Multiobjective Vehicle Routing Optimization with Time Windows: A Hybrid Approach Using Deep Reinforcement Learning and NSGA-II

Rixin Wu, Ran Wang, *Member, IEEE*, Jie Hao, *Member, IEEE*, Qiang Wu, *Member, IEEE*, Ping Wang, *Fellow, IEEE*, and Dusit Niyato, *Fellow, IEEE*

Abstract—This paper proposes a weight-aware deep reinforcement learning (WADRL) approach designed to address the multiobjective vehicle routing problem with time windows (MOVRPTW), aiming to use a single deep reinforcement learning (DRL) model to solve the entire multiobjective optimization problem. The Non-dominated sorting genetic algorithm-II (NSGA-II) method is then employed to optimize the outcomes produced by the WADRL, thereby mitigating the limitations of both approaches. Firstly, we design an MOVRPTW model to balance the minimization of travel cost and the maximization of customer satisfaction. Subsequently, we present a novel DRL framework that incorporates a transformer-based policy network. This network is composed of an encoder module, a weight embedding module where the weights of the objective functions are incorporated, and a decoder module. NSGA-II is then utilized to optimize the solutions generated by WADRL. Finally, extensive experimental results demonstrate that our method outperforms the existing and traditional methods. Due to the numerous constraints in VRPTW, generating initial solutions of the NSGA-II algorithm can be time-consuming. However, using solutions generated by the WADRL as initial solutions for NSGA-II significantly reduces the time required for generating initial solutions. Meanwhile, the NSGA-II algorithm can enhance the quality of solutions generated by WADRL, resulting in solutions with better scalability. Notably, the weight-aware strategy significantly reduces the training time of DRL while achieving better results, enabling a single DRL model to solve the entire multiobjective optimization problem.

Index Terms—Multiobjective optimization, Vehicle routing problem, Deep reinforcement learning, Transformer, Weight-aware strategy.

I. INTRODUCTION

VEHICLE routing problem (VRP), a pivotal sector in transportation logistics, plays a crucial role in modern traffic management and operational efficiency. This significance is underscored by the fact that optimized vehicle routing can significantly reduce operational costs, a critical factor in the highly competitive and cost-sensitive transportation

Corresponding author: Ran Wang.

Rixin Wu, Ran Wang, Jie Hao, and Qiang Wu are with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China, and also with the Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing, Jiangsu 210038, China (e-mail: wurixin@nuaa.edu.cn; wangran@nuaa.edu.cn; haojie@nuaa.edu.cn; wu.qiang@nuaa.edu.cn).

Ping Wang is with the Department of Electrical Engineering and Computer Science, Lassonde School of Engineering, York University M3J 1P3, Canada (e-mail: pingw@eecs.yorku.ca).

D. Niyato is with the School of Computer Science and Engineering, Nanyang Technological University 639798, Singapore (e-mail: dniyato@ntu.edu.sg).

industry [1]. As the demand for precise delivery schedules escalates and the transportation sector becomes more competitive, incorporating customer delivery time windows has become not just a preference but a necessity. This evolution has catapulted the study of the vehicle routing problem with time windows (VRPTW) to the forefront of transportation research, marking it as a key area for innovative solutions in traffic and logistics management [2], [3]. The advancements in VRPTW are not just academic pursuits; they directly translate to more efficient, reliable, and cost-effective transportation systems, reinforcing the backbone of global trade and commerce.

In the study of VRPTW, time windows are typically categorized into two distinct types: hard time windows and soft time windows. In scenarios involving hard time windows, vehicles are obligated to adhere strictly to the scheduled delivery times. Specifically, for a VRP with a hard time window, the vehicle must arrive and conduct deliveries within the predefined time window. In instances where the vehicle arrives before the onset of the hard time window, it is requisite for the vehicle to wait until the specified start time before proceeding with the delivery. Conversely, if the vehicle arrives within the defined hard time window, it is authorized to proceed with the delivery directly. However, delivering goods subsequent to the expiration of the hard time window is strictly prohibited [4], [5].

In the VRP with soft time windows, there exists a permissible degree of flexibility for delivery vehicles to operate outside the designated time windows [6], [7]. This flexibility, however, often incurs a consequential trade-off with customer satisfaction levels. An optimization strategy prioritizing the shortest path distance (or minimal cost) can result in the infringement of time windows for certain customers, thereby adversely affecting overall customer satisfaction [8]. To enhance customer satisfaction, managers may find themselves compelled to augment the fleet size or extend the driving routes, which paradoxically escalates transportation costs. This scenario elucidates a prevalent conflict in VRPTW: the dichotomy between minimizing transportation costs and maximizing customer satisfaction. Consequently, the incorporation of multiobjective optimization emerges as a pivotal consideration within VRPTW.

In the domain of single-objective VRPTW (SOVRPTW) which predominantly focuses on minimizing travel costs, constraints such as time windows, vehicle capacity, and the number of available vehicles need to be considered simul-

taneously, hence significantly amplifying the complexity of solving SOVRPTW. Presently, the most commonly used algorithms for solving SOVRPTW are heuristic algorithms, such as genetic algorithm [9], particle swarm algorithm [10], etc. The typical procedure for solving SOVRPTW using these algorithms involves the random generation of an initial set of solutions, followed by the application of heuristic strategies to enhance the quality of these solutions, thereby obtaining more cost-effective routes [9]. Particularly in largerscale VRPTW instances, generating initial solutions that meet the constraints can be time-consuming, and the quality of these initial solutions is often low. These challenges become even more pronounced when dealing with multiobjective VRPTW (MOVRPTW), which simultaneously considers driving costs and customer satisfaction. Therefore, the development of an effective algorithm for solving MOVRPTW becomes an urgent and pressing need.

In this paper, we propose an MOVRPTW model which takes into account both travel cost and customer satisfaction in a holistic manner. To address this model, we develop a novel multiobjective optimization algorithm that combines weight-aware deep reinforcement learning (WADRL) and the non-dominated sorting genetic algorithm-II (NSGA-II) to solve MOVRPTW. The main contributions of this paper are summarized as follows:

- To mitigate the extensive training time typically associated with training individual subproblems in deep reinforcement learning (DRL) for multiobjective optimization, we have innovatively conceptualized and implemented a WADRL method. Specifically, during each training process, a set of random weights is generated to describe the relationship between the two objective functions, and a set of continuous weight combinations are adopted during the test process to obtain vehicle routes under different weight combinations. Such a method enables only one single DRL model to obtain the Pareto front of the entire multiobjective optimization problem.
- The implementation of the WADRL approach, while innovative and time-saving, may lead to suboptimal performance, as the solutions generated cannot assure Pareto optimal. To enhance solution quality and achieve an ideal Pareto front, we propose integrating the initial solutions generated by WADRL into the NSGA-II. This hybrid approach not only improves solution quality but also addresses the challenge of time-consuming initial solution generation often encountered by traditional heuristic algorithms when dealing with complex constraint problems.
- To the best of our knowledge, based on the existing literature, our study represents the first application of DRL in solving the MOVRPTW. The experimental results demonstrate that our method markedly surpasses existing traditional methods in terms of both convergence speed and solution diversity, particularly in the context of MOVRPTW. Concurrently, the implementation of a weight-aware strategy within the DRL framework significantly reduces training time and yields superior solutions.

The remainder of this paper is organized as follows. In Section II, related work is introduced. We present the system model and formulation of MOVRPTW in Section III. In Section IV, the details of WADRL combined with NSGA-II to solve multiobjective optimization problems are introduced. Simulation results and discussions are presented in Section V. Finally, this paper is concluded in Section VI.

II. Related Work

In recent years, there has been significant research focus on the vehicle routing problem with time windows (VRPTW), leading to the development of numerous algorithms aimed at solving this problem. These algorithms can be categorized into three main groups: exact algorithms [11]–[13], heuristic algorithms [9], [10], [14]–[19], and learning-based algorithms [20], [21].

Exact algorithms approach problems by rigorously deriving mathematical formulations, thus establishing mathematical models for the problem, and proposing algorithms to find optimal solutions based on mathematical principles. The authors in [11] employed the column generation method to achieve the shortest paths in VRPTW. Furthermore, branch and bound [12] as well as branch and price [13] methods have also been applied in the context of VRPTW. However, it's worth noting that exact algorithms, which seek optimal solutions from all feasible solutions according to certain rules, face exponential growth in search space and computational complexity as the number of customers in VRPTW increases. Moreover, these algorithms tend to be less effective when dealing with multiobjective VRPTW (MOVRPTW).

Heuristic algorithms are the most commonly used methods in solving VRPTW. In solving single-objective VRPTW (SOVRPTW), common approaches encompass genetic algorithm (GA) [9], [14], particle swarm algorithm (PSO) [10], and tabu search (TS) algorithm [15], among others. In [9], Ursani *et al.* employed a local genetic algorithm to implement small-scale VRPTW targeting the shortest path and produced better solutions than most other heuristics methods. Another genetic algorithm, which was centered on the insertion heuristic, was proposed for the resolution of VRPTW, as documented in [14]. The experimental results revealed the algorithm was able to find the solution in less time. Furthermore, both PSO with local search [10] and the TS algorithm [15] were also used to solve SOVRPTW.

