Multi-Type Attention for Solving Multi-Depot Vehicle Routing Problems

Jinqi Li, Bing Tian Dai, Yunyun Niu, Jianhua Xiao, and Yaoxin Wu

Abstract—**In recent years, there has been a growing trend towards using deep reinforcement learning (DRL) to solve the NP-hard vehicle routing problems (VRPs). While much success has been achieved, most of the previous studies solely focused on single-depot VRPs, which became less effective in handling more practical scenarios, such as multi-depot VRPs. Although there are many preprocessing measures, such as natural decomposition, those scenarios are still more challenging to optimize. To resolve this issue, we propose the multi-depot multi-type attention (MD-MTA) to solve the multi-depot VRP (MDVRP) and multi-depot open VRP (MDOVRP), respectively. We design a multi-type attention in the network to combine different types of embeddings and the state of the environment at each step, so as to accurately select the next node to visit and construct the route. We introduce a depot rotation augmentation to enhance solution decoding. Results show that it performs favorably against various representative traditional baselines and DRL-based baselines.**

Index Terms—**Deep reinforcement learning, learning to optimize, multi-depot vehicle routing problem, multi-depot open vehicle routing problem, attention mechanism, transformer model.**

I. Introduction

THE vehicle routing problem (VRP) [1] is a classic yet crucial combinatorial optimization (CO) problem for transportation and logistics. In this paper, we study two prominent variants, namely, the multi-depot VRP (MDVRP) [2] and the multi-depot open VRP (MDOVRP) [3]. The objectives of the two MDVRPs are to optimize the routes of vehicles, each of which starts from a depot. However, a discrepancy between MDVRP and MDOVRP lies in the returning rules, i.e., in MDVRP, vehicles are forced to return to their original departure depot, while in MDOVRP, vehicles will be dismissed after serving the last customer. One of the typical use cases of MDVRP and MDOVRP would be that a large, self-operated e-commerce enterprise intends to utilize the fleets it owns to deliver goods to a certain area, where the goods are stored in multiple distributed warehouses (a.k.a., depots).

Manuscript received 11 November 2023; revised 24 March 2024 and 4 June 2024; accepted 6 June 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 62172373, Grant 61872325, and Grant 62072258; and in part by the Fundamental Research Funds for the Central Universities under Grant 2652019028. The Associate Editor for this article was A.-T. Nguyen. *(Corresponding author: Yunyun Niu.)*

Jinqi Li and Yunyun Niu are with the School of Information Engineering, China University of Geosciences (Beijing), Beijing 100083, China (e-mail: jinqili@email.cugb.edu.cn; yniu@cugb.edu.cn).

Bing Tian Dai is with the School of Computing and Information Systems, Singapore Management University, Singapore 178902 (e-mail: btdai@smu.edu.sg).

Jianhua Xiao is with the Research Center of Logistics, Nankai University, Tianjin 300071, China (e-mail: jhxiao@nankai.edu.cn).

Yaoxin Wu is with the Department of Information Systems, Faculty of Industrial Engineering and Innovation Sciences, Eindhoven University of Technology, 5612 AZ Eindhoven, The Netherlands (e-mail: wyxacc@hotmail.com).

Digital Object Identifier 10.1109/TITS.2024.3413077

Traditional methods for solving MDVRPs include exact and heuristic algorithms. Exact algorithms usually suffer from high computational costs, especially on a large scale. Heuristic algorithms are able to solve larger instances, but the cost still increases considerably as the problem scales up [4]. Besides, the effectiveness of heuristic algorithms is highly contingent upon domain-specific knowledge, and the knowledge cannot be directly applied to different problems [5].

To alleviate the reliance on hand-engineering efforts, deep learning (DL) [6] and reinforcement learning (RL) [7] have been exploited. Deep reinforcement learning (DRL), featuring a fusion of DL and RL, utilizes RL to automatically learn a DL-based policy for decision-making. In the early phase, DRL methods were primarily investigated for decision-making in games [7]. Since the NP-hard VRPs can also be cast as a decision-making problem, it attracts many researchers to harness DRL to search for VRP solutions [8], [9], [10], [11], [12]. A widely recognized representative is the attention model (AM) [8], which is tailored from the Transformer architecture [6] and REINFORCE algorithm [13]. A salient advantage of DRL over traditional heuristics is its low computational cost.

