Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2025.069139

ARTICLE

Solving Multi-Depot Vehicle Routing Problems with Dynamic Customer Demand Using a Scheduling System TS-DPU Based on TS-ACO

Tsu-Yang Wu[1], Chengyuan Yu[1], Yanan Zhao[2], Saru Kumari[3] and Chien-Ming Chen[1,*]

[1]School of Artificial Intelligence/School of Future Technology, Nanjing University of Information Science and Technology, Nanjing, 210044, China

[2]School of Transportation Science and Engineering, Beihang University, Beijing, 100191, China

[3]Department of Mathematics, Chaudhary Charan Singh University, Meerut, 250004, Uttar Pradesh, India

*Corresponding Author: Chien-Ming Chen. Email: chienmingchen@ieee.org

Received: 16 June 2025; Accepted: 21 November 2025; Published: 12 January 2026

ABSTRACT: With the increasing complexity of logistics operations, traditional static vehicle routing models are no longer sufficient. In practice, customer demands often arise dynamically, and multi-depot systems are commonly used to improve efficiency. This paper first introduces a vehicle routing problem with the goal of minimizing operating costs in a multi-depot environment with dynamic demand. New customers appear in the delivery process at any time and are periodically optimized according to time slices. Then, we propose a scheduling system TS-DPU based on an improved ant colony algorithm TS-ACO to solve this problem. The classical ant colony algorithm uses spatial distance to select nodes, while TS-ACO considers the impact of both temporal and spatial distance on node selection. Meanwhile, we adopt Cordeau's Multi-Depot Vehicle Routing Problem with Time Windows (MDVRPTW) dataset to evaluate the performance of our system. According to the experimental results, TS-ACO, which considers spatial and temporal distance, is more effective than the classical ACO, which only considers spatial distance.

KEYWORDS: Dynamic vehicle routing; multiple depots; ant colony optimization; temporal-spatial distance; time slice

1 Introduction

Logistics is essential to connect trade and business activities worldwide with high efficiency in today's society. As the logistics industry moves towards intelligence, digitalization, and automation in the era of booming technology, companies aim to establish an effective vehicle dispatching system to reduce operating costs. In this context, the vehicle routing problem has gained significant attention as a crucial aspect of logistics services. In the classical vehicle routing problem (VRP), a set of vehicles is dispatched from a central depot to serve a predetermined group of customers. All information about the routing network and the customer base is known in advance. As the logistics industry continues to evolve, numerous VRP extensions have been proposed to address more practical constraints. These include the Capacitated VRP (CVRP) [1,2], the VRP with Time Windows (VRPTW) [3,4], and Multi-Depot VRP (MDVRP) [5], all of which are categorized as Static VRP (SVRP) [6].

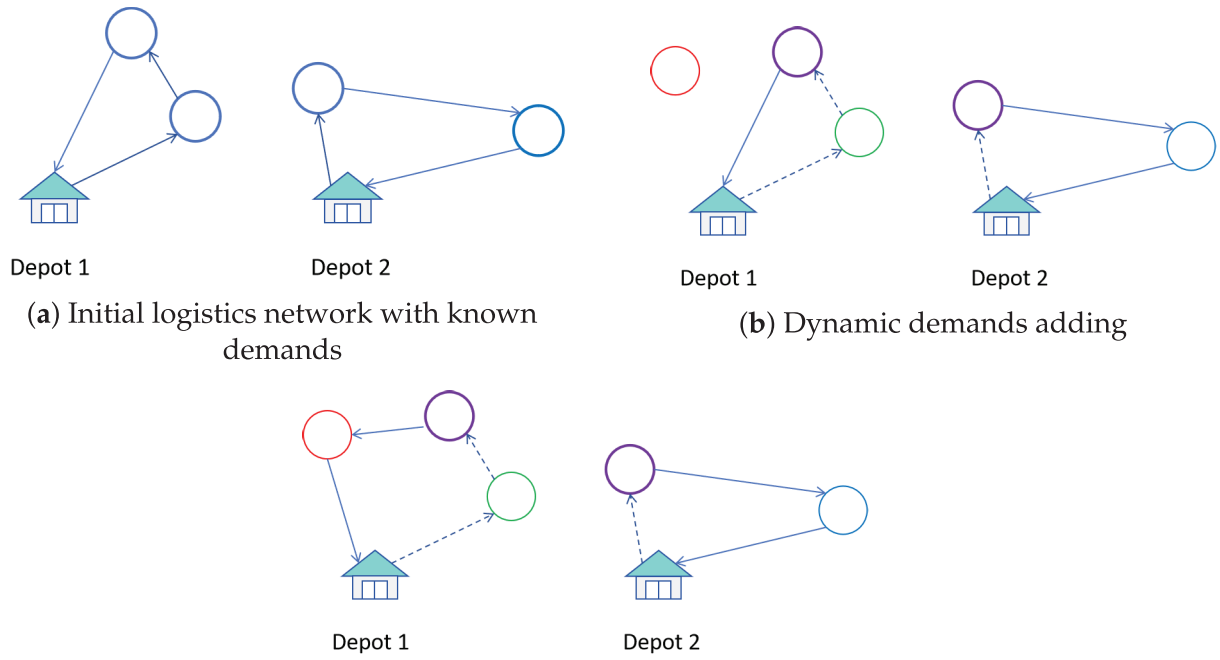
According to [7], a dynamic path problem (DARP) was initially proposed for a single vehicle, in which dynamic requests occur randomly between the start and end nodes. Psaraftis [8] proposed a periodic optimization strategy for dynamic demands but only for small-scale data, and he summarized the dynamic path problems and distinguished them from static path problems. Powell et al. [9] then classified methods

for solving dynamic demand into two categories. The first category is a priori optimization, where dynamic demand is fitted based on probabilistic methods before planning begins to create a distribution network that meets future expectations. The second category is multi-stage optimization, in which the demand is unpredictable and therefore requires a high dynamic processing capability to cope with unexpected demand.

However, due to the advancement of communication technology, customers can now provide their demands to logistics companies at any time, leading to the constant submission of orders during the distribution process. This dynamic customer demand, along with the constraints of the distribution process caused by the real situation's special circumstances, gives rise to a dynamic vehicle routing problem (DVRP). The factors driving the dynamics are broadly classified into two categories [10]: dynamic customer demand and dynamic environment. Savelsberg and Sol [11] proposed a dynamic routing of independent vehicles method and a branch-and-price algorithm, which uses real data to simulate dynamic environments for testing, effectively reducing operating costs. Gendreau et al. [12] proposed a parallel taboo search for solving dynamic versions of courier service applications, which responds to and processes new customer demand as soon as it is available. Kilby et al. [13] dividing each workday into time slots and inserting dynamic customer demand into the appropriate location within each time slot. Montemanni et al. [14] introduced an ant colony-based approach to handle dynamic customer demand by segmenting the workday into discrete time slices and using a re-optimization method to solve the whole again. Potvin et al. [15] investigated a dynamic vehicle routing and scheduling issue incorporating time window constraints and compared the advantages and disadvantages between different scheduling strategies by considering a DVRP that combines two dynamic variables: real-time customer requests and dynamic travel times. Azi et al. [16] proposed a heuristic approach based on adaptive large neighborhood search to address vehicle routing scenarios involving multiple delivery routes, where customer demands arise dynamically and require immediate response. de Armas and Melián-Batists [17] introduced a metaheuristic based on variable neighborhood search to tackle dynamic vehicle routing problems constrained by time windows, and it has been used by a Spanish company. Jia et al. [18] designed a new scheduling system that combines the PSO algorithm with periodic optimization to address the dynamic capacitated VRP (DCVRP), using a region partitioning approach to simplify the problem and solving the subproblems in parallel. Xiang et al. [19] proposed an ant colony algorithm(ACO) based on pairwise proximity learning, called PPL-ACO, for dealing with the DVRP considering the nodal relationships during the cycle period in periodic optimization. Pan and Liu [20] propose GENM-A3C, a graph-POMDP-DRL framework that yields near-optimal routes in milliseconds under dynamic, uncertain, and partially observable conditions. Sze et al. [21] proposed a two-stage AVNS that embeds critical nodes to overcome the traditional AVNS's inability to handle dynamic accidents and impractical diversion constraints, thereby significantly reducing DVRP delays. Demirbilek [22] introduced Multi-Planning with Acception/Rejection Policy (MPA) and Multi-Planning with Mandatory Assignment Policy (MPM) for DVRP. Results show that MPA outperforms under tighter time windows and high-demand settings, while MPM demonstrates robust performance across broader conditions.

Up to now, most research on the DVRP assumes that service is carried out by a single depot [19,23–25]. In practice, logistics networks typically rely on multiple depots to optimize operations. A multi-depot structure enhances demand responsiveness and reduces costs by sharing resources across depots. Therefore, single-depot VRP models are insufficient for addressing real-world logistics challenges [26]. By combining a multi-depot structure into the DVRP, vehicles can be pre-positioned near real-time demand hotspots, thereby reducing response times at the same time, shared inventory and pooled fleets across depots absorb demand surges and mitigate delay risks. In turn, reduced empty-running distances translate into lower operating costs. Consequently, existing studies on DVRP are not effectively applicable in current scenarios.

In this paper, we first introduce a vehicle routing problem with time windows, which combines the multi-depot environment and dynamic customer demand, named Multi-depot and Dynamic VRP with Time Windows (MD-DVRP). Fig. 1 depicted a conception of this problem. At the outset of the day's delivery task, some accepted orders already exist (non-red circles), including those unfulfilled from the previous day and those received the day before the delivery task began. As shown in Fig. 1a, by first making arrangements for these known orders, the routes indicated by the arrows in the figure can satisfy all currently known demands, where the dashed arrows represent routes already completed at the current moment. Over time, dynamic orders (red circles) will continue to be received until the delivery time ends. As can be seen in Fig. 1b, the green, blue, and purple circles represent the current completed orders, the current unfinished orders, and the current vehicle locations, respectively. Fig. 1c presents the adjusted routes that satisfy dynamic orders. The goal of the MD-DVRP is to dynamically adjust and optimize vehicle travel routes based on real-time changes in various information, to achieve efficiency and economy in logistics distribution.



(**a**) Initial logistics network with known demands

(**b**) Dynamic demands adding

(**c**) Final logistics network with known demands + dynamic demands

Figure 1: The conception of MD-DVRP

Based on the conception of MD-DVRP provided, it can be inferred that MD-DVRP has repeating elements over time, such as customers and partial routes. Thus, we adopt the ACO algorithm because it retains memory of what worked well before by the pheromone. It can use pheromones from previous time slices to guide the route plan when new orders are received in a later time slice, whereas GA must re-initialize its entire population and TS must completely reset its tabu list, so neither competitor can rapidly reuse prior experience. Moreover, by constructing routes probabilistically edge-by-edge, ACO is able to insert newly arrived customers in real time without disrupting the existing route backbone, while GA's crossover and mutation operators and TS's neighborhood moves typically trigger large-scale route reconstructions that markedly increase computational overhead. So ACO may make finding good delivery plans faster and easier.