In real-life scheduling optimization, increasing demands have rendered SOVRPTW less capable. Consequently, many researchers have delved into the study of MOVRPTW. Jaber Jemai *et al.* employed the non-dominated sorting genetic algorithm-II (NSGA-II) to simultaneously optimize the minimization of both total travel distance and carbon dioxide emissions [16]. In ambulance route optimization, NSGA-II and the multiobjective particle swarm optimization (MOPSO) algorithm were employed to jointly optimize the latest service completion times and the number of patients whose medical conditions deteriorated due to delayed medical services [17]. Additionally, multiobjective evolutionary algorithm based on decomposition (MOEA/D) [18] and multiobjective simulated annealing (MOSA) algorithm [19] have also been utilized in addressing MOVRPTW.

In the process of employing heuristic algorithms for solving MOVRPTW, it is common practice to begin by generating a set of initial solutions that adhere to the problem's constraints. Subsequently, a set of heuristic strategies, such as those found in genetic algorithms (GA), including selection, crossover, and mutation operations, in conjunction with non-dominated sorting strategies, is applied to obtain the Pareto front. This approach is advantageous due to its simplicity and its capability to yield high-quality solutions. However, it is worth noting that heuristic algorithms typically generate initial solutions using random strategies, and for MOVRPTW instances with numerous constraints, obtaining a feasible solution can be time-consuming. Moreover, the overall effectiveness of the algorithm is greatly influenced by the quality of the initial solutions [22]. Additionally, whenever there is any modification to the problem's information, even minor changes, it necessitates rerunning the heuristic algorithm, which is known as the *No Free Lunch Theorem* [23]. Moreover, when the dimension of the problem is particularly large, these algorithms require a substantial number of iterations for overall updates and iterative searches to achieve relatively favorable results, leading to extended computation times [20].

With the advancement of artificial intelligence (AI) technology, deep reinforcement learning (DRL) has also been employed to tackle multiobjective optimization problems. DRL was initially proposed for solving multiobjective travelling salesman problems (MOTSP) in [20], in which the authors decomposed the MOTSP into several subproblems through weight combinations. They employed a sequence-to-sequence pointer network with two recurrent neural networks (RNNs) to train models for each subproblem. Additionally, a parameter transfer strategy was used in the initialization process of model parameters for adjacent subproblem training. In [21], a similar approach was adopted, where the authors decomposed multiobjective vehicle routing problem (MOVRP) into multiple scalar subproblems based on weights. They utilized the pointer network to address each subproblem and trained the parameters of the policy network using the policy gradient algorithm of reinforcement learning to obtain the Pareto front of MOVRP.

The general method of deep reinforcement learning to solve multiobjective optimization problems is to convert the problem into multiple subproblems based on multiple sets of weight combinations with the same interval. Each subproblem can be regarded as a single objective optimization problem. Subsequently, a DRL-based model is trained for each subproblem, and only after training all models for the subproblems can the entire multiobjective optimization problem be addressed. While this method may be effective for biobjective optimization problems, as the number of objective functions increases, the training time escalates exponentially, which becomes impractical for solving it. Additionally, it is challenging to ensure that the DRL-based models for each subproblem are adequately trained, causing the solutions of some subproblems to be dominated by the solutions of other subproblems [20], rendering these solutions meaningless in multiobjective optimization problems.

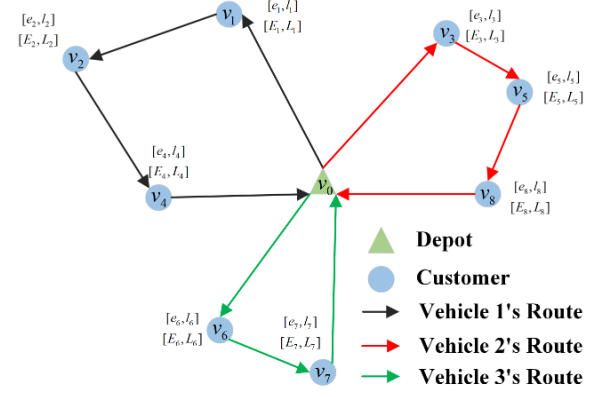Considering the limitations of DRL in solving multiob-

Fig. 1. A MOVRPTW instance with three vehicles and eight customers.

jective optimization problems, we propose WADRL, which enables a single DRL model to address the entire multiobjective optimization problem. Furthermore, to overcome the drawbacks of heuristic algorithms in solving multiobjective optimization problems, we utilize the solutions generated by WADRL as the initial solutions of NSGA-II, ensuring that the initial solutions are feasible and of high quality. After undergoing evolution with NSGA-II, it is basically guaranteed that all solutions are Pareto optimal.

III. PROBLEM STATEMENT

A. System Model

The logistics businesses typically schedule service appointments with customers through phone calls or text messages before delivering goods or providing services. However, the delivery vehicles may arrive earlier or later than the scheduled time due to inefficient delivery routes or traffic jams. Customers usually have some level of tolerance for delays or early arrivals, but this may reduce customer satisfaction. Therefore, logistics companies should aim to improve customer satisfaction while simultaneously reducing travel cost [24].

We consider a multivehicle routing system in which K vehicles are instructed to deliver goods to the set H of customers. As illustrated in Fig. 1, the scenario involves three vehicles tasked with delivering goods to eight customers. Our system is formally conceptualized as a graph $G = (V, E)$, where the set of vertices V comprises a depot vertex v_0 and a set of customer vertices $C = \{v_i, i = 1, 2, \ldots, h\}$. The edges E in the graph represent the roads connecting these vertices, each associated with a specific travel cost.

The coordinate of the depot is u_0, where each vehicle k must depart from and return to. The information of each customer v_i includes the coordinate c_i, the demand for goods d_i, the interval $[e_i, l_i]$ and the interval $[E_i, L_i]$. Among them, the interval $[e_i, l_i]$ denotes the soft time window for customer i. Service provided to the customer within the intervals $[E_i, e_i]$ or $[l_i, L_i]$ leads to a decrease in customer satisfaction. The customer being served outside of the hard time interval $[E_i, L_i]$ is not permitted. The definition of the hard time window interval is typically as follows:

TABLE I
NOMENCLATURE USED IN THIS PAPER

Symbol	Definition
H^*	Set of customer vertices
H	Set of customer and depot vertices, $H = H^* \cup \{0\}$
K	Set of vehicles
x_{ijk}	Whether vehicle k travels from customer i to customer j
y_{ijk}	Whether customer i is served by vehicle k
c_k^1	Travel cost of vehicle k per unit distance
c_k^2	Fixed cost generated by using vehicle k
a_i	Time when the vehicle arrives at customer i
$S_i(a_i)$	Satisfaction of customer i when the vehicle arrives at customer i at time a_i
$[e_i, l_i]$	Soft time window of customer i
$[E_i, L_i]$	Hard time window of customer i
u_k	Capacity of vehicle k
w_i	Waiting time of customer i
v_i	Service time of customer i
t_{ij}	Time required to move from customer i to customer j
dis_{ij}	Distance between customer i and customer j

$$E_i = e_i - \zeta_i^e (l_i - e_i), \tag{1}$$

$$L_i = l_i + \zeta_i^l (l_i - e_i), \tag{2}$$

where ζ_i^e and ζ_i^l are the parameters used to define the allowable maximum violation. We define $S_i(a_i)$ as the satisfaction of customer i when the vehicle arrives at customer i at time a_i, and $S_i(a_i)$ can be defined as:

$$S_i(a_i) = \begin{cases} 0, & a_i < E_i \vee a_i > L_i \\ \frac{a_i - E_i}{e_i - E_i}, & E_i \leq a_i < e_i \\ 1, & e_i \leq a_i \leq l_i \\ \frac{L_i - a_i}{L_i - l_i}, & l_i < a_i \leq L_i \end{cases}. \tag{3}$$

The key nomenclature used in this paper, along with their respective definitions, is outlined in TABLE I.

B. Problem Formulation

In this section, we propose a multiobjective vehicle routing problem with time windows model that considers the cost incurred by travel and vehicle usage as well as customer satisfaction, as follows:

The decision variables in this paper are defined as follows:

$$x_{ijk} = \begin{cases} 1, & \text{if vehicle } k \text{ travels from customer } i \text{ to } j \\ 0, & \text{otherwise,} \end{cases} \tag{4}$$

$$y_{ik} = \begin{cases} 1, & \text{if customer } i \text{ is served by vehicle } k \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

The objective functions in this paper are defined as follows:

Objective function 1: minimizing the total travel cost, which can be expressed as:

$$\min f_1 = \sum_{k \in K} c_k^1 \sum_{i \in H} \sum_{j \in H} dis_{ij} x_{ijk} + \sum_{k \in K} c_k^2 \sum_{j \in H^*} x_{0jk}, \tag{6}$$

where c_k^1 represents the travel cost of vehicle k per unit distance, c_k^2 denotes the fixed cost associated with the utilization of vehicle k, and dis_{ij} represents the distance between customer i and customer j.