Despite the success, previous DRL-based methods are still lacking in considering the characteristics of multiple depots, rendering them less effective for solving MDVRPs. To bridge this gap, we propose **M**ulti-**D**epot with **M**ulti-**T**ype **A**ttention mechanism, i.e., MD-MTA. We design multi-type attention to aggregate different types of embeddings to select the node at each step. We introduce a depot rotation augmentation to enhance the decoding. Results show that MD-MTA performed favorably against the baselines.

The remainder of this paper is as follows: Section II reviews the traditional methods for solving MDVRPs and DRL methods for solving VRPs; Section III introduces the formulations for MDVRP and MDOVRP; Section IV elaborates on MD-MTA; Section V displays experimental results; and Section VI concludes this paper and envisions future works.

II. Related Works

In this section, we review the classical methods for solving MDVRPs and DRL-based methods for solving VRPs.

A. Traditional Methods for Solving MDVRPs

MDVRP first solved in [2], where a tabu search algorithm was developed and evaluated on the benchmark instances. The relationship between depots and customers is many-to-many, where the pre-assignment of the customers is discussed in [14]. Liu et al. [15] proposed a hybrid genetic algorithm (HGA) for solving MDOVRP and MDVRP. Soto et al. [16] proposed a multiple neighborhood search coupled with a tabu search (MNS-TS) algorithm for solving MDOVRP, and settled a unified view of ejection chains to generate the neighborhood of MNS-TS. Wang et al. Sadati et al. [17] presented a variable tabu neighborhood search algorithm (VTNS) to solve MDVRP, MDVRP with time window (MDVRPTW), and MDOVRP, which improved the intensification phase with a granular local search and improved the diversification phase with a tabu shaking mechanism. Solution time of traditional methods is the most challenging issue.

B. DRL-Based Methods for Solving VRPs

Bello et al. [18] exploited the Pointer Network (PtrNet) [19] coupled with a sequence-to-sequence (seq2seq) [20] attention mechanism and actor-critic algorithm [21]. It achieved promising solutions to the travelling salesman problem (TSP). Kool et al. [8] proposed an attention model (AM) featuring a self-attention mechanism [6] and trained it with the REINFORCE algorithm [13]. AM achieved a higher solution quality than that of PtrNet. Kwon et al. [9] proposed the POMO, which samples multiple trajectories as a baseline to affect the training result. POMO can produce nearly optimal results for TSP and CVRP, yet it still falls short for solving MDVRP and MDOVRP. TSP and CVRP have only one depot; one overall embedding in plain self-attention can include all information, which is different from MDVRPs. Zou et al. [22] tailored a transformer model with attention-to-attention mechanism (TAOA) for solving the low-carbon MDVRP. Zhang et al. [23] proposed a reinforcement learning method based on graph attention network (GAT-RL) to solve MDVRP with time windows that consider the spatial-temporal correlation. While being able to solve specific variants of MDVRP, most of them omitted to consider the characteristics of multi-depots while proposing a general DRL-based method.

III. Problem Formulation

This section presents the formulation of MDVRP and MDOVRP based on mathematical programming and the Markov Decision Process (MDP), respectively.

A. Preliminary

In MDVRP and MDOVRP, one or more fully loaded vehicles depart from each of the depots to sequentially visit and serve the customers with different demands at different locations. Each customer can be visited only once. The vehicles in MDVRP will return to the original depot after serving the dedicated customers. Differently, the vehicles in MDOVRP do not return to the depot. The detail of mathematical formulation will be introduced in appendix A.

B. MDP Formulation

The representation for MDVRP and MDOVRP could also be cast as a Markov Decision Process (MDP) [24] and then solved using reinforcement learning (RL). Particularly, the MDP is defined by a 4-tuple $M = \{S, A, T, R\}$, i.e., S denotes the state space, A denotes the action space, T denotes the transition rule, and R denotes the reward.