To solve MD-DVRP, we proposed a scheduling system, TS-DPU. It centers on a dynamic processing unit and addresses the MD-DVRP in the following steps. Initially, it captures dynamic customer demands in real time. To address dynamic customer demand, we first adopt a time-slicing strategy [13] by partitioning the entire day into equal intervals. Subsequently, at the end of each time interval, our system obtains the positions of all vehicles and the completed orders. Then, it combines incomplete orders with current vehicle locations to form static problems for different depots. An improved ACO algorithm, named TS-ACO, is proposed to optimize vehicle routes and assign the results to the corresponding vehicles for each static problem. Finally, we conduct several simulations to evaluate our system. The results showed that our system achieved great results. By flexibly converting dynamic problems into static ones and solving them efficiently, this system effectively responds to dynamic distribution scenarios, enhancing the timeliness and adaptability of vehicle routing optimization.

Our main contributions are summarized as follows.

(1) We first introduce a vehicle routing problem with time windows, which combines the multi-depot environment with dynamic customer demand, MD-DVRP. MD-DVRP can better match today's complex logistics than DVRP.
(2) We propose a scheduling system, TS-DPU, which transforms our proposed MD-DVRP into several static problems. TS-DPU uses proven techniques for low-cost, high-quality solutions.
(3) We propose an improved ACO algorithm with spatio-temporal distances, TS-ACO, and apply it to our TS-DPU. It achieves better results in the MD-DVRP compared to the classical ACO algorithm.

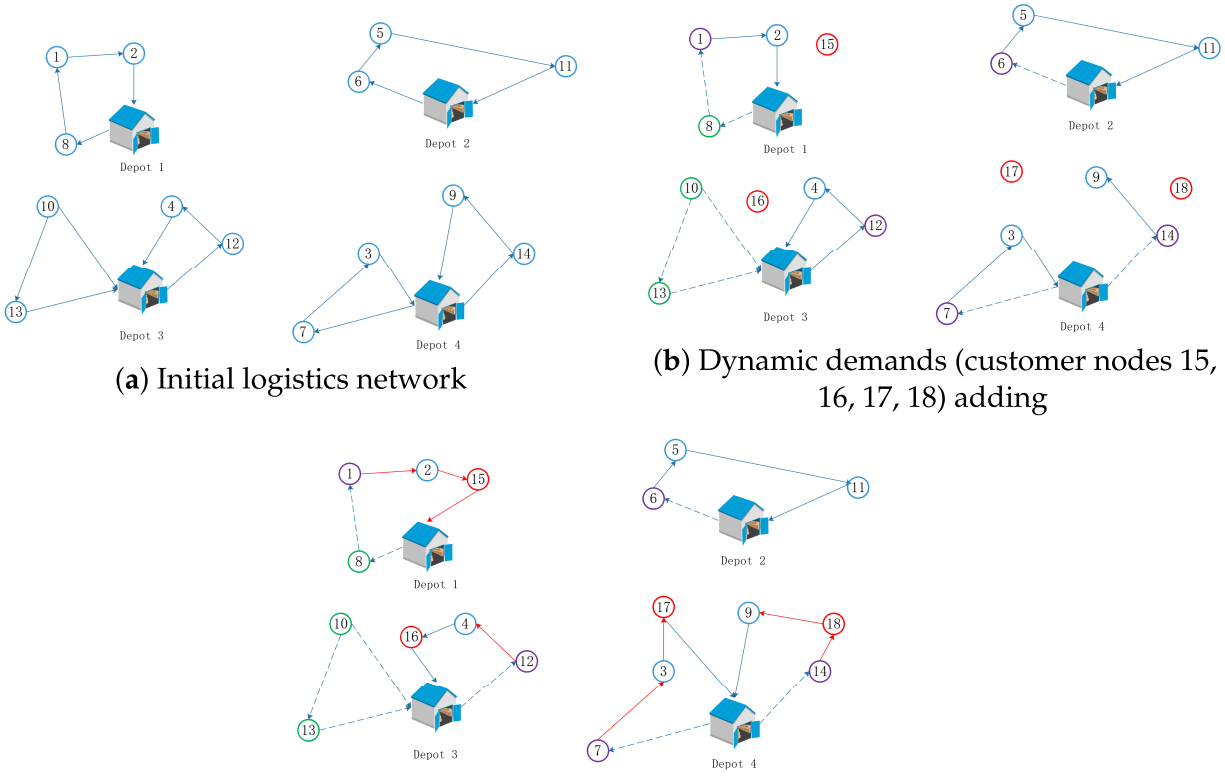
The remainder of this paper is structured as follows. Section 2 provides a detailed description of the proposed MD-DVRP, along with its corresponding mathematical formulation. Section 3 outlines the general procedure and implementation of the scheduling system TS-DPU proposed in this study. Section 4 conducts a parameter sensitivity analysis of the improved ACO algorithm, TS-ACO, using the Cordeau instance and compares its performance against the classical ACO algorithm in 19 instances. Section 5 concludes the paper by summarizing the key findings and contributions, and also discusses the limitations of the study.

2 Problem Definition and Mathematical Modeling

2.1 The Definition of MD-DVRP

The classical VRP is defined on an undirected graph $G = (V, E)$, where the set of nodes $V = V_0, V_1, V_2, ..., V_n$ includes a central depot V_0 and customer nodes V_1 through V_n. The edge set $E = i, j \mid i, j \in V$ represents the connections between nodes, each associated with a spatial distance. Every customer node i is assigned a demand q_i and a soft delivery time window denoted by $[ET_i, LT_i]$. DVRP turns the static undirected graph G into a dynamic undirected graph $G^t = (V^t, E^t)$, which means that when a customer demand is suddenly submitted to the depot or modified, the undirected graph adds this customer node and all the arcs with this customer node as the end node, and the undirected graph is changed.

MD-DVRP is an extension of the above unique depot V_0 into multiple depots $D = \{D_1, D_2, ..., D_n\}$, then MD-DVRP is defined as a dynamic undirected graph $G^t = (V^t, E^t)$, where $V^t = \{D, V_1, V_2, ..., V_n\}, E^t = \{i, j | i, j \in V^t\}$. Here in Fig. 2 as an example, the red customer nodes 15, 16, 17, 18 in Fig. 2b indicate the new demand submitted to the depot from Fig. 2a in the current time slice. If the original route cannot be modified, logistics enterprises need an additional vehicle to pick up the new route node for additional service. In other words, Depot 1 and Depot 3 need to add one vehicle to serve the new customer nodes 15 and 16; Depot 4 further needs two additional vehicles to serve the new customer nodes 17 and 18. This situation will greatly increase the overall operating costs, as a vehicle to serve a single customer will also make the vehicle utilization rate low.



(**a**) Initial logistics network

(**b**) Dynamic demands (customer nodes 15, 16, 17, 18) adding

(**c**) Final logistics network

Figure 2: An example of MD-DVRP

Therefore, in order to improve vehicle utilization and reduce operating costs, it is a critical issue to modify the original route to add new customer nodes to the original route. As shown in Fig. 2c, adding node 16 to the original route $R = [D_2, 12, 4, D_2]$, then $R' = [D_2, 12, 4, 16, D_2]$, and similarly adding customer node 17 and customer node 18 to the corresponding routes to get brand new routes $[D_4, 7, 3, 17, D_4]$ and $[D_4, 14, 18, 9, D_4]$, respectively, without Additional new routes can serve additional customers, which will greatly improve the utilization of vehicles, effectively improve operational effectiveness of each depot, greatly reduce operating cost and significantly improve the profit of the enterprise.

The symbols and parameter definitions employed throughout the paper are provided in Table 1.

Table 1: Parameter description

Parameters	Definitions
n_{ts}	Number of time slices
D	The set of depots
C^n/E^n	All known customers/edges in the n-th time slice
E^n	All known edges in the n-th time slice
V^n	All known nodes within the n-th time slice, $V^n = D \cup C^n$.
C/E	All known customers/edges in the n_{ts}-th time slice

(Continued)

Table 1 (continued)

Parameters	Definitions
E	All known edges in the n_{ts}-th time slice
V	All known nodes in the n_{ts}-th time slice, $V = D \cup C$.
$d_{ij}^s/d_{ij}^t/d_{ij}$	Spatial/Temporal/Temporal-Spatial distance between node i and node j
d_{ij}^t	Temporal distance between node i and node j
d_{ij}	Temporal-Spatial distance between node i and node j
K	Set comprising all vehicles
Q	Load-carrying limit of the vehicle
q_i	Delivery quantity of customer i
$[ET_i, LT_i]$	Time window for node i
$start\_time/end\_time$	The start/end time of the delivery service
end_time	Distribution service end time
v	Travel speed, in km/h
t_i^d/t_i^a	Time of departure/arrival of node i
t_i^a	Time of arrival of node i
t_i^s	Service time required for customer i
t_{ij}	Travel time between node i and node j
c_e/c_l	Penalty for arriving at a node before/after the time window
c_l	Penalty for arriving at a node after the time window
c_d/c_{fixed}	Fuel/Fixed cost per kilometer/vehicle
c_{fixed}	Fixed cost per vehicle
x_{ijk}^n	Vehicle k drives from node i to node j in the n-th time slice. If $x_{ijk}^n = 1$, it means that there exists a path from i to node j. Otherwise, $x_{ijk}^n = 0$.
y_{ij}	Depot j provides service to customer i. If $y_{ij} = 1$ means that customer i is served by a vehicle that departs from Depot j. Otherwise, $y_{ij} = 0$.