Objective function 2: maximizing the average customer satisfaction, which can be expressed as:

$$\max f_2 = \frac{1}{h} S_i(a_i), \tag{7}$$

where a_i represents the arrival time of the vehicle at the location of customer i. Furthermore, $S_i(a_i)$ is defined as the satisfaction of customer i when the vehicle arrives at customer i at time a_i.

Subject to:

$$\sum_{i \in H^*} d_i y_{ik} < u_k \quad \forall k \in K \tag{8}$$

$$\sum_{k \in K} y_{ik} = 1 \quad \forall i \in H^* \tag{9}$$

$$\sum_{j \in H^*} x_{0jk} - \sum_{i \in H^*} x_{i0k} = 0 \quad \forall k \in K \tag{10}$$

$$\sum_{i \in H} x_{ijk} = y_{jk} \quad \forall k \in K, \forall j \in H^* \tag{11}$$

$$\sum_{j \in H} x_{ijk} = y_{ik} \quad \forall k \in K, \forall i \in H^* \tag{12}$$

$$w_0 = v_0 = 0 \tag{13}$$

$$w_i = \max\{0, E_i - a_i\} \quad \forall i \in H^* \tag{14}$$

$$E_i \leq a_i + w_i \leq L_i \quad \forall i \in H^* \tag{15}$$

$$a_j = \sum_{k \in K} \sum_{i \in H} x_{ijk}(a_i + w_i + v_i + t_{ij}) \quad \forall j \in H^* \tag{16}$$

$$x_{ijk} \in \{0, 1\}, y_{ik} \in \{0, 1\}, a_i \geq 0 \quad \forall i \in H, j \in H, k \in K \tag{17}$$

In the aforementioned mathematical model, let H^* represent the set of customer vertices, H represent the combined set of customer and depot vertices, defined as $H = H^* \cup \{0\}$. The set of vehicles is denoted by K. The constraint (8) ensures that the demand of customers does not exceed the capacity of each vehicle, where u_k represents the capacity of vehicle k. The constraint (9) mandates that each customer is served by exactly one vehicle. The constraint (10) guarantees that each vehicle must start and end its route at the depot. The constraints (11) and (12) ensure that each customer is served exactly once. The constraint (13) defines the waiting and service times of the depot, with w_i and v_i denoting the waiting and service
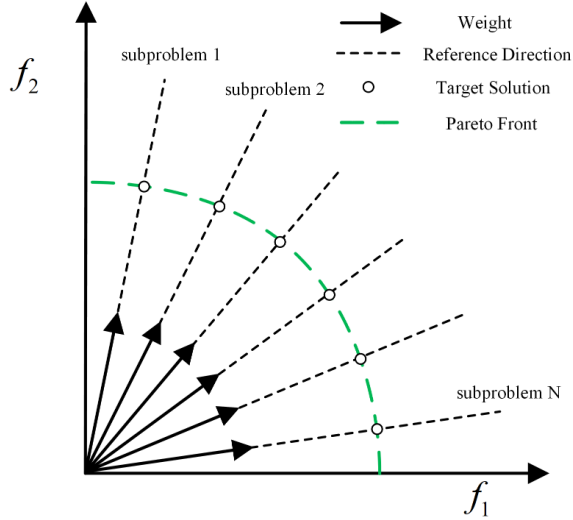
Fig. 2. The decomposition strategy used in DRL for mops.

times of customer i, respectively. The constraint (14) defines the waiting time. The constraint (15) enforces the hard time window requirements. Finally, the constraint (16) defines the relationship between the arrival time of a customer and the arrival time of the previous customer, and t_{ij} represents the time required to travel from customer i to customer j.

IV. SOLUTIONS AND ALGORITHMS

Considering the complexity of multiobjective vehicle routing problem with time windows (MOVRPTW), particularly in scenarios involving a large number of customers, existing scheduling methods seem insufficient to effectively solve the problem. Therefore, this section introduces an innovative approach that amalgamates a weight-aware deep reinforcement learning (WADRL) methodology with the non-dominated sorting genetic algorithm-II (NSGA-II). We design this hybrid method to tackle the challenges posed by MOVRPTW. Initially, the WADRL algorithm is employed to generate the initial population for the NSGA-II algorithm. Subsequently, the NSGA-II algorithm undertakes the task of optimizing this initial population, thus enhancing the solution's efficacy.

A. General Framework

In traditional deep reinforcement learning (DRL) methods for solving multiobjective optimization problems, as illustrated in Fig. 2, a biobjective optimization problem is decomposed into N subproblems according to N weight combinations. For the solution of each subproblem, the training of a DRL model is required. Consequently, a total of N DRL models need to be trained to solve the biobjective optimization problem [20]. This approach appears to be effective for solving biobjective optimization problems. However, as the number of objective functions increases, the time required to solve the entire multiobjective optimization problem will increase exponentially.

Considering the shortcomings of the existing DRL methods to solve multiobjective optimization problems, our proposed

algorithm takes a different approach. Similarly, we decompose the entire multiobjective optimization problem into N subproblems based on N weight combinations. During the training process of DRL, the algorithm randomly selects a weight combination for training each time. Furthermore, during the testing phase, the algorithm generates the corresponding optimal solutions for these N subproblems. This approach enables a single DRL model to solve the entire multiobjective optimization problem, significantly reducing the model training time, particularly for many-objective optimization problems. The WADRL framework is drawn in Fig. 3, in which a novel transformer architecture is also adopted.

In the original NSGA-II algorithm, the initial population is typically generated through a random process, and if the solution does not satisfy the constraints, it will be regenerated. In MOVRPTW, it is obviously impossible to generate solutions that satisfy constraints through a completely random method. While there exist methods to expedite the initial solution generation, they are still relatively inefficient [25], and the quality of these initial solutions is generally low. In our proposed algorithm, solutions generated by WADRL are used as the initial solutions for the NSGA-II algorithm. NSGA-II is then applied to optimize these solutions. The advantage of this approach is that the initial solutions generated by WADRL are guaranteed to satisfy the constraints. Furthermore, the generation of these initial solutions adheres to a carefully designed decomposition strategy. This approach guarantees a relatively uniform distribution across the Pareto front of the initial solutions. Importantly, this strategy not only facilitates coverage but also significantly enhances the quality of the initial solutions.

B. The Markov Decision Process of MOVRPTW in Weight-aware Deep Reinforcement Learning

Given the proficiency of reinforcement learning in managing sequential decision-making problems, we employ deep reinforcement learning to address the multiobjective vehicle routing problem with time windows (MOVRPTW). In this approach, MOVRPTW is modeled as a markov decision process (MDP), characterized by specific components: state S, action A, state transition function, and reward R. These elements are defined as follows.

1) State: At any given step t, the state of the system encompasses three primary components: the vehicle state v^t, customer state c^t, and weight state w^t, defined as follows. The vehicle state v^t is characterized by two dynamic attributes: its load, denoted as u^t, and the traveled time, symbolized as τ^t. The term 'dynamic' implies that these states may change as actions occur. The customer states encompass static states, including soft and hard time windows e_i^t, l_i^t, E_i^t and L_i^t, coordinate c_i^t, and service time v_i^t. Static states indicate that these states will not change with the occurrence of actions. The dynamic state of a customer is demand d_i^t. The weight state is static, signifying the weights of two objective functions w_1^t and w_2^t.

2) Action: The action at each step t, denoted as a^t, represents the next vertex to which the vehicle will travel. The entire

sequence of actions from the initial step 0 to the final step T is represented as $\{a^0, a^1, a^2 \ldots, a^T\}$. $a^t \in \{0, 1, \ldots, h\}$, where h is the total number of vertices in the system. $a^t = 0$ means that the vehicle returns to the depot. $a^t \in \{1, 2, \ldots, h\}$ signifies that the vehicle travels towards the customer vertex. The vehicle is initially at the depot vertex, i.e. $a^0 = 0$.