State: Each state $S^t = (M^t, X^t)$ $(S^t \in S)$ comprises two components. The first component relates to the vehicle state, i.e., $M^t = \{C^t, L^t, O^t\}$, where C^t denotes the remaining capacity of the vehicle at step t, L^t denotes the total route length that the fleet of vehicles have travelled till step t, and $O^t = \{x_i^0, x_j^1, \ldots, x_k^t\}$ represents the travelled route, i.e., the array of visited nodes, where x_k^t indicates that node x_k has been visited at step t. The number of vehicles in MDVRP is set the same as the number of depots. However, the vehicles in MDOVRP are essentially unable to return to the depots, so the number of vehicles is not limited. Upon departure from a depot, the initial vehicle state is set to $M^0 = \{C_{max}, 0, \{x_i^0\}\}$ where C_{max} is the fixed maximum capacity of the vehicle. The second component relates to the node state, i.e., $X^t = \{x_0^t, x_1^t, x_2^t, \ \ldots, x_n^t\} = \{(G_0, D_0), \ (G_1, D_1), \ (G_2, D_2), \ldots, (G_n, D_n)\}$, where G_i denotes the two-dimensional coordinates as the location of node x_i, and D_i^t denotes the demand of node x_i at step t. In MDOVRP, the distance from any customer to a depot is set to 0, as indicated in Eq. (32). Each customer can be visited only once.

Action: Each action is defined as a visit to a node. Specifically, the action A^t $(A^t \in A)$ is represented as x_i^t, indicating that node x_i has been selected to visit at step t.

Transition: The transition rule T evolves the preceding state S^t to the current state S^{t+1} based on the last action $A^t = x_i^t$ and the current action $A^{t+1} = x_j^{t+1}$. The vehicle state M^t is updated to $M^{t+1} = \{C^{t+1}, L^{t+1}, O^{t+1}\}$ as follows,

$$C^{t+1} = \begin{cases} C^t - D_j^t, & \text{if } x_j \text{ is not a depot} \\ C_{max}, & \text{if } x_j \text{is a depot} \end{cases} \tag{1}$$

$$L^{t+1} = L^t + Z(x_i, x_j), \tag{2}$$

$$O^{t+1} = [O^t, x_j^{t+1}], \tag{3}$$

where [,] is a concatenation operator. The node state X^t will be updated to $X^{t+1} = \{\ldots, (G_j, D_j^{t+1}), \ldots\}$ with $D_j^{t+1} = 0$.

Reward: The reward R relates to the total length L of the travelled route, which is represented as the negative L as follows,

$$R = -L^f, \tag{4}$$

where f indicates the index of the terminal state.

Policy: The goal of RL is to find a policy π_θ that starts the routing from the initial state S^0 and completes it at the terminal state S^f by governing the action selection at each step. Typically, the joint probability distribution of this process executed by this policy is expressed as follows,