2.2 Mathematical Model

Eq. (1) represents the final solution to this problem, minimizing the total cost including fixed cost, time window cost, and travel cost. Eq. (2) means that the vehicles used by each depot in all time slices should not exceed the maximum available vehicles for each depot. Eq. (3) implies that the total demand allocated to individual vehicles in all time slices should stay under the vehicle's maximum load constraint. Eq. (4) ensures that each customer is visited exactly once across all time slices and by only one vehicle. Eq. (5) restricts each vehicle to having a single departure and return depot. Eq. (6) indicates that each customer's demand must

be served by a unique depot and cannot be split. Eq. (7) is used to eliminate subtour constraints. Eq. (8) implies that all routes leave the repository and return to the repository no later than the final cut-off time after serving all customer nodes. Eqs. (9) and (10) represent decision variables in the model.

$$\begin{aligned}\min \quad C = {} & c_{fixed} \sum_{n=1}^{n_{ts}} \sum_{i\in D} \sum_{j\in C} \sum_{k\in K} x_{ijk}^{n} \\ & + c_e \sum_{n=1}^{n_{ts}} \sum_{i\in V} \sum_{j\in C} \sum_{k\in K} x_{ijk}^{n} \max\left\{(ET_j - t_j^a), 0\right\} \\ & + c_l \sum_{n=1}^{n_{ts}} \sum_{i\in V} \sum_{j\in C} \sum_{k\in K} x_{ijk}^{n} \max\left\{(t_j^a - LT_j), 0\right\} \\ & + c_d \sum_{n=1}^{n_{ts}} \sum_{i,j\in V} \sum_{k\in K} x_{ijk}^{n} d_{ij}^{s}\end{aligned} \tag{1}$$

s.t.

$$\sum_{n=1}^{n_{ts}} \sum_{j\in C} \sum_{k\in K} x_{ijk}^{n} \le |K|, \quad \forall i \in D \tag{2}$$

$$\sum_{n=1}^{n_{ts}} \sum_{i\in V} \sum_{j\in C} x_{ijk}^{n} q_j \le Q, \quad \forall k \in K \tag{3}$$

$$\sum_{n=1}^{n_{ts}} \sum_{i\in V} \sum_{k\in K} x_{ijk}^{n} = \sum_{n=1}^{n_{ts}} \sum_{i\in V} \sum_{k\in K} x_{jik}^{n} = 1, \quad \forall j \in C \tag{4}$$

$$\sum_{n=1}^{n_{ts}} \sum_{i\in D} \sum_{j\in C} x_{ijk}^{n} = \sum_{n=1}^{n_{ts}} \sum_{i\in C} \sum_{j\in D} x_{jik}^{n} \le 1, \quad \forall k \in K \tag{5}$$

$$\sum_{i\in D} y_{ij} = 1, \quad \forall j \in C \tag{6}$$

$$\sum_{n=1}^{n_{ts}} \sum_{i\in S} \sum_{j\in S} x_{ijk}^{n} \le |S| - 1, \quad \forall k \in K, \quad |S| = \sum_{n=1}^{n_{ts}} \sum_{j\in C} x_{ijk}^{n}, \quad \forall i \in D, \ \forall k \in K \tag{7}$$

$$T_s + \sum_{n=1}^{n_{ts}} \sum_{i\in V} \sum_{j\in V} t_{ij} x_{ijk}^{n} + \sum_{n=1}^{n_{ts}} \sum_{i\in V} \sum_{j\in C} t_i^s x_{ijk}^{n} \le T_f, \quad \forall k \in K \tag{8}$$

$$\sum_{n=1}^{n_{ts}} x_{ijk}^{n} \in \{0,1\}, \quad \forall i \in V, \ \forall j \in V, \ \forall k \in K \quad \text{and} \quad \{i \in D\} \cap \{j \in D\} = \varnothing \tag{9}$$

$$y_{ij} \in \{0,1\}, \quad \forall i \in D, \ \forall j \in C \tag{10}$$

By integrating dynamic requirements, multi depots and more comprehensive cost-benefit analysis, MD-DVRP solves the problems of vehicle scheduling and resource allocation between warehouses in the existing MDVRP and the insufficient applicability of DVRP under the complex logistics challenges. Therefore, It helps enterprises reduce logistics costs and improve economic benefits.

Consider any instance of the (single-depot) dynamic VRP (DVRP). Build an instance of MD-DVRP by setting the number of depots to 1 and copying all other data and dynamic revelation events unchanged. Any algorithm that solves MD-DVRP in polynomial time would therefore solve the original DVRP instance in polynomial time. But DVRP is known to be NP-hard [27], so a polynomial-time solver would contradict these results. Hence MD-DVRP is NP-hard.

3 Proposed Scheduling System TS-DPU

Our system depicted in Fig. 3 is centered on the dynamic processing unit, which has the functions of capturing dynamic customer demands, obtaining current distribution progress, creating static problems and planning optimal routes. Whenever a dynamic requirement is submitted to the depot, the dynamic processing unit will capture the dynamic order and save it to the unprocessed order set. Assign dynamic orders to corresponding depots based on the principle of proximity. The dynamic processing unit provides separate services for each depot. At each time slice end, the dynamic processing unit will obtain the positions of all vehicles and completed orders. Then, the unplanned order set is combined with the current vehicle location to form multiple new static vehicle routing problems based on different depots. Finally, to solve each static problem to obtain the route of vehicles, the results are assigned to the corresponding vehicles.

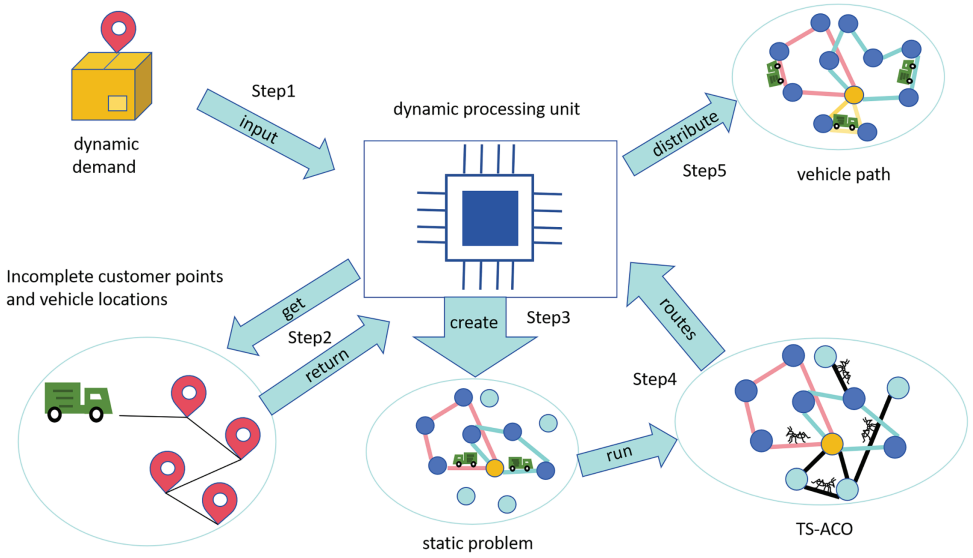


Figure 3: The architecture of TS-DPU

To further clarify the operation of the proposed TS-DPU system, a simple example is provided to illustrate how a dynamic order flows through each stage shown in Fig. 3.

Step 1 (Dynamic demand reception). During each time slice, the dynamic processing unit continuously receives new customer requests. For example, at t = 2.5 h, a new demand (C_new) arises near Depot A and is stored in the unprocessed order set.

Step 2 (Status collection). At the end of the time slice (t = 3 h), the system gathers real-time information on all vehicles and unserved customers. Suppose Vehicle 2 from Depot A has just completed a delivery and is available for new tasks.

Step 3 (Static problem formulation). The dynamic processing unit integrates the new order, current vehicle locations, and remaining customers to construct a new static MDVRP for the next time slice.

Step 4 (Static optimization). The TS-ACO algorithm solves each static subproblem to obtain optimized routes. In this case, C_new is assigned to Vehicle 2 due to its spatial proximity and availability.

Step 5 (Route execution). The optimized routes are transmitted to vehicles for execution, and Vehicle 2 immediately proceeds to serve C_new.

This concise example demonstrates how TS-DPU dynamically transforms real-time information into solvable static problems and maintains system responsiveness to emerging customer demands.

The MILP formulation in the previous chapter addresses the static vehicle routing problem (VRP) with time windows in a multi-depot environment. In this paper, the TS-DPU scheduling system adapts this formulation by converting dynamic problems into a series of static subproblems at each time slice. When new customer demands arrive, they are treated as new static instances, solved using the TS-ACO algorithm. This approach allows us to maintain the MILP structure while efficiently incorporating dynamic demands into the routing plan without disrupting existing routes.

3.1 Temporal-Spatial Distance

In this system, we use a combined distance, which is referred to as the temporal-spatial distance. The details of the temporal-spatial distance are explained below.

As a critical component of the temporal-spatial distance, the temporal distance is calculated using the method proposed by Fan et al. [28]. Suppose that there are two nodes i and j with time windows $[ET_i, LT_i]$ and $[ET_j, LT_j]$, and $ET_i \leq ET_j$. Then at some time $t \in [ET_i, LT_i]$ the time it takes for a vehicle arriving at node i to arrive at node j is $t + t_i^s + d_{ij}^s/v$. Thus, the time t' required to go from node i to node j lies between the interval $[ET_i + t_i^s + d_{ij}^s/v, LT_i + t_i^s + d_{ij}^s/v]$, using $[a, b]$ to represent the above interval. Fig. 4 presents the relationship between the intervals $[a, b]$ and $[ET_j, LT_j]$.

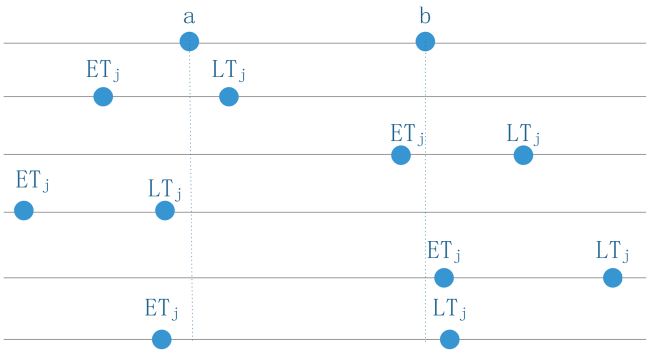


Figure 4: Example chart of interval distance

There are four cases of temporal distance between customer nodes:

- When there is a duplicated part between $[a, b]$ and $[ET_j, LT_j]$, then it means that there is an overlapping part between the moment when the vehicle arrives at node j and the time window of node j. Then the temporal distance between the customer node i and the customer node j is defined as $\mu_1(t_{ij} + t_i^s)$.
- When $a > LT_j$, this indicates that the vehicle's arrival at node j from node i will be delayed. The temporal distance between the customer node i and the customer node j is $\mu_2(a - LT_j)$.
- When $b < ET_j$, this implies that the vehicle will arrive at node j from node i earlier than allowed. In this situation, the temporal distance between the customer node i and the customer node j is $\mu_3(ET_j - b)$.
- When $ET_j < a < b < LT_j$, arrival at node j from node i takes place within the valid time window. Thus, the temporal distance between the customer node i and the customer node j is $(t_{ij} + t_i^s)$.