3) State transition function: The current system state S^t will transition to the next state S^{t+1} based on the currently executed action a^t. All static states remain unchanged, while dynamic states may change. Once a customer node is visited, the demand of the customer becomes zero, which can be expressed as:

$$d_i^t = \begin{cases} 0, & \text{if } a^t = i \\ d_i^{t-1}, & \text{otherwise .} \end{cases} \tag{18}$$

In addition, the state of the vehicle will also change. If the vehicle travels to a customer node, its load will decrease accordingly. Otherwise, if the vehicle travels to the depot node, its load will be replenished, as given by:

$$u^t = \begin{cases} u^{t-1} - d(a^t), & \text{if } a^t \neq 0 \\ u^0, & \text{if } a^t = 0. \end{cases} \tag{19}$$

The traveled time is determined by the travel time between two vertices and previous traveled time, as given as follows:

$$\tau^t = \begin{cases} \max(\tau^{t-1}, E(a^{t-1})) \\ \quad + v(a^{t-1}) + t(a^{t-1}, a^t), & \text{if } a^t \neq 0 \\ 0, & \text{if } a^t = 0, \end{cases} \tag{20}$$

where $E(a^{t-1})$ denotes the earliest service start time of the customer a^{t-1}, $v(a^{t-1})$ represents the service time of the customer a^{t-1}, and $t(a^{t-1}, a^t)$ is the travel time between the two vertices. The vehicle returning to the depot means dispatching a new vehicle and updating the traveled time.

4) Reward: By executing actions $\{a_0, a_1, a_2 \ldots, a_T\}$, the traveled paths of all vehicles, as well as the arrival time and satisfaction of all customers, can be obtained. The objective function values can be calculated according to Eq. (6) and Eq. (7), and the total reward R can be calculated using Eq. (21):

$$R = \begin{cases} -1000, & \text{if } \sum_{i \in N^*} d_i^T \neq 0 \\ -w_1 \frac{f_1 - f_1^{\min}}{f_1^{\max} - f_1^{\min}} + w_2 \frac{f_2 - f_2^{\min}}{f_2^{\max} - f_2^{\min}}, & \text{otherwise ,} \end{cases} \tag{21}$$

where $R = -1000$ indicates that after all vehicles have delivered the goods and returned to the depot, there are still customer vertices that have not been visited, which means the constraints are not met. Otherwise, the minus sign before the first objective function means that the objective function is a minimization function. f_1^{min}, f_1^{max}, f_2^{min} and f_2^{max} denote the minimum and maximum values of the two objective functions respectively, which aim to normalize the objective function. These values can be obtained using single objective DRL model, objective function structures or the existing research.

C. The Policy network in Weight-aware Deep Reinforcement Learning

In our system model, various information needs to be considered simultaneously, such as customer coordinates, demand, time windows, and the weights of the objective functions. Therefore, learning-based methods need to be carefully designed to process the information. However, simple neural networks or learning strategies often struggle to handle the complex information mentioned above. Meanwhile, transformer architecture, as a type of self-attention mechanism, has been proven to perform well in many fields, including natural language processing (NLP) [26], computer vision (CV) [27], and recommender systems [28]. With the development of transformer technology, it has found wide applications in deep reinforcement learning, displaying excellence in problems like path planning [7], the knapsack problem [29], and reservoir operation [30]. Its advantage is that the attention mechanism in the transformer can effectively extract information through key-value-query maps.

In this paper, we employ the transformer architecture [31] to model the delivering agent for solving MOVRPTW, as illustrated in Fig. 3. The model primarily comprises an encoder module, a weight embedding module, and a decoder module [31]. In the encoder process, the encoder embeds the original information of MOVRPTW into a high-dimensional space and utilizes a self-attention mechanism to extract features of the problem. In the weight embedding process, the embedding module encapsulates the state of vehicles and the weights associated with the two objectives of MOVRPTW. In the decoder process, the decoder generates vertex probabilities based on contextual information.

1) The information encoder process: Initially, the encoder employs a linear layer to transform the features of the vertices (including both the depot and customers) into a high-dimensional space. This transformation results in the generation of initial embedding information, represented as $h^{(0)} = \{h_0^{(0)}, h_1^{(0)}, \ldots, h_N^{(0)}\}$, where $h_i^{(0)}$ corresponds to the initial embedded representation of vertex i. We employ separate linear projections to handle distinct information of warehouse and customer vertices, as follows [32]:

$$h_i^{(0)} = \begin{cases} W_0^x [c_0, L_0] + b_0^x, & \text{if } i = 0 \\ W_i^x [c_i, E_i, e_i, l_i, L_i, d_i, v_i] + b_i^x, & \text{if } i \neq 0, \end{cases} \tag{22}$$

where the depot vertex only needs to embed coordinate and latest arrival time information, while the customer vertices need to embed coordinate, soft and hard time window, demand, and service time information. The operation $[\cdot, \cdot, \ldots, \cdot]$ concatenates the information of the same dimension together. W_0^x, b_0^x, W_i^x and b_i^x are trainable linear projection parameters, and $W_0^x \in \mathbb{R}^{d_h \times 2}$, $b_0^x \in \mathbb{R}^{d_h}$, $W_i^x \in \mathbb{R}^{d_h \times 8}$, $b_i^x \in \mathbb{R}^{d_h}$, where d_h represents the dimension of $h^{(0)}$. In our algorithm $d_h = 128$.

Subsequent to the initial embedding, the encoded information undergoes processing through L identical layers to yield the final embedding. Each of these layers is composed of several key components: multi-head attention layer (MHA), skip connection layer, batch normalization (BN) layer,
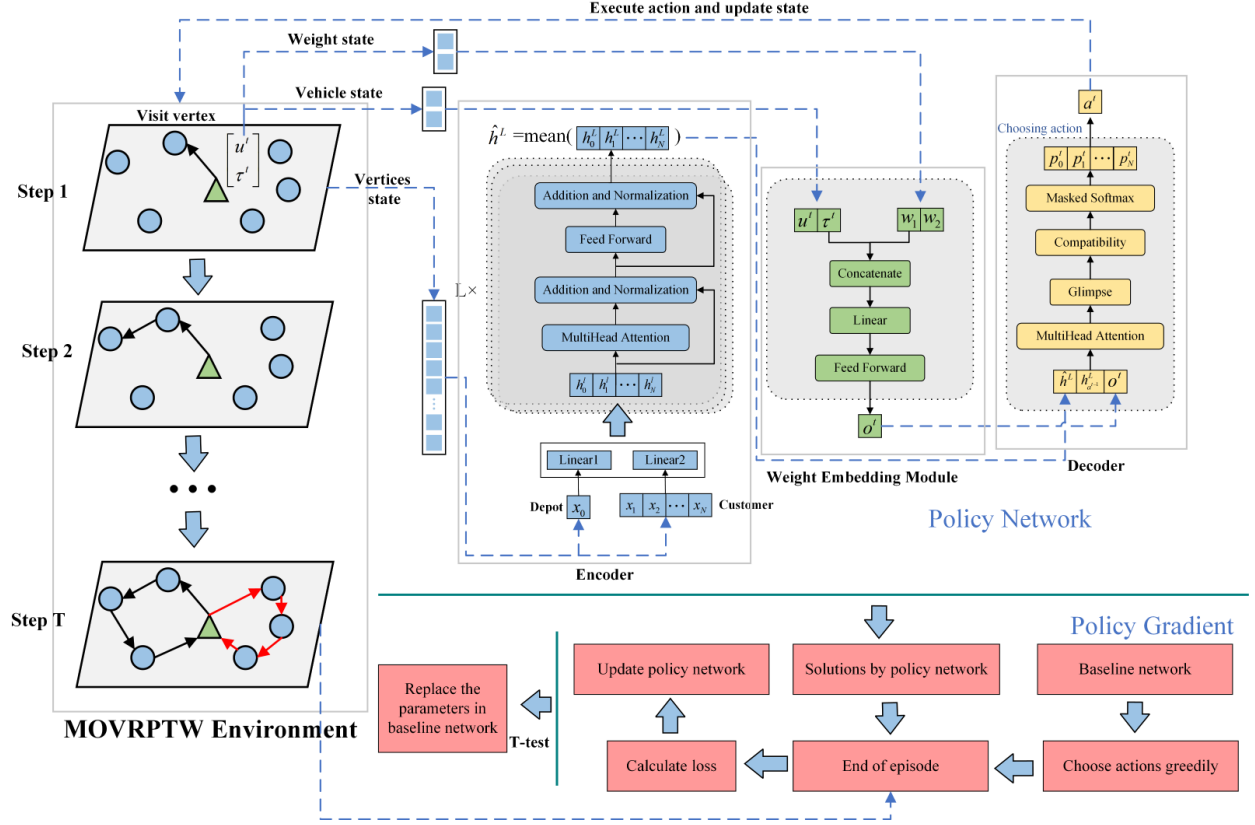
Fig. 3. The framework of WADRL method.

