.
Clarke, G., Wright, J.W., 1964. Scheduling of vehicles from a central depot to a number of delivery nodes. Oper. Res. 12 (4), 568–581.
Cordeau, J.-F., Gendreau, M., Laporte, G., Potvin, J.-Y., Semet, F., 2002. A guide to vehicle routing heuristics. J. Operational Res. Soc. 53 (5), 512–522.
Dai, J., Che, W., Lim, J.J., Shou, Y., 2020. Service innovation of cold chain logistics service providers: a multiple-case study in China. Ind. Mark. Manage. 89, 143–156.
Dantzig, G., Ramser, J., 1959. The Truck Dispatching Problem. Manage. Sci. 6 (1), 80–91.
Dekker, R., Bloemhof, J., Mallidis, I., 2012. Operations Research for green logistics – an overview of aspects, issues, contributions and challenges. Eur. J. Oper. Res. 219 (3), 671–679.
Demir, E., Bektaş, T., Laporte, G., 2011. A comparative analysis of several vehicle emission models for road freight transportation. Trans. Res. Part D: Trans. Environ. 16 (5), 347–357.
Demir, E., Bektaş, T., Laporte, G., 2012. An adaptive large neighborhood search heuristic for the Pollution-Routing Problem. Eur. J. Oper. Res. 223 (2), 346–359.
Demir, E., Bektaş, T., Laporte, G., 2014. A review of recent research on green road freight transportation. Eur. J. Oper. Res. 237 (3), 775–793.
Erdoğan, S., Miller-Hooks, E., 2012. A green vehicle routing problem. Trans. Res. Part E: Logistics and Trans. Rev. 48 (1), 100–114.
Fan, H., Zhang, Y., Tian, P., Lv, Y., Fan, H., 2021. Time-dependent multi-depot green vehicle routing problem with time windows considering temporal-spatial distance. Comput. Oper. Res. 129, 105211.
Florio, A.M., Absi, N., Feillet, D., 2021. Routing electric vehicles on congested street networks. Trans. Sci. 55 (1), 238–256.
Fontaine, P., 2022. The vehicle routing problem with load-dependent travel times for cargo bicycles. Eur. J. Oper. Res. 300 (3), 1005–1016.
Franceschetti, A., Honhon, D., Van Woensel, T., Bektaş, T., Laporte, G., 2013. The time-dependent pollution-routing problem. Trans. Res. Part B: Methodol. 56, 265–293.
Franceschetti, A., Demir, E., Honhon, D., Van Woensel, T., Laporte, G., Stobbe, M., 2017. A metaheuristic for the time-dependent pollution-routing problem. Eur. J. Oper. Res. 259 (3), 972–991.
Franceschetti, A., Honhon, D., Laporte, G., Van Woensel, T., 2018. A shortest-path algorithm for the departure time and speed optimization problem. Trans. Sci. 52 (4), 756–768.
Ghannadpour, S.F., Zarrabi, A., 2019. Multi-objective heterogeneous vehicle routing and scheduling problem with energy minimizing. Swarm Evol. Comput. 44, 728–747.
Gmira, M., Gendreau, M., Lodi, A., Potvin, J.-Y., 2021. Tabu search for the time-dependent vehicle routing problem with time windows on a road network. Eur. J. Oper. Res. 288 (1), 129–140.
Huang, Y., Zhao, L., Van Woensel, T., Gross, J.-P., 2017. Time-dependent vehicle routing problem with path flexibility. Trans. Res. Part B: Methodol. 95, 169–195.
Ichoua, S., Gendreau, M., Potvin, J.Y., 2003. Vehicle dispatching with time-dependent travel times. Eur. J. Oper. Res. 144 (2), 379–396.
Kwon, Y.-J., Choi, Y.-J., Lee, D.-H., 2013. Heterogeneous fixed fleet vehicle routing considering carbon emission. Trans. Res. Part D: Trans. Environ. 23, 81–89.
Li, Y., Soleimani, H., Zohal, M., 2019. An improved ant colony optimization algorithm for the multi-depot green vehicle routing problem with multiple objectives. J. Cleaner Prod. 227, 1161–1172.
Li, X., Zhou, K., 2021. Multi-objective cold chain logistic distribution center location based on carbon emission. Environ. Sci. Pollut. Res. 28 (25), 32396–32404.
Lin, B., Chen, Y., 2020. Will land transport infrastructure affect the energy and carbon dioxide emissions performance of China's manufacturing industry? Appl. Energy 260, 114266.
Liu, B., Guo, X., Yu, Y., Zhou, Q., 2019. Minimizing the total completion time of an urban delivery problem with uncertain assembly time. Trans. Res. Part E: Logistics and Trans. Rev. 132, 163–182.
Liu, G., Hu, J., Yang, Y., Xia, S., Lim, M.K., 2020b. Vehicle routing problem in cold chain logistics: a joint distribution model with carbon trading mechanisms. Resour. Conserv. Recycl. 156, 104715.
Liu, C., Kou, G., Zhou, X., Peng, Y., Sheng, H., Alsaadi, F.E., 2020a. Time-dependent vehicle routing problem with time windows of city logistics with a congestion avoidance approach. Knowl.-Based Syst. 188, 104813.
Malandraki, C., Daskin, M.S., 1992. Time dependent vehicle routing problems: formulations, properties and heuristic algorithms. Trans. Sci. 26 (3), 185–200.
Mansourianfar, M.H., Gu, Z., Saberi, M., 2022. Distance-based time-dependent optimal ratio control scheme (TORCS) in congested mixed autonomy networks. Trans. Res. Part C: Emerging Technol. 141, 103760.