Here, we define an equation to describe the above four cases in Eq. (11).

$$d_{ij}^{t} = \begin{cases} \mu_1(t_{ij} + t_i^s) & a < ET_j < b \quad \text{or} \quad a < LT_j < b \\ \mu_2(a - LT_j) & LT_j < a \\ \mu_3(ET_j - b) & ET_j > b \\ t_{ij} + t_i^s & ET_j < a < b < LT_j \end{cases} \tag{11}$$

The spatial distance is calculated using the classical Euclidean distance. That is, Eq. (12).

$$d_{ij}^{s} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{12}$$

Due to the difference between temporal and spatial distances in the representative meaning, they are summed up as temporal distances after normalization, respectively. As shown in Eq. (13).

$$\begin{aligned} d_{ij} = \alpha & \frac{d_{ij}^{t} - \min\limits_{m,n \in C, m \neq n} d_{mn}^{t}}{\max\limits_{m,n \in C, m \neq n} d_{mn}^{t} - \min\limits_{m,n \in C, m \neq n} d_{mn}^{t}} \\ + \beta & \frac{d_{ij}^{s} - \min\limits_{m,n \in C, m \neq n} d_{mn}^{s}}{\max\limits_{m,n \in C, m \neq n} d_{mn}^{s} - \min\limits_{m,n \in C, m \neq n} d_{mn}^{s}}, \\ \alpha + \beta = 1, & \quad i, j \in C \end{aligned} \tag{13}$$

To illustrate the computation of the temporal distance in Eq. (11), consider two customers i and j. The spatial distance between customers i and j is 6, the time distance t_{ij} is constant at 1 hour for all cases, and the service time at each node is $t_i^s = 1$ h. The weighting factors for the temporal and spatial distances are $\alpha = 0.1$ and $\beta = 0.9$, respectively. Furthermore, the parameters used for the temporal distance calculation are $\mu_1 = 3$, $\mu_2 = 5$, and $\mu_3 = 1$. For all customer j nodes, the time window is fixed at $[10, 14]$. The calculation results of temporal-spatial distance are shown in Table 2.

Table 2: Temporal-spatial distance calculation for different cases

Case	$[ET_i, ET_i]$	Formula	Result
$\mathbf{ET_j < a < LT_j}$	$[7, 9]$	$a = ET_i + 2 = 9$ $b = LT_i + 2 = 11$ $d_{ij}^{t} = \mu_1(t_{ij} + t_i^s) = 3 \times (1 + 1) = 6$ $d_{ij} = 0.1 \times 6 + 0.9 \times 6 = 6$	$d_{ij} = 6$
$\mathbf{a > LT_j}$	$[13, 15]$	$a = ET_i + 2 = 15$ $b = LT_i + 2 = 17$ $d_{ij}^{t} = \mu_2(a - LT_j) = 5 \times (15 - 14) = 5$ $d_{ij} = 0.1 \times 5 + 0.9 \times 6 = 5.9$	$d_{ij} = 5.9$
$\mathbf{ET_j > b}$	$[5, 7]$	$a = ET_i + 2 = 7$ $b = LT_i + 2 = 9$ $d_{ij}^{t} = \mu_3(ET_j - b) = 1 \times (10 - 9) = 1$ $d_{ij} = 0.1 \times 1 + 0.9 \times 6 = 5.5$	$d_{ij} = 5.5$

(Continued)

Table 2 (continued)

Case	$[ET_i, ET_i]$	Formula	Result
$\mathbf{ET}_j < \mathbf{a} < \mathbf{b} < \mathbf{LT}_j$	$[9, 11]$	$a = ET_i + 2 = 11$ $b = LT_i + 2 = 13$ $d_{ij}^t = t_{ij} + t_i^s = 1 + 1 = 2$ $d_{ij} = 0.1 \times 2 + 0.9 \times 6 = 5.6$	$d_{ij} = 5.6$

This example shows how overlapping and non-overlapping time windows yield different temporal distance values. After normalization, these values are combined with spatial distance in Eq. (13) to obtain the final temporal-spatial distance d_{ij}.

3.2 TS-ACO

In Section 3.1, we explain the temporal-spatial distance, which takes into account the influence of the time window and the spatial distance between nodes. Here, we propose an improved ACO algorithm, TS-ACO. It utilizes the temporal-spatial distance to solve the proposed MD-DVRP.

3.2.1 The Proposed TS-ACO

In the classical ACO algorithm, the choice of the next visited node in the construction of the optimal route is guided by the probability distribution over all feasible nodes. This probability distribution is solved as shown in Eq. (14).

$$p_{ij} = \begin{cases} \dfrac{(\pi_{ij})^{\sigma}(\eta_{ij})^{\rho}}{\sum\limits_{u \in \Omega}(\pi_{iu})^{\sigma}(\eta_{iu})^{\rho}} & j \in \Omega \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

π_{ij} denotes the pheromone generated during the construction of the optimal route by the ants, and η_{ij} denotes the visibility of the ants to all feasible nodes. Typically, π_{ij} and η_{ij} use the inverse of the spatial distance directly, as shown in Eqs. (15)–(17).

$$\pi_{ij} = (1 - \rho)\pi_{ij} + \rho \Delta \pi_{ij} \tag{15}$$

$$\Delta \pi_{ij} = \begin{cases} \dfrac{1}{R_{\text{best}}}, & \text{if } (i, j) \in \text{ best solution} \\ 0, & \text{otherwise} \end{cases} \tag{16}$$

$$\eta_{ij} = \frac{1}{d_{ij}^s} \tag{17}$$

where ρ is the rate of evaporation of pheromones, and R_{best} is the best routes in this iteration.

However, MD-DVRP involves the time window cost of customers, so it is insufficient to consider only the spatial distance to the selection of feasible nodes. Therefore, we use the temporal-spatial distance to represent the relationship between the two nodes, so that the algorithm can comprehensively consider how to reduce operating cost while meeting customer needs on time when planning routes. A detailed pseudocode can be found in Algorithm 1.

Algorithm 1: The improved ACO, TS-ACO

Require: N_{max}: Maximum number of iterations. m: Number of ants. R^{n-1}: Travel route of the previous time slice. V^n: All known customer nodes for the current time slice. π^{n-1}: Pheromone matrix of the previous time slice.

Ensure: R^n: The best route for the current time slice. /*Initialize the pheromone matrix.*/

π = Pheromone initialization(π^{n-1})
/*Get the current position of the vehicle*/
for $i = 1 \rightarrow length(R^{n-1})$ **do**
  $L \leftarrow Get\ Local(R^{n-1})$
end for
/*Building temporal-spatial distance*/
for $i = 1 \rightarrow length(V^n)$ **do**
  for $j = 1 \rightarrow length(V^n)$ **do**
    $d_{ij}^s \leftarrow$ Calculate~Euclidean~distance(i,j)
    $d_{ij}^t \leftarrow$ Calculate~time~window~distance(i,j)
    $d_{ij} \leftarrow$ Calculate~distance(d_{ij}^s, d_{ij}^t)
  end for
end for
/*Constructing the optimal route*/
for $i = 1 \rightarrow N_{max}$ ***do***
  $R \leftarrow$ Constructe~the~solution(π, d, L, V^n)
  $F \leftarrow$ Calculate~the~cost(R_0)
  $R_{best} \leftarrow$ Optimal~route(R,F)
  $\pi \leftarrow$ Pheromone Update(R_{best}, R, F)
end for
return R_{best}

The differences between TSACO and ACO are shown in Table 3.

Table 3: Comparison between Classic ACO and TS-ACO

Feature	Classic ACO	TS-ACO
Computation steps	Path construction based on probabilistic selection using spatial distance and pheromone levels.	Builds upon Classic ACO by integrating temporal-spatial distance, considering both time windows and spatial distances.
Pheromone update	Pheromone update is based on spatial distance.	Pheromone update incorporates temporal-spatial distance, reflecting both time window constraints and spatial distances.
Node selection	Node selection is based on spatial distance and pheromone probability distribution.	Node selection includes both spatial distance and the overlap of time windows to ensure no violation of time constraints.

(Continued)

Table 3 (continued)

Feature	Classic ACO	TS-ACO
Computational complexity	Relatively low, mainly dependent on spatial distance calculations.	Similar to Classic ACO, with minimal additional complexity due to the temporal-spatial distance calculation.
Expected performance	Optimizes vehicle routes in terms of spatial proximity, suitable for static problems without time windows.	Suitable for dynamic vehicle routing problems, effectively optimizing routes under time window constraints and spatial requirements, minimizing penalties for early or late arrivals.
Application scenarios	Primarily used for standard vehicle routing problems without time windows.	Specifically designed for dynamic vehicle routing problems with time windows, demonstrating superior performance in multi-depot and dynamic demand scenarios.
Optimization effect	Minimizes vehicle travel distance.	In addition to minimizing travel distance, TS-ACO reduces penalties for early or late arrivals, improving the timeliness and overall cost efficiency.

3.2.2 The Time Complexity of TS-ACO

A. Single Dynamic Stage.

At each stage $n \in \{1, \ldots, n_{\text{st}}\}$, a new instance of the TSACO solver is executed. The computational work at this stage comprises two main parts:

1. **Initialization:** The solver initializes based on the current state of the system. This involves constructing the spatiotemporal and heuristic information matrices for all V_n known nodes. The complexity of this step is determined by the pairwise calculations between all nodes.

$$O(V_n^2) \tag{18}$$

2. **Iterative Solving:** The core Ant Colony Optimization process runs for NC_{max} iterations to generate routes for the C_n unassigned customers. Within each iteration, the most computationally intensive task is the solution construction for all m ants, which has a quadratic relationship with the number of customers to be routed.

$$O(NC_{max} \cdot m \cdot C_n^2) \tag{19}$$

B. Total Dynamic Stages.

The total computational complexity for the entire dynamic simulation is the sum of the complexities from each individual planning stage.

$$\text{Total Complexity} = \sum_{n=1}^{n_{\text{st}}} O(V_n^2 + NC_{max} \cdot m \cdot C_n^2) \tag{20}$$

We can approximate the overall complexity as:

$$O(n_{\text{st}} \cdot NC_{max} \cdot m \cdot C_{\text{total}}^2) \tag{21}$$

4 Experiment

In this section, we compare the evaluation results of TS-ACO, which considers both temporal and spatial distances, with that of the classical ACO, which only considers spatial distance. The experiments are implemented in Pycharm2025.1 and executed on a Windows 11 platform, equipped with an Intel Core i5-14400 (2.50 GHz) and 32 GB RAM. The algorithm was evaluated using the following parameter settings: $\mu_1 = 3$, $\mu_2 = 5$, $\mu_3 = 1.0$, $v = 80$, $c_e = 50$, $c_l = 100$, $c_d = 20$, and $c_{fixed} = 300$.