and fully connected feedforward (FF) layer. The input of layer l is the output of the preceding layer, i.e. $h^{(l-1)} = \left\{ h_0^{(l-1)}, h_1^{(l-1)}, \ldots, h_N^{(l-1)} \right\}$. The MHA layer employs M heads with a dimensionality of d_k. In our algorithm, we set $L = 3$, $M = 8$ and $d_k = \frac{d_h}{M} = 16$. For each head $m \in \{1, 2, \ldots, M\}$, the query, key, and value are computed as follows [33]:

$$Q_m^{(l)} = W_{Qm}^{(l)} h^{(l-1)}, \qquad (23)$$

$$K_m^{(l)} = W_{Km}^{(l)} h^{(l-1)}, \qquad (24)$$

$$V_m^{(l)} = W_{Vm}^{(l)} h^{(l-1)}, \qquad (25)$$

where $W_{Qm}^{(l)}, W_{Km}^{(l)} \in \mathbb{R}^{d_k \times d_l}$, and $W_{Vm}^{(l)} \in \mathbb{R}^{d_h \times d_h}$. Then the attention value $A_m^{(l)}$ and $MHA\left(h^{(l-1)}\right)$ are calculated as:

$$A_m^{(l)} = \text{softmax}\left(\frac{Q_m^{(l)} \left(K_m^{(l)}\right)^T}{\sqrt{d_k}}\right) V_m^{(l)}, \qquad (26)$$

$$MHA\left(h^{(l-1)}\right) = \left[A_1^{(l)}, A_2^{(l)}, \ldots, A_M^{(l)}\right] W_O^{(l)}. \qquad (27)$$

$\hat{h}^{(l)}$ is obtained from the attention value through skip connection layer and BN layer, which can be expressed as follows:

$$\hat{h}^{(l)} = BN^{(l)}\left(h^{(l-1)} + MHA\left(h^{(l-1)}\right)\right). \qquad (28)$$

Ultimately, the output of the node embedding at layer l is denoted as $h^{(l)}$, which can be obtained through an FF layer and a skip connection layer as follows:

$$h^{(l)} = BN^{(l)}\left(\hat{h}^{(l-1)} + FF\left(\hat{h}^{(l-1)}\right)\right). \qquad (29)$$

2) The weight embedding process: As MOVRPTW is modeled as an MDP, each vehicle is treated as an agent. When the agent selects actions, it needs to consider the weights of the two objective functions in MOVRPTW. Hence, we incorporate a specialized weight embedding module within our framework. This module is carefully designed to capture the current state of the vehicle and the state of the weights associated with the objective functions. The strategy allows the agent to focus on the weights of the objective functions when making decisions. The output of the weight embedding module is defined as follows:

$$o^t = FF\left(W_o\left[u^t, \tau^t, w_1, w_2\right] + b_o\right). \qquad (30)$$

3) The tour decoder process: At each given step t, the agent selects a decision a^t based on the current state s^t, which contains the embedding of the entire graph denoted as $\hat{h}^{(L)}$, the embedding information of the vertex corresponding to the vehicle's current location $h_{a^{t-1}}^{(L)}$ and the output o^t of the weight embedding module as follows:

Algorithm 1 The Training Process of Weight-Aware Deep Reinforcement Learning

---

Input: Batch size and training epoch
Output: The trained parameters θ

1: Initialization θ for policy network π^θ, θ^{BL} for baseline network π^{BL};
2: Generate an MOVRPTW instance randomly;
3: **for** each epoch **do**
4:   **for** each batch **do**
5:     **for** each step t **do**
6:       Calculate vertex embedding in Eqs.(22)-(29);
7:       Calculate weight embedding in Eq.(30);
8:       Calculate vertex probability vector in Eqs.(31)-(37);
9:       Choose action a^t;
10:      **if** All available vehicles are returned to the depot vertex **or** all customer vertices are visited **then**
11:        Break;
12:      **end if**
13:    **end for**
14:    Calculate the reward $R\left(\pi^\theta\right)$ and $R\left(\pi^{BL}\right)$ in Eq.(21);
15:    Calculate $\nabla_\theta L(\theta)$;
16:    Update the parameters θ;
17: **end for**
18: **if** T-Test result < 0.95 **then**
19:    $\theta^{BL} \leftarrow \theta$;
20: **end if**
21: **end for**

$$s^t = \left[\hat{h}^{(L)}, h^{(L)}_{a^{t-1}}, o^t\right]. \tag{31}$$

Subsequently, we calculate the contextual information of s^t through the MHA layer, defining the query vector as the embedding of the agent, and the key vector and value vector as the embedding of customers:

$$Q^{dt} = W^d_Q s^t, \tag{32}$$

$$K^d = W^d_K h^L, \tag{33}$$

$$V^d = W^d_V h^L. \tag{34}$$

The compatibility between each vertex and the vehicle is calculated through an attention mechanism:

$$\lambda_t = C \cdot \tanh\left(\frac{\left(Q^{dt}\right)^T K^d}{\sqrt{d_k}}\right). \tag{35}$$

In MOVRPTW, various constraints are established, meaning that at each step t, certain vertices are inaccessible due to these constraints. To navigate the diverse constraints inherent in the system, we also implement a masking rule. This rule functions to selectively exclude vertices that are not feasible for visitation during the current step. The primary masking rules include the following: ① Apart from the depot vertex, other
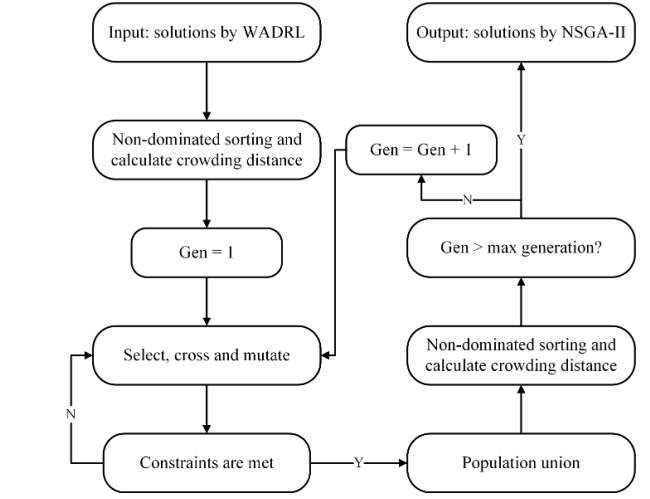

Fig. 4. The process of combining WADRL and NSGA-II.

previously visited vertices are masked; ② Customer vertices with demand exceeding the load of vehicle are masked; ③ Vertices where the vehicle's arrival time at the node exceeds the hard time window are masked. Whether the vertex i is masked at step t is defined as:

$$\text{mask}^t_i = \left\{ \begin{array}{ll} 1, & \text{if node } i \text{ is masked at step } t \\ 0, & \text{otherwise .} \end{array} \right. \tag{36}$$

Finally, the probability vector for vertex selection is generated by combining the compatibility vector between the vehicle and the vertices:

$$p^t = \text{softmax}\left(\lambda^t + O \cdot \text{mask } k^t\right), \tag{37}$$

where O is a large negative number, e.g., $O = -999999$, which means that the customer vertex cannot be visited.

4) Model training: We employ a policy gradient method to train all parameters θ within the neural network. The algorithm comprises two networks, namely the policy network and the baseline network. Both the policy and baseline networks share the same network structure, with the only difference being that the policy network selects actions based on sampling from the probability vector, while the baseline network adopts a strategy where actions are selected based on the action possessing the highest probability, namely 'greedy policy'. The gradient of the loss function is defined as follows [32]

$$\nabla_\theta L(\theta) = E_{p_\theta(\pi^\theta)}\left[\left(R\left(\pi^\theta\right) - R\left(\pi^{BL}\right)\right)\nabla_\theta \log p_\theta\left(\pi^\theta\right)\right], \tag{38}$$

where $R\left(\pi^\theta\right)$ and $R\left(\pi^{BL}\right)$ are utilized to denote the rewards accrued by the policy network and the baseline network, respectively. During each training batch, a t-test is employed to ascertain the statistical significance of the difference in performance between these two networks at a 95% confidence level. Should this test yield a significant result, it prompts an update wherein the parameters of the policy network supersede those of the baseline network, thereby optimizing the learning process. The detailed procedural steps of this algorithm are given in Algorithm 1.

TABLE II
AVERAGE OBJECTIVE FUNCTIONS VALUES AND RUNNING TIME OF GENERATING INITIAL SOLUTIONS BY WADRL AND NSGA-II. INSTANCES OF 20-, 50- AND 100- CUSTOMER ARE LISTED.
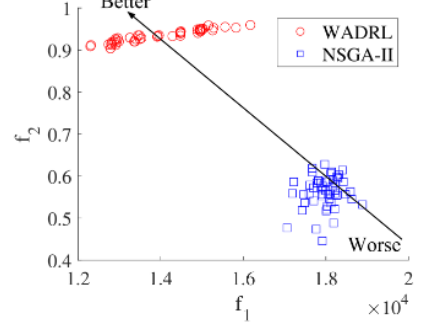
	20-customer			50-customer			100-customer		
	f_1	f_2	Time/s	f_1	f_2	Time/s	f_1	f_2	Time/s
WADRL	2812.98	0.7464	2.11	5481.24	0.9072	4.59	13953.19	0.9331	7.13
NSGA-II	3705.36	0.5452	2.07	9640.94	0.6013	9.12	17961.76	0.5616	35.58
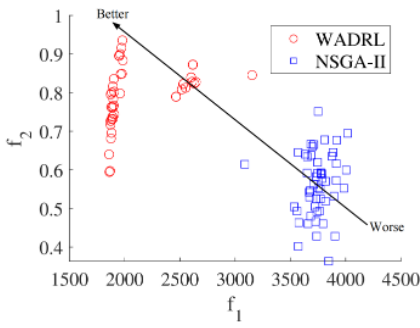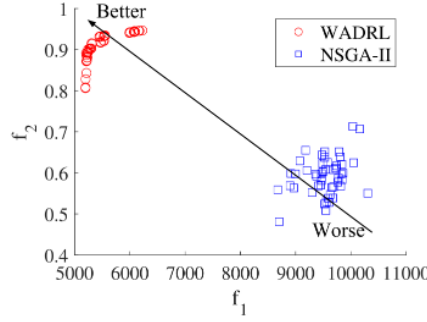
| (a) 20-customer instance | (b) 50-customer instance | (c) 100-customer instance |

Fig. 5. Initial solutions generated by WADRL and NSGA-II in the instances of 20-, 50- and 100-customer.