$$p(S^f|S^0) = \prod_{t=0}^{f-1} \pi_\theta(A^t|S^t). \tag{5}$$

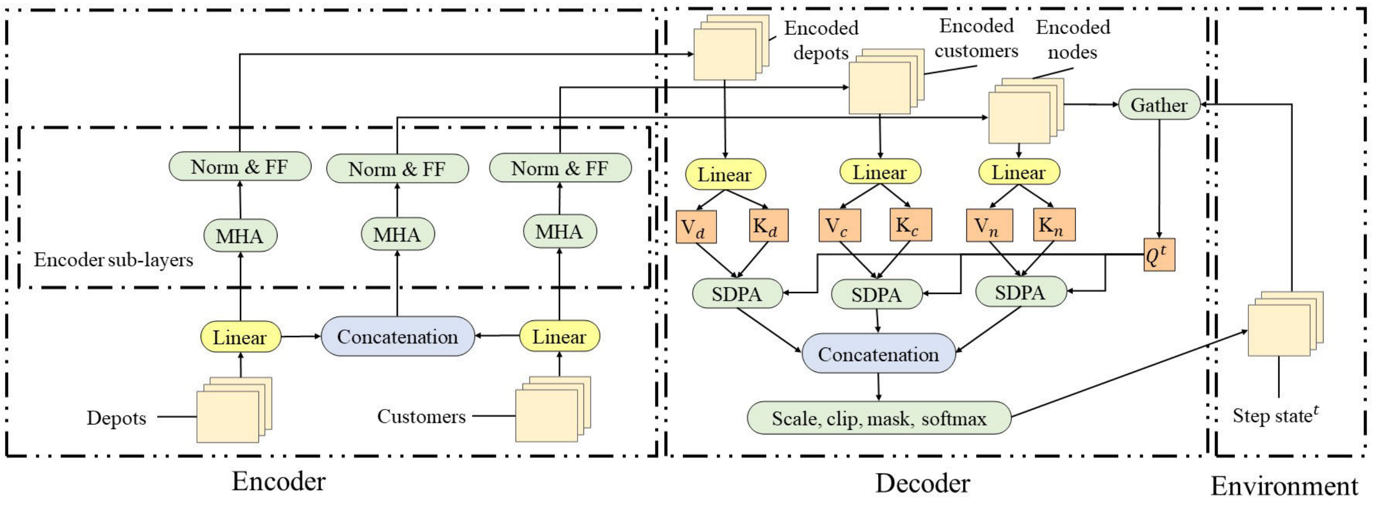


Fig. 1. The overall architecture of the MD-MTA network.

IV. Methodology

In this section, we introduce the proposed MD-MTA (Multi-Depot with Multi-Type Attention), including its architecture, training, and inference.

A. Architecture

Overview: As illustrated in Figure 1, the MD-MTA is primarily composed of an encoder and a decoder. The encoder first linearly projects the depots, customers, and overall nodes (both depots and customers) into three embeddings. Then the sub-layers iteratively perform multi-head attention (MHA), normalization, and feed-forward operations. The decoder leverages the three types of embeddings while interacting with the environment to decide the next node to visit.

Encoder: In previous DRL methods, nodes information is always encoded to one embedding. In this case, the DRL network cannot understand the difference between depots and customers because all nodes are included in one embedding. MD-MTA comprehensively yields three types of embeddings to learn the relationship between depots and customers. Note that MD-MTA is a one-stage method [14] for solving MDVRPs (without pre-dispatching), so the relationship between depots and customers is important yet challenging to capture. In MD-MTA, the depot and customer will be handled separately. As Figure 1 displays, the encoder takes the two-dimensional coordinates and demands of nodes as input and leverages two linear projections to yield the initial embeddings for depots E^d and customers E^c, respectively, which are also concatenated to form the initial embedding for the overall nodes E^n. This process could be expressed as follows,

$$E^d = I^d W_{E^d}, \tag{6}$$
$$E^c = I^c W_{E^c}, \tag{7}$$
$$E^n = \text{concat}(I^d, I^c) W_{E^n}, \tag{8}$$

where I^d represents depots, I^c represents customers, and W_{E^d}, W_{E^c}, and W_{E^n} are trainable parameters. The embeddings of depots E^d, customers E^c and overall nodes E^n are respectively processed through U ($U = 6$) sub-layers, which start with the MHA layer. There are Y ($Y = 16$) heads in the MHA mechanism, and the shape of query, key, and value is defined as $dim_Q = dim_E / Y$, where dim_E is the dimension of the embeddings. These components are used to compute their respective scaled dot-product attention (SDPA) H^e_{uy} [6]. Finally, all the SDPAs are concatenated together to form the MHA output H^e_u, which has the same shape as the input. This process could be expressed as follows,

$$Q^e_{uy} = E^e_u W_{Q^e_u}, \tag{9}$$
$$K^e_{uy} = E^e_u W_{K^e_u}, \tag{10}$$
$$V^e_{uy} = E^e_u W_{V^e_u}, \tag{11}$$
$$H^e_{uy} = \frac{Q^e_{uy} (K^e_{uy})^T}{\sqrt{dim_{embed}}} V^e_{uy}, \tag{12}$$
$$H^e_u = \text{concat}(H^e_{u1}, H^e_{u2}, \ldots, H^e_{uy}) W_{H^e_u}, \tag{13}$$