4.1 Dataset

As there was no available dataset for the problem addressed in this paper, we modified the instances proposed by Cordeau et al. [29] to obtain the dataset needed for our study. Specifically, assuming that $n_{ts} = 5$, we divided the first half of the instance into initially known customers in Table 4 and then equally divided the second half into four datasets of dynamic customers in Tables 5–8. The time window of each customer was generated randomly from the interval [8,16], with a fixed random seed 42 to ensure the reproducibility of the experiment.

Table 4: Initial known customers (pr1)

Node	x	y	Service	Demand	ET	LT
1	4.352	14.685	0.3300	11	11.5	13.0
2	−29.730	64.136	0.4900	12	10.0	13.0
3	5.243	22.260	0.5000	13	11.5	12.0
4	−40.942	83.209	0.0027	16	12.5	15.5
5	11.877	−24.933	0.0110	22	12.5	16.0
6	1.294	7.349	0.3500	14	12.5	13.0
7	−41.376	50.824	0.1100	25	11.5	13.5
8	−76.672	99.341	0.0640	9	10.0	13.0
9	−18.927	−23.730	0.0220	24	8.0	16.5
10	23.029	11.639	0.1700	18	11.0	14.0
11	−30.664	5.463	0.2300	8	11.0	16.0
12	42.883	−2.966	0.3500	10	9.5	14.5
13	18.597	96.716	0.2600	3	12.0	16.5
14	−42.615	−26.392	0.4000	6	9.0	16.0
15	16.229	9.320	0.3000	22	8.5	13.5
16	−46.545	97.974	0.1300	19	8.5	12.0
17	51.642	5.469	0.4600	16	11.0	12.5
18	−38.562	−3.705	0.3800	13	11.0	12.0
19	−35.297	−24.896	0.3100	19	10.0	11.0
20	−22.833	−9.814	0.0650	13	12.0	14.5
21	−26.404	29.529	0.2600	10	8.0	16.0
22	−49.329	33.374	0.4100	6	11.5	13.0
23	−16.779	19.537	0.0130	10	8.0	11.5
24	12.268	−55.811	0.0340	19	8.5	10.5

Table 5: Dynamic customers in the first time slice (pr1)

Node	x	y	Service	Demand	ET	LT
1	12.268	−55.811	0.0340	19	8.5	10.5
2	48.907	6.274	0.4500	5	10.0	15.5
3	−91.943	27.588	0.1700	5	8.0	14.0
4	−37.933	−21.613	0.0250	21	8.5	11.5
5	−65.002	77.234	0.3600	20	10.5	15.0
6	−65.118	30.212	0.4800	17	8.0	14.0

Table 6: Dynamic customers in the second time slice (pr1)

Node	x	y	Service	Demand	ET	LT
1	−22.754	55.408	0.2200	9	10.5	11.0
2	−54.755	14.368	0.0035	14	12.0	14.5
3	−71.100	−18.616	0.0100	15	12.5	14.5
4	25.482	6.287	0.1600	7	10.5	15.5
5	−11.560	11.615	0.0840	16	8.5	16.5
6	−67.413	68.323	0.1000	12	11.0	13.5

Table 7: Dynamic customers in the third time slice (pr1)

Node	x	y	Service	Demand	ET	LT
1	−11.920	11.755	0.1900	3	11.0	16.5
2	−56.622	73.340	0.2400	20	11.0	15.5
3	−52.039	6.567	0.3700	4	9.5	14.0
4	57.404	23.822	0.3000	16	12.0	15.5
5	23.767	29.083	0.3200	21	12.5	15.5
6	−20.673	57.892	0.4300	9	10.0	12.5

Table 8: Dynamic customers in the fourth time slice (pr1)

Node	x	y	Service	Demand	ET	LT
1	−37.756	−33.325	0.3800	25	11.5	16.0
2	−43.030	20.453	0.2600	14	8.0	9.0
3	29.840	11.633	0.4800	25	10.5	13.0
4	−50.665	−23.126	0.4700	15	10.0	11.0
5	−7.849	32.074	0.0029	8	8.5	14.0
6	−4.175	−1.569	0.2300	13	11.5	14.0

4.2 Parameter Sensitivity Analysis

For an ACO algorithm, the colony size and the iteration count are two very important parameters. Here, we make a parameter sensitivity analysis to find the best combination of them shown in Fig. 5. The parameter ranges used in this analysis were selected with reference to previous studies on ACO parameter

tuning [30], ensuring that the experimental settings are consistent with established practices in the literature. According to Fig. 5, it is obvious that when the iteration number and the number of ant colonies are 150 and 60, respectively, the solution cost is the lowest.

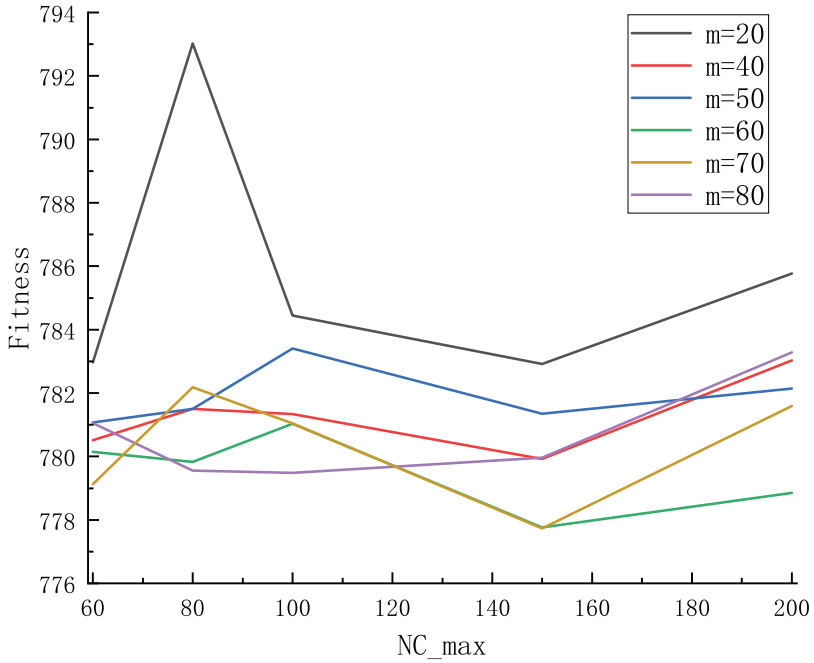
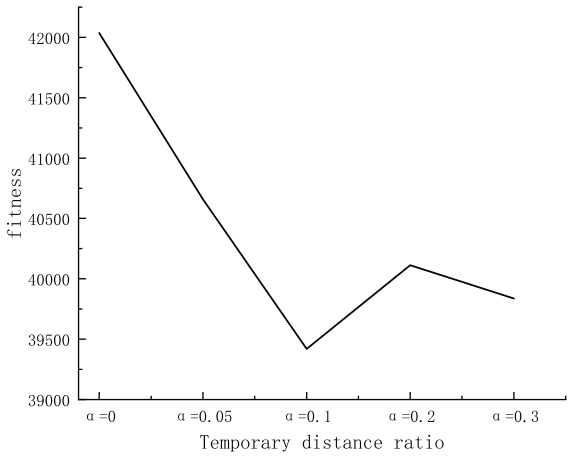


Figure 5: Fitness under different iteration times and ant colony number combinations

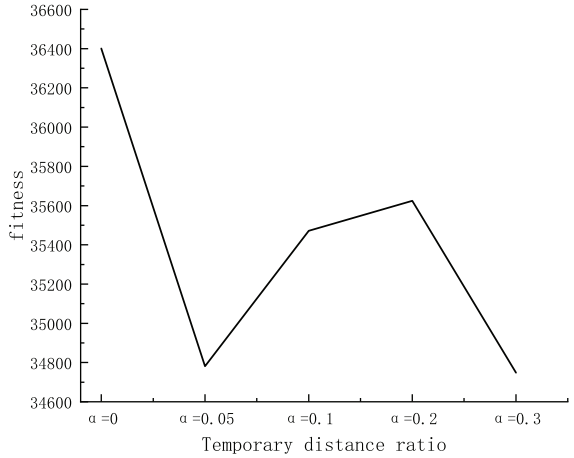
Under NC_{max} = 150, m = 60, pheromone evaporation rate is equal to 0.7, heuristic factor is equal to 2.2, pheromone factor is equal to 1.4. We then to find the best α. The results are shown in Fig. 6. Note that $\alpha = 0$ means that spatial distance is only considered. As can be seen in Fig. 6a, in terms of average fitness, when α is equal to 0.1 and β is equal to 0.9, the solution effect is optimal. Although the results obtained with different values of α are very similar and in terms of the minimum fitness value, the result obtained by $\alpha = 0.3$ is better than $\alpha = 0.1$. It is easy to find that considering the temporal distance is significantly better than considering only the spatial distance. Moreover, The optimal values of α and β reflects the fact that in many dynamic logistics scenarios, especially those with multiple depots and time windows, spatial distances typically play a more dominant role in determining optimal routes. The time component, while important, is generally less critical when the logistics network is relatively stable, and the customer demand is spread across a broader area, making spatial factors more influential in the overall solution quality.

Similarly, under the above parameter settings, we also conducted a sensitivity analysis on the μ_1, μ_2 and μ_3.

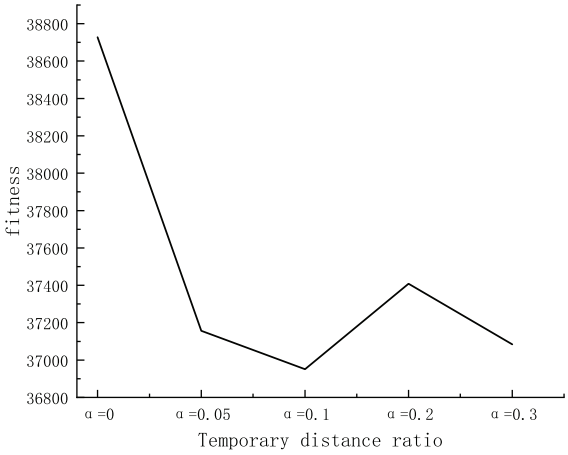
As shown in Table 9, the combination ($\mu_1 = 3$, $\mu_2 = 5$, $\mu_3 = 1$) achieves the lowest average cost (32,856.65) among all tested configurations. It also yields the smallest minimum cost (29,970.17), indicating that this parameter setting provides the most cost-efficient performance in the spatio–temporal distance calculation. Therefore, $(3, 5, 1)$ can be regarded as the optimal configuration for minimizing the overall routing cost.