D. Combine Weight-Aware Deep Reinforcement Learning with NSGA-II

In the pursuit of solving the MOVRPTW, the application of WADRL may reveal promising results. However, several limitations still exist: ① The neural network sometimes struggles to iterate optimally, falling into a local optimal solution; ② Its performance may exhibit instability under specific objective weight combinations; ③ Most of the obtained solutions may be dominated by other solutions.

To address these limitations, we propose the approach that combines WADRL with NSGA-II. Our approach involves utilizing WADRL to generate initial solutions for the MOVRPTW. These initial solutions serve as the foundation for subsequent optimization through NSGA-II. WADRL ensures that the initial solutions are of high quality, meet all constraints, and have a relatively uniform distribution due to the applied decomposition strategy.

This combined approach is designed to improve the overall optimization process, enhance solution quality, and mitigate the inherent limitations of WADRL. The integration of NSGA-II complements the strengths of WADRL, providing a more robust and effective solution strategy for MOVRPTW. The process of NSGA-II optimizing the solution generated by WADRL is shown in Fig. 4.

V. RESULTS AND ANALYSIS

To demonstrate the advantages of our proposed approach that combines weight-aware deep reinforcement learning (WADRL) and non-dominated sorting genetic algorithm-II (NSGA-II), we apply it to the Solomon dataset [34]. Furthermore, we conduct comparative studies on datasets with varying
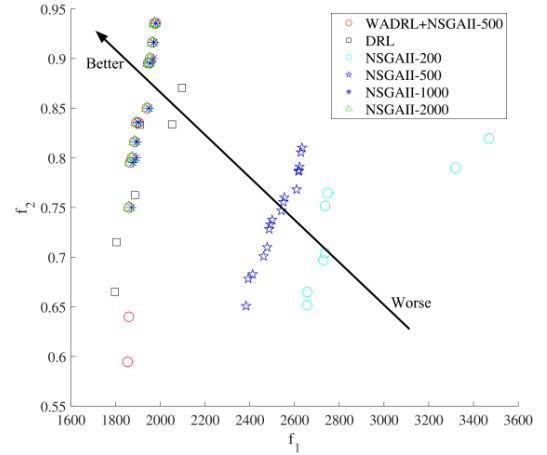

Fig. 6. 20-customer MOVRPTW problem instance: The Pareto front obtained using our method in comparison with DRL and NSGA-II. 200, 500, 1000, 2000 iterations are applied respectively.

numbers of customers, comparing our method with state-of-the-art approaches, including traditional deep reinforcement learning (DRL) and NSGA-II.

A. Experimental Setup

1) Training sets: The model is trained on instances with 20-, 50- and 100-customer vertices and a single depot vertex, respectively. In the training dataset, the demand for customer vertices is uniformly sampled from the range $[1, 40]$. For the instances with 20-, 50-, and 100-customer instances, the coordinates of the vertices are generated from a uniform distribution within the range $[0, 100]$. We randomly generate

TABLE III
AVERAGE OBJECTIVE FUNCTIONS VALUES OF THE PFS BY WADRL+NSGA-II, NSGA-II, WADRL AND DRL. INSTANCES OF 20-, 50- AND 100-
CUSTOMER ARE LISTED. THE RUNNING TIME ARE ALSO LISTED.

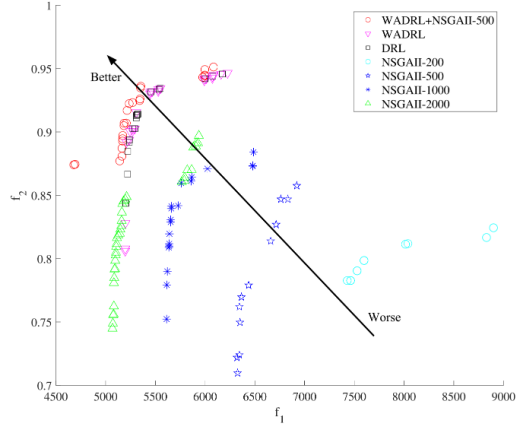|	20-customer			50-customer			100-customer		
	f_1	f_2	Time/s	f_1	f_2	Time/s	f_1	f_2	Time/s
NSGAII-200	2913.21	0.7377	9.18	7825.94	0.7977	60.29	14993.88	0.6132	169.51
NSGAII-500	2526.57	0.7421	18.69	6536.26	0.7895	72.96	14761.331	0.7754	216.09
NSGAII-1000	1929.04	0.8584	35.04	5813.88	0.8327	112.35	13581.44	0.7831	620.48
NSGAII-2000	1916.32	0.8511	68.41	5323.13	0.8237	245.65	13139.78	0.8780	1126.98
DRL	1923.99	0.78	3.15	5424.69	0.91	5.41	14026.93	0.9293	8.56
WADRL	1916.66	0.8284	2.11	5584.11	0.9122	4.59	13857.09	0.9346	7.13
WADRL+NSGAII-500	1911.56	0.8203	15.55	5339.38	0.9126	61.22	12085.43	0.9284	193.41

Fig. 7. 50-customer MOVRPTW problem instance: The Pareto front obtained using our method in comparison with DRL, WADRL and NSGA-II. 200, 500, 1000, 2000 iterations are applied respectively.

Fig. 8. 100-customer MOVRPTW problem instance: The Pareto front obtained using our method in comparison with DRL, WADRL and NSGA-II. 200, 500, 1000, 2000 iterations are applied respectively.
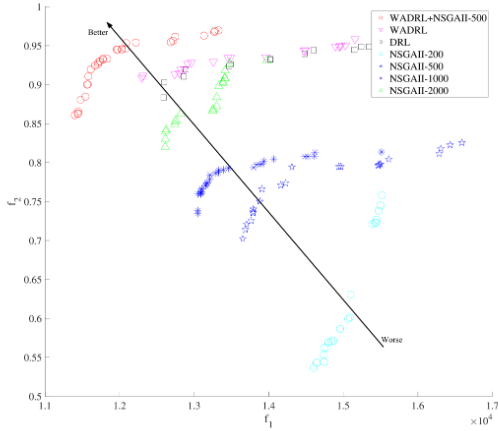
soft time windows with intervals greater than 30 within the range $[0, 240]$.

2) Test set: For the 20-customer instance, we use the information of the first 20 customers in RC101; For the 50-

customer instance, we use the information of the first 50 customers in RC101; For the 100-customer instance, we use RC102 [35].

3) Parameter settings: In MOVRPTW, the travel cost of vehicle k per unit distance and fixed cost generated by using vehicle, denoted as k $\left(c_k^1, c_k^2\right)$, are set to be fixed, as described in $(2.0, 400)$ [36]. The allowable maximum violation of time window parameters, denoted as $\left(\zeta_i^e, \zeta_i^l\right)$, are set as $(0.25, 0.25)$.

In each iteration, a set of weight combinations is extracted according to the Dirichlet distribution. We use the Adam optimizer with a learning rate of 10^{-4} with decay of 10^{-6} [37]. In total, we train the model for 300 epochs and the batch size is set to 64.

DRL and WADRL use weight combinations with an interval of 0.02 to decompose MOVRPTW. Specifically, these weight combinations are defined as [[1.00, 0.00], [0.98, 0.02], ... , [0.00, 1.00]], resulting in a total of 51 subproblems. Furthermore, we configure the population size of the NSGA-II algorithm to also be 51.

B. Comparison of Initial Solutions

We first compare the performance of our method and the original NSGA-II in generating initial solutions, as shown in Fig. 5. The results indicate that our method consistently outperforms the traditional NSGA-II algorithm in terms of the quality of initial solutions, across instances with 20-, 50-, 100-customer. Furthermore, the average execution time together with the average objective function value of the initial solution are presented in TABLE III.