with $\forall u \in \{1, \ldots, U\}$, $\forall y \in \{1, \ldots, Y\}$, $\forall e \in \{n, d, c\}$, where E^e_u represents the input of the u-th sub-layer for type e embedding, i.e., E^e_1 is the initial embedding in the encoder; $W_{Q^e_u}$, $W_{K^e_u}$, $W_{V^e_u}$, $W_{H^e_u}$ are trainable parameters. The MHA output H^e_u then undergoes further processing through normalization and feed-forward operations as follows,

$$F^{e1}_u = \text{norm}(H^e_u + E^e_u), \tag{14}$$
$$F^{e2}_u = \text{relu}(W_{F^{e1}_u} F^{e1}_u) W_{F^{e2}_u}, \tag{15}$$
$$F^{e3}_u = \text{norm}(F^{e1}_u + F^{e2}_u), \tag{16}$$

with $\forall u \in \{1, \ldots, U\}$, $e \in \{n, d, c\}$, where F^{e1}_u denotes the output of the first normalization, F^{e2}_u denotes the feed-forward output, F^{e3}_u denotes the output of the second normalization, and $W_{F^{e1}_u}$ and $W_{F^{e2}_u}$ are trainable parameters. The output of the last sub-layer corresponds to the encoded embedding of depots E^d_z, customers E^c_z, and overall nodes E^n_z, where z indicates its presence in the decoder.

Decoder: The relationship between depots and customers in MDVRPs is many-to-many. For instance, a customer may be associated with any of those depots. Therefore, processing the nodes with sole overall embedding may not obtain high-quality solutions. To tackle this issue, a multi-type attention decoder is designed. As Figure 1 exhibits, the decoder has a similar MHA mechanism as in the encoder, which shares the same number of heads (Y) and the shape of query, key and value (dim_Q). The decoder yields the key of overall nodes K^n_{zy}, depots K^d_{zy}, and customers K^c_{zy}, as well as the value of overall nodes V^n_{zy}, customers V^c_{zy}, and depots V^d_{zy} at the initial state as follows,

$$K^e_{zy} = E^e_z W_{K^e_{zy}}, \tag{17}$$

$$V^e_{zy} = E^e_z W_{V^e_{zy}}, \tag{18}$$

with $\forall y \in \{1, \ldots, Y\}$, $\forall e \in \{n, d, c\}$, where $W_{K^e_{zy}}$ and $W_{V^e_{zy}}$ are trainable parameters. The query of the decoder is yielded by gathering the encoded embedding of overall nodes E^n_z and the current step state S^t, which can be expressed as follows,

$$Q^t_{zy} = \text{select}(E^n_z, S^t) W_{Q^t_{zy}}, \tag{19}$$

with $\forall y \in \{1, \ldots, Y\}$, where t is the index of step state and $W_{Q^t_{zy}}$ is a trainable parameter, select is an operator to extract the chosen embedding from the embedding of overall node according to the chosen node. For instance, an embedding has a shape of "batch size" * "trajectory size" * "embedding size", while the chosen node has a shape of "batch size" * "trajectory size". As a result, the chosen node is used as an index to extract the 1 * 1 * "embedding size" for updating the state. In MD-MTA, the decoder will extract $n-1$ nodes at the same time used to generate multiple trajectories; $n-1$ means that the decoder generates one trajectory for each candidate node in the initial state. The decoder then yields three types of SDPAs for each type of key and value in each head y, including overall-nodes-to-overall-nodes H^{nt}_{zy}, overall-nodes-to-depots H^{dt}_{zy} and overall-nodes-to-customers H^{ct}_{zy} for step t as follows,

$$H^{et}_{zy} = \frac{Q^t_{zy}(K^e_{zy})^T}{\sqrt{dim_{embed}}} V^e_{zy}, \tag{20}$$