(**a**) Maximum fitness of 150 iterations under different α



(**b**) Minimum fitness of 150 iterations under different α



(**c**) Average fitness of 150 iterations under different α

Figure 6: Fitness of 150 iterations under different α

Table 9: Cost Analysis for Different μ_1, μ_2, μ_3 Combinations

Combination	Minimum cost	Average cost	Maximum cost
($\mu_1 = 2$, $\mu_2 = 3$, $\mu_3 = 1$)	31,731.9960	34,167.3788	38,701.9194
($\mu_1 = 2$, $\mu_2 = 3$, $\mu_3 = 2$)	30,937.4593	33,417.9629	36,243.5240
($\mu_1 = 2$, $\mu_2 = 5$, $\mu_3 = 1$)	31,792.8099	33,668.7954	35,956.8726
($\mu_1 = 2$, $\mu_2 = 5$, $\mu_3 = 2$)	30,422.3212	33,787.5412	36,343.0589
($\mu_1 = 3$, $\mu_2 = 3$, $\mu_3 = 1$)	31,243.0371	33,717.0119	35,934.4687
($\mu_1 = 3$, $\mu_2 = 3$, $\mu_3 = 2$)	30,507.4774	33,129.4381	35,591.9067
($\mu_1 = 3$, $\mu_2 = 5$, $\mu_3 = 1$)	29,970.1679	32,856.6465	36,056.2564
($\mu_1 = 3$, $\mu_2 = 5$, $\mu_3 = 2$)	30,835.9040	33,394.3635	36,890.4576

4.3 Further Experiments

To further evaluate the performance of TS-ACO, additional 18 instances (pr2–pr19) from Cordeau et al. [29] were adapted, and comparative experiments were carried out. As illustrated in Fig. 7, TS-ACO demonstrates notable improvements over classical ACO in instances pr1–pr4, pr7–pr15, while performing close in the remaining instances. Specifically, instances pr5, pr6, pr16 and pr19 involve a relatively large number of customers. In contrast, although pr17 and pr18 contain fewer customers, they include more depots. The results demonstrate that TS-ACO performs better in reducing operating costs in scenarios with moderate data set sizes and depot quantities.

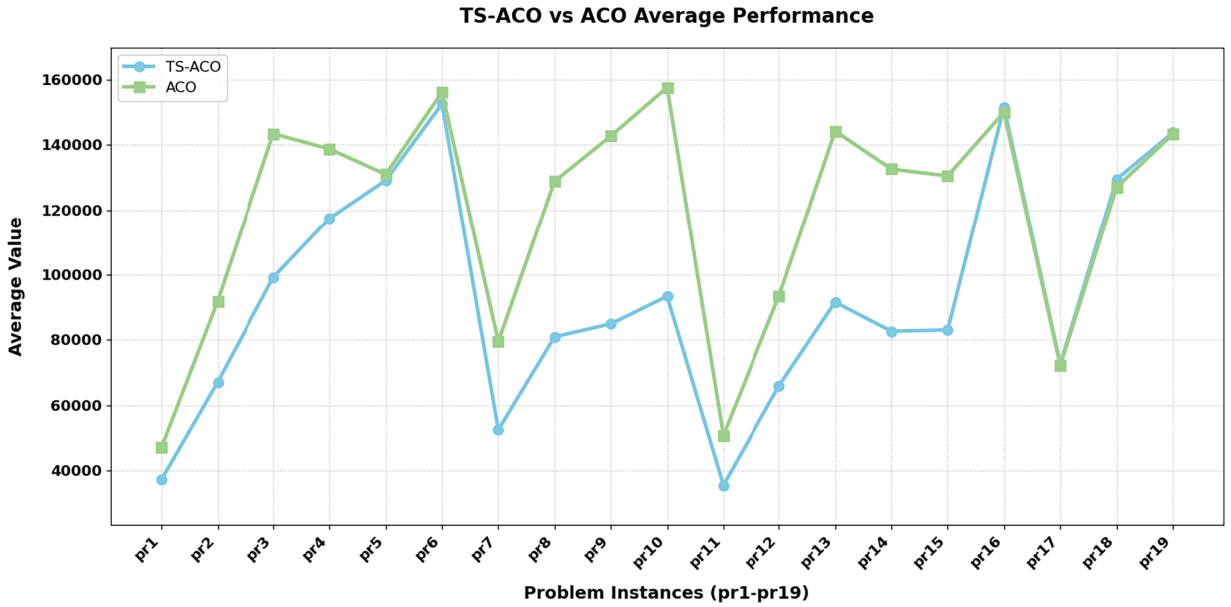


Figure 7: The average operating cost of ACO and TS-ACO

In addition, a Wilcoxon signed-rank test was performed between TS-ACO and classical ACO to statistically assess the significance of the observed performance differences. As shown in Table 10, TS-ACO achieves statistically significant improvements over classical ACO in pr11, pr13, and pr15, with p-values of 0.0215, 0.0049, and 0.0094, respectively—all below the 0.05 significance level. In pr16, the p-value reaches 0.0897, which is close to the threshold, indicating a marginal yet consistent advantage of TS-ACO. These results suggest that TS-ACO delivers significantly better performance in several instances and maintains generally superior or comparable outcomes across the remaining datasets. Overall, TS-ACO achieves a favorable balance between solution quality and computational efficiency, demonstrating both effectiveness and practicality for dynamic routing scenarios.

Moreover, a comprehensive comparison was conducted among classical ACO, PSO [31], VNS [32], TSACS [30], and TS-ACO.

Table 10: *p*-value analysis for different instances

Instances	*p*-value	Instances	*p*-value
pr1	0.3118	pr11	0.0215
pr2	0.2774	pr12	0.8408
pr3	0.6215	pr13	0.0049
pr4	0.2943	pr14	0.5459
pr5	0.5459	pr15	0.0094
pr6	0.2943	pr16	0.0897
pr7	0.7841	pr17	0.5217
pr8	0.8695	pr18	0.6477
pr9	0.8408	pr19	0.8983
pr10	0.9854		

As shown in Tables 11 and 12, although PSO and VNS achieve slightly better solution quality than TS-ACO, their computational times are significantly longer. As the number of customers increases, their runtime grows rapidly—reaching several times that of TS-ACO. PSO is somewhat faster than VNS but still far from meeting real-time requirements. In contrast, TSACS exhibits high computational efficiency but produces inferior solutions compared with TS-ACO.

Table 11: Performance comparison of algorithms on different datasets

Dataset	Algorithm	Min cost	Avg cost	Max cost	Avg time (s)
pr1	ACO	2.9900E + 04	3.4043E + 04	3.5969E + 04	1.0935E + 02
	PSO	2.4554E + 04	2.6307E + 04	2.8366E + 04	3.8479E + 02
	VNS	2.4803E + 04	2.5358E + 04	2.6162E + 04	1.0582E + 03
	TSACS	6.5296E + 04	6.7765E + 04	7.1901E + 04	4.9536E + 00
	TSACO	3.1288E + 04	3.4051E + 04	3.5707E + 04	1.1707E + 02
pr2	ACO	5.9245E + 04	6.3456E + 04	6.8077E + 04	4.3362E + 02
	PSO	4.4041E + 04	4.6367E + 04	4.9371E + 04	1.4250E + 03
	VNS	3.8343E + 04	3.9055E + 04	3.9842E + 04	8.9880E + 03
	TSACS	1.3011E + 05	1.3681E + 05	1.4388E + 05	1.0246E + 01
	TSACO	6.0374E + 04	6.3449E + 04	6.9126E + 04	4.6374E + 02
pr3	ACO	9.4648E + 04	1.0104E + 05	1.0475E + 05	9.5437E + 02
	PSO	6.6094E + 04	6.9274E + 04	7.4845E + 04	3.2527E + 03
	VNS	5.5907E + 04	5.6699E + 04	5.7479E + 04	1.0005E + 04
	TSACS	2.5131E + 05	2.6567E + 05	2.7843E + 05	1.6769E + 01
	TSACO	9.6807E + 04	1.0155E + 05	1.0754E + 05	1.0298E + 03
pr4	ACO	1.2256E + 05	1.2925E + 05	1.3268E + 05	1.6769E + 03
	PSO	8.3578E + 04	8.9438E + 04	9.7026E + 04	4.1882E + 03
	VNS	6.6571E + 04	6.7527E + 04	6.8118E + 04	1.9422E + 04
	TSACS	3.0175E + 05	3.1556E + 05	3.3573E + 05	2.5315E + 01
	TSACO	1.2183E + 05	1.2888E + 05	1.3332E + 05	1.7866E + 03

(Continued)

Table 11 (continued)

Dataset	Algorithm	Min cost	Avg cost	Max cost	Avg time (s)
pr5	ACO	1.4815E + 05	1.5307E + 05	1.5995E + 05	2.4924E + 03
	PSO	1.0384E + 05	1.0925E + 05	1.1729E + 05	4.5921E + 03
	VNS	7.6735E + 04	7.7891E + 04	7.9703E + 04	2.5864E + 04
	TSACS	3.3494E + 05	3.5052E + 05	3.6164E + 05	3.3761E + 01
	TSACO	1.4748E + 05	1.5282E + 05	1.5783E + 05	2.6291E + 03
pr6	ACO	1.7994E + 05	1.8885E + 05	1.9578E + 05	3.6997E + 03
	PSO	1.2068E + 05	1.2535E + 05	1.3431E + 05	8.2708E + 03
	VNS	8.8160E + 04	8.9500E + 04	9.0476E + 05	4.7719E + 04
	TSACS	4.8006E + 05	4.8919E + 05	5.0252E + 05	4.4759E + 01
	TSACO	1.8494E + 05	1.9019E + 05	1.9498E + 05	3.8879E + 03
pr7	ACO	4.5826E + 04	4.9311E + 04	5.1579E + 04	2.5393E + 02
	PSO	3.3749E + 04	3.4600E + 04	3.8289E + 04	9.1160E + 02
	VNS	3.2308E + 04	3.3264E + 04	3.4496E + 04	1.7203E + 03
	TSACS	1.0863E + 05	1.1447E + 05	1.2276E + 05	7.5805E + 00
	TSACO	4.7079E + 04	4.9251E + 04	5.3034E + 04	2.7551E + 02
pr8	ACO	9.6427E + 04	9.8720E + 04	1.0094E + 05	9.0008E + 02
	PSO	6.2320E + 04	6.8102E + 04	7.2306E + 04	2.5500E + 03
	VNS	5.2837E + 04	5.4592E + 04	5.6432E + 04	8.3073E + 03
	TSACS	2.3447E + 05	2.4197E + 05	2.5040E + 05	1.7394E + 01
	TSACO	9.4411E + 04	9.9669E + 04	1.0544E + 05	9.6215E + 02
pr9	ACO	1.3940E + 05	1.4523E + 05	1.5341E + 05	2.0364E + 03
	PSO	8.9303E + 04	9.6196E + 04	1.0459E + 05	5.2632E + 03
	VNS	7.0152E + 04	7.0872E + 04	7.1387E + 04	2.5677E + 04
	TSACS	3.6203E + 05	3.7674E + 05	3.8651E + 05	2.9803E + 01
	TSACO	1.3823E + 05	1.4533E + 05	1.5235E + 05	2.1578E + 03
pr10	ACO	1.9066E + 05	1.9886E + 05	2.0951E + 05	3.5412E + 03
	PSO	1.2868E + 05	1.3836E + 05	1.4678E + 05	6.7795E + 03
	VNS	9.1630E + 04	9.4876E + 04	9.7056E + 04	3.8991E + 04
	NPC	4.8122E + 05	4.9644E + 05	5.0840E + 05	5.2066E + 01
	TSACO	1.9298E + 05	1.9948E + 05	2.0514E + 05	3.7861E + 03