As shown in Fig. 5, all initial solutions generated by the WADRL-based NSGA-II outperform the initial solutions generated by the original NSGA-II using random policies, both in terms of objective functions and time efficiency. This improvement is particularly pronounced as the number of customers increases, with our method exhibiting more significant enhancements over the traditional NSGA-II approach. For example, in the case of the 100-customer instance, our method, on average, reduces the cost by 4008.6, increases customer satisfaction by 0.3715, and decreases the time required for generating initial solutions by 28.45 seconds. Furthermore, we observe that within the solutions generated by WADRL, many

(a) 20-customer instance   (b) 50-customer instance   (c) 100-customer instance
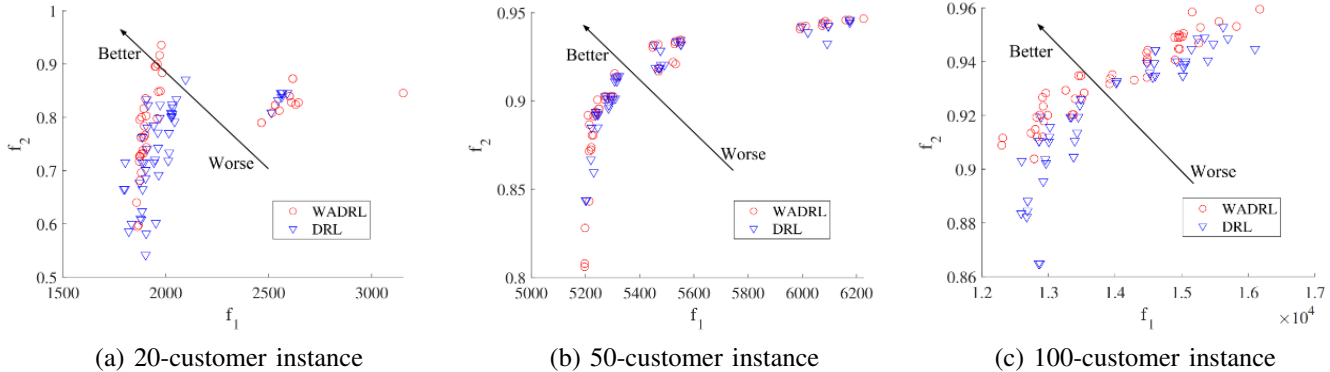
Fig. 9. Solutions generated by WADRL and DRL in the instances of 20-, 50- and 100-customer.

of them are dominated by other solutions, rendering them non-meaningful in MOVRPTW. As a result, it is reasonable to use the NSGA-II algorithm to optimize these solutions generated by WADRL.

C. Experimental Results

We implement NSGA-II algorithms with 200, 500, 1000, and 2000 iterations, as well as NSGA-II with 500 iterations using the solution of WADRL as the initial solution. We compare these results with DRL and WADRL, and the average objective function values and running times are presented in TABLE III.

Fig. 6, Fig. 7 and Fig. 8 depict the experimental results on instances of 20-, 50-, and 100-customer, respectively. Our approach (WADRL+NSGA-II) consistently achieves optimal results on all instances, particularly as the problem scale increases. WADRL itself has achieved relatively competitive results, and when combined with the NSGA-II algorithm, WADRL's solutions exhibit higher quality and extended coverage on the Pareto front.

In the 20-customer instance, our method, NSGAII-1000 and NSGAII-2000 show almost the same performance, but our method still finds solutions which have a lower travel cost. Not to mention that in the 100-customer instance, the advantage of our method is overwhelming, using less than 200 seconds to obtain effects that the NSGA-II algorithm cannot obtain in 1000 seconds.

In terms of the average objective function values, as shown in TABLE III, the WADRL+NSGA-II algorithm outperforms other benchmark algorithms under all instances. Meanwhile, the runtime of WADRL is significantly lower than that of the traditional DRL algorithms and all NSGA-II algorithms.

D. Effectiveness of Weight-aware Strategy

In this section, the weight-aware strategy in deep reinforcement learning is experimentally validated. Fig. 9 displays the results of all solutions generated by WADRL and DRL on instances on 20-, 50-, and 100-customer instances. TABLE IV presents the average training times. The results indicate that WADRL achieves superior results while utilizing only about 1/10 of the training time compared of the traditional DRL.

TABLE IV
THE TRAINING TIME OF WADRL AND DRL. INSTANCES OF 20-, 50- AND 100-CUSTOMER ARE LISTED.

	20-customer	50-customer	100-customer
WADRL	24min	48min	1h43min
DRL	1h54min	5h17min	17h32min

In the 20-customer instance, as shown in Fig. 9a, the results of DRL are relatively unstable, and the results of WADRL are relatively stable. In the 50-customer instance, as shown in Fig. 9b, the results of DRL and WADRL are almost the same, but WADRL still finds some solutions with lower travel cost. In the 100-customer instance, as shown in Fig. 9c, the results of WADRL are significantly better than those of DRL.

VI. CONCLUSION

This paper proposed a weight-aware deep reinforcement learning (WADRL) framework combined with non-dominated sorting algorithm-II (NSGA-II) to solve the multiobjective vehicle routing problem with time windows (MOVRPTW). Firstly, a comprehensive MOVRPTW model was proposed, which takes into account both travel cost and customer satisfaction. Subsequently, a WADRL framework was introduced, where the weights of objective functions were integrated into the state of deep reinforcement learning (DRL), enabling a single DRL model to solve the entire multiobjective optimization problem. An innovative DRL framework was introduced, which is centered around a transformer-based policy network. The architecture of this policy network was carefully designed, comprising three integral components: the encoder module, which is responsible for embedding customer information, the weight embedding module, which plays a critical role in capturing and encoding the weights of objective functions, and the decoder module, tasked with generating executable actions based on the contextual information. Furthermore, considering the limitations of DRL, we employed the NSGA-II algorithm to optimize the solutions generated by WADRL. Experimental results demonstrated the promising performance of our method across all MOVRPTW instances. Specifically, our method produced Pareto fronts with better coverage and

quality. Finally, the weight-aware strategy reduces the training time of DRL greatly and achieves superior performance.

Regarding future research, in the current use of WADRL to solve MOVRPTW, a single DRL model can solve the entire MOVRPTW for the same scale of customers. However, for different MOVRPTW scales, multiple models still need to be trained. We will investigate how to train a single DRL model to handle MOVRPTW instances of varying scales.

REFERENCES

[1] M. Asghari, S. M. J. M. Al-e *et al.*, "Green vehicle routing problem: A state-of-the-art review," *International Journal of Production Economics*, vol. 231, p. 107899, 2021.

[2] H. Fan, Y. Zhang, P. Tian, Y. Lv, and H. Fan, "Time-dependent multi-depot green vehicle routing problem with time windows considering temporal-spatial distance," *Computers & Operations Research*, vol. 129, p. 105211, 2021.

[3] B. Lin, B. Ghaddar, and J. Nathwani, "Deep reinforcement learning for the electric vehicle routing problem with time windows," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 8, pp. 11 528–11 538, 2021.

[4] W. Zhang, D. Yang, G. Zhang, and M. Gen, "Hybrid multiobjective evolutionary algorithm with fast sampling strategy-based global search and route sequence difference-based local search for vrptw," *Expert Systems with Applications*, vol. 145, p. 113151, 2020.

[5] H. Zhang, H. Ge, J. Yang, and Y. Tong, "Review of vehicle routing problems: Models, classification and solving algorithms," *Archives of Computational Methods in Engineering*, pp. 1–27, 2021.

[6] K. Zhang, F. He, Z. Zhang, X. Lin, and M. Li, "Multi-vehicle routing problems with soft time windows: A multi-agent reinforcement learning approach," *Transportation Research Part C: Emerging Technologies*, vol. 121, p. 102861, 2020.

[7] K. Zhang, X. Lin, and M. Li, "Transformer-based reinforcement learning for pickup and delivery problems with late penalties," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 12, pp. 24 649–24 661, 2022.

[8] Y. Xia and Z. Fu, "Improved tabu search algorithm for the open vehicle routing problem with soft time windows and satisfaction rate," *Cluster Computing*, vol. 22, no. Suppl 4, pp. 8725–8733, 2019.

[9] Z. Ursani, D. Essam, D. Cornforth, and R. Stocker, "Localized genetic algorithm for vehicle routing problem with time windows," *Applied Soft Computing*, vol. 11, no. 8, pp. 5375–5390, 2011.

[10] F. Belmecheri, C. Prins, F. Yalaoui, and L. Amodeo, "Particle swarm optimization algorithm for a vehicle routing problem with heterogeneous fleet, mixed backhauls, and time windows," *Journal of intelligent manufacturing*, vol. 24, pp. 775–789, 2013.

[11] A. Chabrier, "Vehicle routing problem with elementary shortest path based column generation," *Computers & Operations Research*, vol. 33, no. 10, pp. 2972–2990, 2006.

[12] P. Toth and D. Vigo, "Branch-and-bound algorithms for the capacitated vrp," in *The vehicle routing problem*. SIAM, 2002, pp. 29–51.

[13] L. Costa, C. Contardo, and G. Desaulniers, "Exact branch-price-and-cut algorithms for vehicle routing," *Transportation Science*, vol. 53, no. 4, pp. 946–985, 2019.