Meneghetti, A., Ceschia, S., 2019. Energy-efficient frozen food transports: the refrigerated routing problem. Int. J. Prod. Res. 58 (14), 4164–4181.
Pan, B., Zhang, Z., Lim, A., 2021. Multi-trip time-dependent vehicle routing problem with time windows. Eur. J. Oper. Res. 291 (1), 218–231.
Pisinger, D., Ropke, S., 2007. A general heuristic for vehicle routing problems. Comput. Oper. Res. 34 (8), 2403–2435.
Raeesi, R., Zografos, K.G., 2019. The multi-objective steiner pollution-routing problem on congested urban road networks. Trans. Res. Part B: Methodol. 122, 457–485.
Rauniyar, A., Nath, R., Muhuri, P.K., 2019. Multi-factorial evolutionary algorithm based novel solution approach for multi-objective pollution-routing problem. Comput. Ind. Eng. 130, 757–771.
Rincon-Garcia, N., Waterson, B., Cherrett, T.J., Salazar-Arrieta, F., 2020. A metaheuristic for the time-dependent vehicle routing problem considering driving hours regulations – an application in city logistics. Trans. Res. Part A: Policy and Practice. 137, 429–446.
Ropke, S., Pisinger, D., 2006. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. Trans. Sci. 40 (4), 455–472.
Scora, G., Barth, M., 2006. Comprehensive modal emission model (CMEM), version 3.01, user's guide. University of California, Riverside. Technical report.
Shaw, P., 1998. In: Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems. Springer, pp. 417–431.
Stellingwerf, H.M., Kanellopoulos, A., van der Vorst, J.G.A.J., Bloemhof, J.M., 2018. Reducing CO2 emissions in temperature-controlled road transportation using the LDVRP model. Trans. Res. Part D: Trans. Environ. 58, 80–93.
Sun, P., Veelenturf, L.P., Dabia, S., Van Woensel, T., 2018. The time-dependent capacitated profitable tour problem with time windows and precedence constraints. Eur. J. Oper. Res. 264 (3), 1058–1073.

Sun, P., Veelenturf, L.P., Hewitt, M., Van Woensel, T., 2020. Adaptive large neighborhood search for the time-dependent profitable pickup and delivery problem with time windows. Trans. Res. Part E: Logistics and Trans. Rev. 138, 101942.
Suzuki, Y., 2016. A dual-objective metaheuristic approach to solve practical pollution routing problem. *Int. J. Production Economics*. 176, 143–153.
Toro, E., Franco, J., Echeverri, M., Guimarães, F., 2017. A multi-objective model for the green capacitated location-routing problem considering environmental impact. Comput. Ind. Eng. 110, 114–125.
Visser, T.R., Spliet, R., 2020. Efficient Move Evaluations for time-dependent vehicle routing problems. Trans. Sci. 54 (4), 1091–1112.
Vu, D.M., Hewitt, M., Boland, N., Savelsbergh, M., 2020. Dynamic discretization discovery for solving the time-dependent traveling salesman problem with time windows. Trans. Sci. 54 (3), 703–720.
Wang, Y., Assogba, K., Fan, J., Xu, M., Liu, Y., Wang, H., 2019. Multi-depot green vehicle routing problem with shared transportation resource: Integration of time-dependent speed and piecewise penalty cost. J. Cleaner Prod. 232, 12–29.
Wang, Z., Leng, L., Wang, S., Li, G., Zhao, Y., 2020. A Hyperheuristic Approach for location-routing problem of cold chain logistics considering fuel consumption. Comput. Intelligence and Neurosci. 1–17.
Xiao, Y., Zhao, Q., Kaku, I., Xu, Y., 2012. Development of a fuel consumption optimization model for the capacitated vehicle routing problem. Comput. Oper. Res. 39 (7), 1419–1431.
Xiao, Y., Zuo, X., Huang, J., Konak, A., Xu, Y., 2020. The continuous pollution routing problem. Appl. Math. Comput. 387, 125072.
Xu, Z., Elomri, A., Pokharel, S., Mutlu, F., 2019. A model for capacitated green vehicle routing problem with the time-varying vehicle speed and soft time windows. Comput. Ind. Eng. 137, 106011.
Yao, Y., Zhao, X., Zhang, Y., Chen, C., Rong, J., 2020. Modeling of individual vehicle safety and fuel consumption under comprehensive external conditions. Trans. Res. Part D: Transport and Environ. 79, 102224.
Yao, Y., Zhu, X., Dong, H., Wu, S., Wu, H., Tong, L.C., Zhou, X., 2019. ADMM-based problem decomposition scheme for vehicle routing problem with time windows. Trans. Res. Part B: Methodol. 129, 156–174.
Zeng, W., Miwa, T., Morikawa, T., 2020. Eco-routing problem considering fuel consumption and probabilistic travel time budget. Trans. Res. Part D: Transport and Environ. 78, 102219.
Zhang, R., Guo, J., Wang, J., 2020. A time-dependent electric vehicle routing problem with congestion tolls. IEEE Trans. Eng. Manage. 69 (4), 861–873.
Zhen, L., Xu, Z., Ma, C., Xiao, L., 2019. Hybrid electric vehicle routing problem with mode selection. Int. J. Prod. Res. 58 (2), 562–576.
Zhou, Z., Roncoli, C., 2022. A scalable vehicle assignment and routing strategy for real-time on-demand ridesharing considering endogenous congestion. Trans. Res. Part C: Emerging Technol. 139, 103658.
Zhu, L., Hu, D., 2018. Study on the vehicle routing problem considering congestion and emission factors. Int. J. Prod. Res. 57 (19), 6115–6129.