Table 12: Performance comparison of algorithms on different datasets continued

Dataset	Algorithm	Min cost	Avg cost	Max cost	Avg time (s)
pr11	ACO	3.3888E + 04	3.7113E + 04	4.0191E + 04	1.1951E + 02
	PSO	2.7804E + 04	2.8971E + 04	3.0140E + 04	3.6441E + 02
	VNS	2.5882E + 04	2.6587E + 04	2.7530E + 04	6.9670E + 02
	TSACS	6.6133E + 04	7.0827E + 04	7.4593E + 04	4.2702E + 00
	TSACO	3.3880E + 04	3.7185E + 04	4.1178E + 04	1.2843E + 02

(Continued)

Table 12 (continued)

Dataset	Algorithm	Min cost	Avg cost	Max cost	Avg time (s)
pr12	ACO	5.9275E + 04	6.1757E + 04	6.5060E + 04	4.4286E + 02
	PSO	4.3179E + 04	4.6325E + 04	4.9586E + 04	1.4327E + 03
	VNS	3.9027E + 04	4.0113E + 04	4.1538E + 04	6.3364E + 03
	TSACS	1.2905E + 05	1.3438E + 05	1.5276E + 05	9.6289E + 00
	TSACO	5.4165E + 04	6.1091E + 04	6.7038E + 04	4.7250E + 02
pr13	ACO	8.8683E + 04	9.4127E + 04	1.0048E + 05	9.7407E + 02
	PSO	6.6204E + 04	6.9734E + 04	7.5088E + 04	3.2024E + 03
	VNS	5.5421E + 04	5.7220E + 04	5.8096E + 04	2.4042E + 04
	TSACS	2.6397E + 05	2.7865E + 05	3.0340E + 05	1.6167E + 01
	TSACO	9.1256E + 04	9.6009E + 04	1.0263E + 05	1.0403E + 03
pr14	ACO	1.1429E + 05	1.2184E + 05	1.2789E + 05	1.6934E + 03
	PSO	8.4251E + 04	9.0605E + 04	9.9722E + 04	4.0020E + 03
	VNS	6.6319E + 04	6.7494E + 04	6.8196E + 04	4.9015E + 04
	TSACS	3.1490E + 05	3.2717E + 05	3.4245E + 05	2.4704E + 01
	TSACO	1.1813E + 05	1.2392E + 05	1.3081E + 05	1.7750E + 03
pr15	ACO	1.3689E + 05	1.4604E + 05	1.5192E + 05	2.4518E + 03
	PSO	1.0370E + 05	1.1142E + 05	1.2159E + 05	4.5502E + 03
	VNS	7.5835E + 04	7.7798E + 04	7.9154E + 04	2.4615E + 04
	TSACS	3.5562E + 05	3.6444E + 05	3.7292E + 05	3.2003E + 01
	TSACO	1.3852E + 05	1.4503E + 05	1.5215E + 05	2.6628E + 03
pr16	ACO	1.7566E + 05	1.8610E + 05	1.9328E + 05	3.6624E + 03
	PSO	1.2107E + 05	1.2838E + 05	1.3496E + 05	8.9803E + 03
	VNS	8.9879E + 04	9.0789E + 04	9.3100E + 04	4.3167E + 04
	TSACS	4.8858E + 05	5.0177E + 05	5.1045E + 05	4.2279E + 01
	TSACO	1.7484E + 05	1.8456E + 05	1.8975E + 05	3.8915E + 03
pr17	ACO	4.4252E + 04	4.8781E + 04	5.3627E + 04	2.5679E + 02
	PSO	3.3667E + 04	3.4899E + 04	3.5941E + 04	9.7131E + 02
	VNS	3.3003E + 04	3.3779E + 04	3.4590E + 04	2.6868E + 03
	TSACS	1.1258E + 05	1.2219E + 05	1.2989E + 05	6.4754E + 00
	TSACO	4.3758E + 04	4.8950E + 04	5.3081E + 04	2.7418E + 02
pr18	ACO	8.6587E + 04	9.2137E + 04	9.6238E + 04	8.9736E + 02
	PSO	6.6347E + 04	7.1548E + 04	7.9300E + 04	2.7439E + 03
	VNS	5.2822E + 04	5.4908E + 04	5.6815E + 04	1.8595E + 04
	TSACS	2.3521E + 05	2.5349E + 05	2.6965E + 05	1.6104E + 01
	TSACO	8.2562E + 04	8.9091E + 04	9.4115E + 04	9.7834E + 02

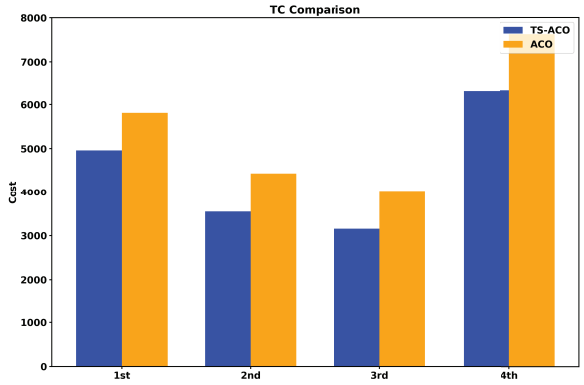
(Continued)

Table 12 (continued)

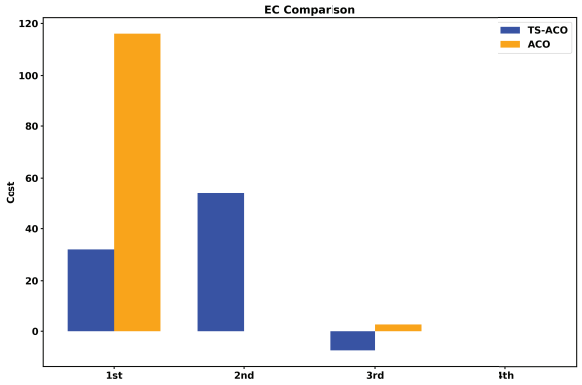
Dataset	Algorithm	Min cost	Avg cost	Max cost	Avg time (s)
pr19	ACO	1.3243E + 05	1.3779E + 05	1.4364E + 05	2.0180E + 03
	PSO	9.3189E + 04	9.7585E + 04	1.0382E + 05	5.8974E + 03
	VNS	6.9065E + 04	7.1682E + 04	7.3680E + 04	7.5497E + 04
	TSACS	3.6997E + 05	3.9261E + 05	4.0901E + 05	2.8987E + 01
	TSACO	1.3202E + 05	1.3839E + 05	1.4404E + 05	2.1529E + 03

In this section, we compare the evaluation, TS-ACO, which incorporates temporal-spatial distances, outperforms the classical ACO algorithm in 13 instances. To further verify the effectiveness of TS-ACO at different time slices, we compare the different cost increments between consecutive time slices for both algorithms. As illustrated in Fig. 8, where TC, EC, LC, and DC denote the total cost, the early arrival cost, the late arrival cost and the distance cost, respectively. Compared to the classical ACO algorithm, TS-ACO reduces all the cost increment in all time slices except the early arrival cost. Although the 2nd time slice has a higher early arrival cost increment than the classical ACO in the early arrival cost, TS-ACO reduces the cost increment by a large amount at the other time slices, especially at the 1st time slice. This comparative analysis confirms that the ACO algorithm with temporal-spatial distance achieves superior performance in cost reduction across all time slices.

The results demonstrate that TS-ACO achieves a significant cost reduction within each time slice, attesting to its superior performance in contemporary logistics. When new customer orders emerge in real time, it seamlessly inserts these dynamic requests into the existing delivery routes at a lower marginal cost, thereby offering a measurable cost-saving advantage for the enterprise.

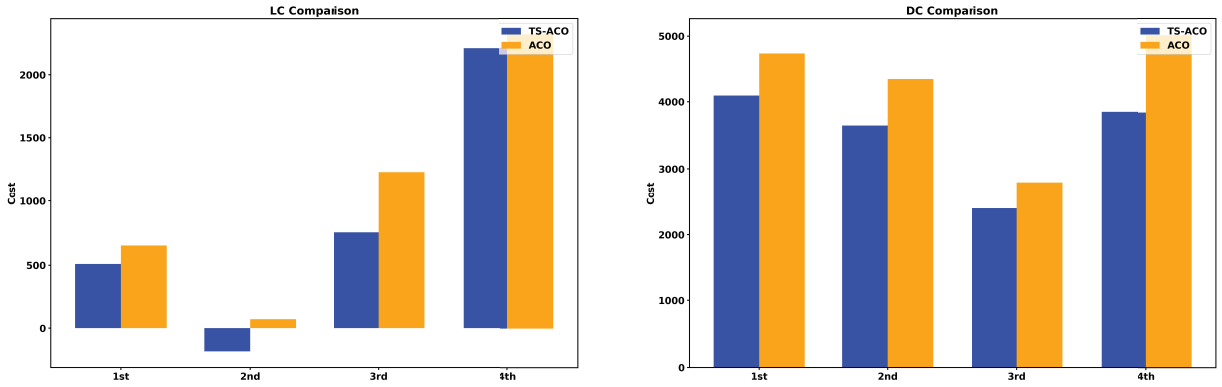


(**a**) Comparison of TC increment between each consecutive time slice(pr1)



(**b**) Comparison of EC increment between each consecutive time slice(pr1)

Figure 8: (Continued)

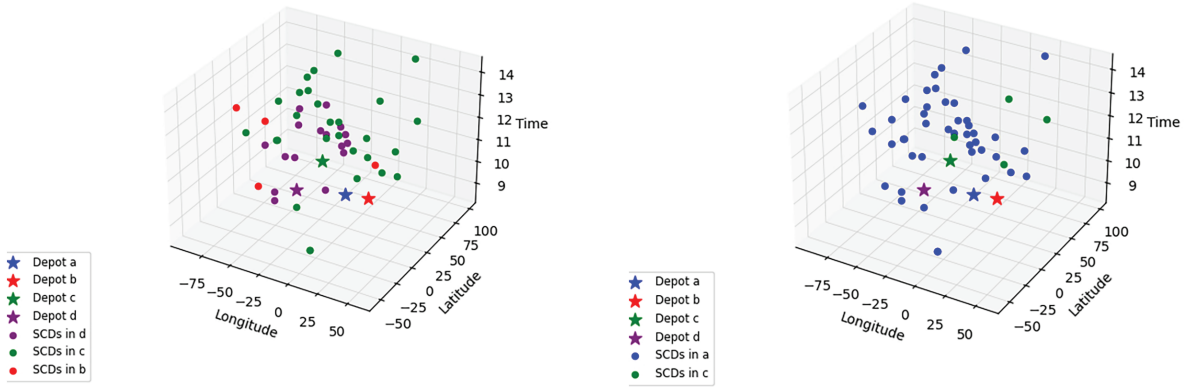


(**c**) Comparison of LC increment between each consecutive time slice(pr1)

(**d**) Comparison of DC increment between each consecutive time slice(pr1)

Figure 8: Comparison of TS-ACO and ACO at different costs

Fig. 9 illustrates the distribution of depots and customers, where star-shaped icons represent depots and circular dots denote customers. Customers who share the same color as a depot are served by vehicles departing from that depot. The figures show the depot–customer assignments under the optimal routes generated by the classical ACO and TS-ACO, respectively, using the pr1 instance. In particular, TS-ACO requires fewer deployed depots to complete the routing task, thereby reducing overall operational costs.