[14] G. Vaira and O. Kurasova, "Genetic algorithm for vrp with constraints based on feasible insertion," *Informatica*, vol. 25, no. 1, pp. 155–184, 2014.

[15] S. C. Ho and D. Haugland, "A tabu search heuristic for the vehicle routing problem with time windows and split deliveries," *Computers & Operations Research*, vol. 31, no. 12, pp. 1947–1964, 2004.

[16] J. Jemai, M. Zekri, and K. Mellouli, "An nsga-ii algorithm for the green vehicle routing problem," in *Evolutionary Computation in Combinatorial Optimization: 12th European Conference, EvoCOP 2012, Málaga, Spain, April 11-13, 2012. Proceedings 12*. Springer, 2012, pp. 37–48.

[17] M. Rabbani, N. Oladzad-Abbasabady, and N. Akbarian-Saravi, "Ambulance routing in disaster response considering variable patient condition: Nsga-ii and mopso algorithms." *Journal of Industrial & Management Optimization*, vol. 18, no. 2, 2022.

[18] F. Tan, Z.-y. Chai, and Y.-l. Li, "Multi-objective evolutionary algorithm for vehicle routing problem with time window under uncertainty," *Evolutionary Intelligence*, vol. 16, no. 2, pp. 493–508, 2023.

[19] A. Motaghedi-Larijani, "Solving the number of cross-dock open doors optimization problem by combination of nsga-ii and multi-objective simulated annealing," *Applied Soft Computing*, vol. 128, p. 109448, 2022.

[20] K. Li, T. Zhang, and R. Wang, "Deep reinforcement learning for multiobjective optimization," *IEEE Transactions on Cybernetics*, vol. 51, no. 6, pp. 3103–3114, 2020.

[21] J. Zhang, R. Hu, Y.-J. Wang, Y.-Y. Yang, and B. Qian, "Deep reinforcement learning for solving multi-objective vehicle routing problem," in *International Conference on Intelligent Computing*. Springer, 2023, pp. 146–155.

[22] A. Lambora, K. Gupta, and K. Chopra, "Genetic algorithm-a literature review," in *2019 international conference on machine learning, big data, cloud and parallel computing (COMITCon)*. IEEE, 2019, pp. 380–384.

[23] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997.

[24] H. Zhang, Q. Zhang, L. Ma, Z. Zhang, and Y. Liu, "A hybrid ant colony optimization algorithm for a multi-objective vehicle routing problem with flexible time windows," *Information Sciences*, vol. 490, pp. 166–190, 2019.

[25] A. Gutiérrez-Sánchez and L. B. Rocha-Medina, "Vrp variants applicable to collecting donations and similar problems: A taxonomic review," *Computers & Industrial Engineering*, vol. 164, p. 107887, 2022.

[26] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz *et al.*, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, 2020, pp. 38–45.

[27] X. Meng, N. Wang, F. Shao, and S. Li, "Vision transformer for pansharpening," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–11, 2022.

[28] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, and P. Jiang, "Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer," in *Proceedings of the 28th ACM international conference on information and knowledge management*, 2019, pp. 1441–1450.

[29] B. Yildiz, "Reinforcement learning using fully connected, attention, and transformer models in knapsack problem solving," *Concurrency and Computation: Practice and Experience*, vol. 34, no. 9, p. e6509, 2022.

[30] R. Wu, R. Wang, J. Hao, Q. Wu, and P. Wang, "Multiobjective multihydropower reservoir operation optimization with transformer-based deep reinforcement learning," *Journal of Hydrology*, p. 130904, 2024.

[31] M. Tang, W. Zhuang, B. Li, H. Liu, Z. Song, and G. Yin, "Energy-optimal routing for electric vehicles using deep reinforcement learning with transformer," *Applied Energy*, vol. 350, p. 121711, 2023.

[32] K. Zhang, X. Lin, and M. Li, "Transformer-based reinforcement learning for pickup and delivery problems with late penalties," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 12, pp. 24 649–24 661, 2022.

[33] W. Kool, H. Van Hoof, and M. Welling, "Attention, learn to solve routing problems!" *arXiv preprint arXiv:1803.08475*, 2018.

[34] "Solomon benchmark," https://www.sintef.no/projectweb/top/vrptw/solomon-benchmark/, April 18, 2008.

[35] M. M. Solomon and J. Desrosiers, "Survey paper—time window constrained routing and scheduling problems," *Transportation science*, vol. 22, no. 1, pp. 1–13, 1988.

[36] D. Taş, O. Jabali, and T. Van Woensel, "A vehicle routing problem with flexible time windows," *Computers & Operations Research*, vol. 52, pp. 39–54, 2014.

[37] T. Ye, Z. Zhang, J. Chen, and J. Wang, "Weight-specific-decoder attention model to solve multiobjective combinatorial optimization problems," in *2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2022, pp. 2839–2844.

Rixin Wu was born in Taizhou, Jiangsu, China in 2000, and obtained an B.E. degree from Nanjing University of Chinese Medicine in 2022 and is currently pursuing a M.E degree at Nanjing University of Aeronautics and Astronautics. His research focuses on deep reinforcement learning and multi-objective optimization problems.

Ran Wang (Member, IEEE) is an Associate professor and Doctoral Supervisor with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China. He received his B.E. in Electronic and Information Engineering from Honors School, Harbin Institute of Technology, China in July 2011 and Ph.D. in Computer Science and Engineering from Nanyang Technological University, Singapore in April 2016. He has authored or co-authored over 50 papers in top-tier journals and conferences. He received the Nanyang Engineering Doctoral Scholarship (NEDS) Award in Singapore and the innovative and entrepreneurial Ph.D. Award of Jiangsu Province, China in 2011 and 2017, respectively. He is the recipient of the Second Prize for Scientific and Technological Progress awarded by the China Institute of Communications and he is the ChangKong Scholar of NUAA. His current research interests include telecommunication networking and cloud computing.

Jie Hao (Member IEEE) received her BS degree from Beijing University of Posts and Telecommunications, China, in 2007, and the Ph.D. degree from University of Chinese Academy of Sciences, China, in 2014. From 2014 to 2015, she has worked as post-doctoral research fellow in the School of Computer Engineering, Nanyang Technological University, Singapore. She is currently an Associate Professor at College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China. Her research interests are wireless sensing, wireless communication and etc.

Qiang Wu (Member, IEEE) is currently a Professor with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China. Before that, he worked as a general engineer of ZTE Central Research Institute, and received his Ph.D. degree from the College of Computer Science and Technology, Zhejiang University, Hangzhou, China. Prof. Wu is a Fellow of the China Institute of Communications, an Executive Member of Service Computing Technical Committee of CCF, and a Member of the Technical Committee of National Engineering Laboratory for Mobile Internet System and Application Security. His main research fields include future networks, industrial Internet, cyber security, and space-earth integration networks. The Chinese government honored him with the Second-Class National Science and Technology Progress Award in 2009 and the Second-Class National Technology Innovation Award in 2014. He has more than 100 authorized patents, of which more than 20 have corresponding relations with international standards.

Ping Wang (Fellow, IEEE) is a Professor at the Department of Electrical Engineering and Computer Science, York University, and a Tier 2 York Research Chair. Prior to that, she was with Nanyang Technological University, Singapore, from 2008 to 2018. Her research interests are mainly in radio resource allocation, network design, performance analysis and optimization for wireless communication networks, mobile cloud computing and the Internet of Things. Her scholarly works have been widely disseminated through top-ranked IEEE journals/conferences and received four Best Paper Awards from IEEE prestigious conferences. Her publications received 28,900+ citations with H-index of 77 (Google Scholar). She is a Fellow of IEEE and a Distinguished Lecturer of the IEEE Vehicular Technology Society.
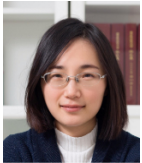
Dusit Niyato (Fellow, IEEE) received a Ph.D. degree in Electrical and Computer Engineering from the University of Manitoba, Canada, in 2008. He is currently a Professor in the School of Computer Science and Engineering at Nanyang Technological University, Singapore. He has published more than 400 technical papers in the area of wireless and mobile computing. He has won the Best Young Researcher Award of the Asia/Pacific chapter of the IEEE Communications Society and the 2011 IEEE Communications Society Fred W. Ellersick Prize Paper Award. Currently, he is serving as a senior editor of IEEE Wireless Communications Letters, an area editor of IEEE Transactions on Wireless Communications, an area editor of IEEE Communications Surveys and Tutorials, an editor of IEEE Transactions on Communications, and an associate editor of IEEE Transactions on Mobile Computing, IEEE Transactions on Vehicular Technology, and IEEE Transactions on Cognitive Communications and Networking. He was a Distinguished Lecturer of the IEEE Communications Society in 2016-2017. He has been named as a highly cited researcher in computer science. He is a Fellow of IEEE.