(**a**) Distribution of customers of each depot after ACO optimization (pr1)

(**b**) Distribution of customers of each depot after TS-ACO optimization (pr1)

Figure 9: Distribution of customers of each depot after optimization

5 Conclusion

In today's rapidly developing era, the logistics industry needs real time service, so many logistics companies choose to expand a single depot into multiple depots to meet customer demand in time. Due to the rapid development of communication technology, customers can submit their requirements to service providers at any time. Consequently, a scheduling system TS-DPU driven by the TS-ACO is developed in this paper to tackle the proposed MD-DVRP. It expands the traditional DVRP and brings it closer to the real environment. Here, we add the time cost to the calculation of the total cost and propose TS-ACO

that considers the spatial and temporal distance to solve the problem. Experiments have shown that TS-ACO considering spatial and temporal distance is significantly better than the classical ACO only considers spatial distance.

However, our study did not consider multi-objective optimization, thereby overlooking the simultaneous minimization of operational costs and maximization of customer satisfaction, both of which are critical in real-world logistics operations. In addition, environmental dimensions such as carbon emissions and energy consumption, which are increasingly emphasized in contemporary logistics practice, have not been explicitly integrated into the proposed framework. Future work should therefore extend the current single-objective model to a multi-objective model that jointly optimizes economic efficiency, service quality, and environmental sustainability.

Acknowledgement: None.

Funding Statement: This work was supported by the Startup Foundation for Introducing Talent of Nanjing University of Information Science and Technology.

Author Contributions: The authors confirm contribution to the paper as follows: Conceptualization, Tsu-Yang Wu and Chien-Ming Chen; methodology, Tsu-Yang Wu and Yanan Zhao; validation, Chengyuan Yu; formal analysis, Yanan Zhao and Chengyuan Yu; investigation, Saru Kumari and Chien-Ming Chen; data curation, Saru Kumari and Chien-Ming Chen; writing—original draft preparation, Tsu-Yang Wu, Yanan Zhao, Chengyuan Yu, Saru Kumari and Chien-Ming Chen. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The data are contained within the article.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Kalatzantonakis P, Sifaleras A, Samaras N. A reinforcement learning-variable neighborhood search method for the capacitated vehicle routing problem. Expert Syst Appl. 2023;213:118812. doi:10.1016/j.eswa.2022.118812.
2. Souza IP, Boeres MCS, Moraes REN. A robust algorithm based on differential evolution with local search for the capacitated vehicle routing problem. Swarm Evol Comput. 2023;77:101245. doi:10.1016/j.swevo.2023.101245.
3. Wu Q, Xia X, Song H, Zeng H, Xu X, Zhang Y, et al. A neighborhood comprehensive learning particle swarm optimization for the vehicle routing problem with time windows. Swarm Evol Comput. 2024;84:101425. doi:10.1016/j.swevo.2023.101425.
4. Wu H, Gao Y, Wang W, Zhang Z. A hybrid ant colony algorithm based on multiple strategies for the vehicle routing problem with time windows. Complex Intell Syst. 2023;9(3):2491–508. doi:10.1109/icie.2009.237.
5. Zhang K, Lin X, Li M. Graph attention reinforcement learning with flexible matching policies for multi-depot vehicle routing problems. Phys A Stat Mech Appl. 2023;611:128451. doi:10.1016/j.physa.2023.128451.
6. Montemanni R, Gambardella LM, Rizzoli AE, Donati AV. Ant colony system for a dynamic vehicle routing problem. J Comb Optim. 2005;10(4):327–43. doi:10.1007/s10878-005-4922-6.
7. Wilson NHM, Colvin NJ. Computer control of the Rochester dial-a-ride system. Cambridge, MA, USA: Massachusetts Institute of Technology, Center for Transportation Studies; 1977.
8. Psaraftis HN. A dynamic programming solution to the single vehicle many-to-many immediate request dial-a-ride problem. Transp Sci. 1980;14(2):130–54. doi:10.1287/trsc.14.2.130.
9. Powell W, Jaillet P, Odoni A. Stochastic and dynamic networks and routing. Handb Oper Res Manag Sci. 1995;8(C):141–295.
10. Zhang H, Ge H, Yang J, Tong Y. Review of vehicle routing problems: models, classification and solving algorithms. Arch Comput Methods Eng. 2022;29(1):195–221. doi:10.1007/s11831-021-09574-x.

11. Savelsbergh M, Sol M. Drive: dynamic routing of independent vehicles. Oper Res. 1998;46(4):474–90. doi:10.1287/opre.46.4.474.
12. Gendreau M, Guertin F, Potvin JY, Taillard E. Parallel tabu search for real-time vehicle routing and dispatching. Transp Sci. 1999;33(4):381–90. doi:10.1287/trsc.33.4.381.
13. Kilby P, Prosser P, Shaw P. Dynamic VRPs: a study of scenarios. University of Strathclyde Technical Report [Internet]. 2002 [cited 2025 Nov 20]. Available from: https://www.researchgate.net/publication/2944001.
14. Montemanni R, Gambardella LM, Rizzoli AE, Donati AV. A new algorithm for a dynamic vehicle routing problem based on ant colony system. Handbook Syst Autoimmune Dis. 2003;1(1):27–30. doi:10.1007/s10878-005-4922-6.
15. Potvin JY, Xu Y, Benyahia I. Vehicle routing and scheduling with dynamic travel times. Comput Oper Res. 2006;33(4):1129–37.
16. Azi N, Gendreau M, Potvin JY. A dynamic vehicle routing problem with multiple delivery routes. Ann Oper Res. 2012;199(1):103–12. doi:10.1007/s10479-011-0991-3.
17. de Armas J, Melián-Batista B. Variable neighborhood search for a dynamic rich vehicle routing problem with time windows. Comput Ind Eng. 2015;85(1):120–31. doi:10.1016/j.cie.2015.03.006.
18. Jia YH, Chen WN, Gu T, Zhang H, Yuan H, Lin Y, et al. A dynamic logistic dispatching system with set-based particle swarm optimization. IEEE Trans Syst Man Cybern Syst. 2017;48(9):1607–21. doi:10.1109/tsmc.2017.2682264.
19. Xiang X, Tian Y, Zhang X, Xiao J, Jin Y. A pairwise proximity learning-based ant colony algorithm for dynamic vehicle routing problems. IEEE Trans Intell Transp Syst. 2021;23(6):5275–86. doi:10.1109/tits.2021.3052834.
20. Pan W, Liu SQ. Deep reinforcement learning for the dynamic and uncertain vehicle routing problem. Appl Intell. 2023;53(1):405–22. doi:10.1007/s10489-022-03456-w.
21. Sze JF, Salhi S, Wassan N. An adaptive variable neighbourhood search approach for the dynamic vehicle routing problem. Comput Oper Res. 2024;164:106531. doi:10.1016/j.cor.2024.106531.
22. Demirbilek M. Optional and mandatory assignment strategies for dynamic vehicle routing with time windows. Ain Shams Eng J. 2025;16(9):103462. doi:10.1016/j.asej.2025.103462.
23. Hanshar F, Ombuki-Berman BM. Dynamic vehicle routing using genetic algorithms. Appl Intell. 2007;27(1):89–99. doi:10.1007/s10489-006-0033-z.
24. Ouaddi K, Benadada Y, Mhada FZ. Ant colony system for dynamic vehicle routing problem with overtime. Int J Adv Comput Sci Appl. 2018;9(6):306–15. doi:10.14569/ijacsa.2018.090644.
25. Mańdziuk J, żychowski A. A memetic approach to vehicle routing problem with dynamic requests. Appl Soft Comput. 2016;48:522–34. doi:10.1016/j.asoc.2016.06.032.
26. Mogale DG, Ghadge A, Jena SK. Modelling and optimising a multi-depot vehicle routing problem for freight distribution in a retail logistics network. Comput Ind Eng. 2025;207:111315. doi:10.1016/j.cie.2025.111315.
27. Psaraftis HN, Wen M, Kontovas CA. Dynamic vehicle routing problems: three decades and counting. Networks. 2016;67(1):3–31. doi:10.1002/net.21628.
28. Fan H, Zhang Y, Tian P, Lv Y, Fan H. Time-dependent multi-depot green vehicle routing problem with time windows considering temporal-spatial distance. Comput Oper Res. 2021;129:105211. doi:10.1016/j.cor.2021.105211.
29. Cordeau JF, Laporte G, Mercier A. A unified tabu search heuristic for vehicle routing problems with time windows. J Oper Res Soc. 2001;52(8):928–36. doi:10.1057/palgrave.jors.2601163.
30. Zhang W, Gajpal Y, Appadoo SS, Wei Q. Multi-depot green vehicle routing problem to minimize carbon emissions. Sustainability. 2020;12(8):3500. doi:10.3390/su12083500.
31. Ai TJ, Kachitvichyanukul V. A particle swarm optimization for the vehicle routing problem with simultaneous pickup and delivery. Comput Oper Res. 2009;36(5):1693–702. doi:10.1016/j.cor.2008.04.003.
32. Kuo Y, Wang CC. A variable neighborhood search for the multi-depot vehicle routing problem with loading cost. Expert Syst Appl. 2012;39(8):6949–54. doi:10.1016/j.eswa.2012.01.024.