with $\forall y \in \{1, \ldots, Y\}$, $\forall e \in \{n, d, c\}$. Then the SDPAs of respective types are aggregated into three MHAs, i.e., H^{nt}_z, H^{dt}_z and H^{ct}_z for step t, similar to the one in the encoder as follows,

$$H^{et}_z = \text{concat}(H^{et}_{z1}, H^{et}_{z2}, \ldots, H^{et}_{zY}) W_{H^{et}_z}, \tag{21}$$

with $\forall e \in \{n, d, c\}$, where $W_{H^{et}_z}$ is a trainable parameter. The MHA outputs will be summed together, and matrix-multiplied with the embedding of overall nodes E^n_z to attain the attention score H^t_{mm} as follows,

$$H^t_{mm} = (H^{nt}_z + H^{dt}_z + H^{ct}_z) E^n_z, \tag{22}$$

After scaling, clipping and masking, the attention score H^t is ready to be converted into a probability list P^t for node selection. Particularly, the process of deriving H^t mentioned above could be expressed as follows,

$$H^t = \text{CLIP} \times \tanh(H^t_{mm}) + \text{mask}, \tag{23}$$

where CLIP is a constant hyper-parameter and tanh() is an activation function. Regarding the masking rules, at any step, customers with demand greater than the current remaining capacity of the vehicle are masked. In addition, once a vehicle is instantiated with a depot, all other depots are masked to this vehicle. The MD-MTA dynamically masks all invalid nodes according to those rules at each step. The masked nodes are allowed to be released again when they do not breach those rules. Then the probability P^t_x for selecting the node x is derived by applying the SoftMax function as follows,

$$P^t_x = \text{softmax}(H^t_x) = \frac{\exp(H^t_x)}{\sum_{k=1}^{n} \exp(H^t_k)}, \tag{24}$$

with $\forall x \in \{1, \ldots, n\}$, where H^t_x is the unnormalized probability in the list H^t for selecting the node x, and P^t is the list of P^t_x that is used by the MD-MTA to select the node for route construction at each step.

B. Inference

In this section, we introduce the details of inference methods, including instance augmentations (i.e., the coordinate rotation and the depot rotation) and rollout strategies (i.e., the greedy and the sampling strategies).

Coordinate Rotation Augmentation: Instance augmentation is employed to improve the search diversity during inference. The two-dimensional coordinates of the problem instances can be rotated in VRPs similar to how images are rotated in the field of computer vision. Following Kwon et al. [9], 7 rotations are considered based on the original $(x,\ y)$, including $(x,\ 1-y)$, $(1-x,\ y)$, $(1-x,\ 1-y)$, $(y,\ x)$, $(y,\ 1-x)$, $(1-y,\ x)$, $(1-y,\ 1-x)$.

Depot Rotation Augmentation: Given the characteristics of multiple depots, the selection of the initial depot is crucial. In previous work, DRL methods often depended on simple schemes, such as randomly choosing a depot or sampling all depots, which resulted in the inferiority of the final solution. In this regard, we propose a depot rotation scheme to further enrich the instance augmentation. Specifically, the rotation is prescribed as $(1,\ 0,\ 2, \ldots,\ d-1)$, $(2,\ 1,\ 0, \ldots,\ d-1)$, $\ldots$, $(d-1,\ 1,\ 2, \ldots,\ 0)$, based on the original one $(0,\ 1,\ 2, \ldots,\ d-1)$, where d denotes the number of depots. Hence, $d-1$ new augmented instances are considered. Different sequence of the node will be generated as different embeddings by same policy network. Thus, depot rotation augmentation can obtain different solution under the same policy and essentially same instances.

Deployment of Instance Augmentation: The instance augmentation is executed by applying the above two schemes. To keep a reasonable number of instances, we only apply the depot rotation to the original $(x,\ y)$. For a more succinct representation, we use MDVRP-100C3D and MDVRP-100C5D represent the MDVRP instances with 100 customers and 3 or 5 depots, respectively. A total of 10 (1 original + 7 coordinate rotation + 2 depot rotation) and 12 (1 original + 7 coordinate rotation + 4 depot rotation) instance augmentations for MDVRPs-3D and MDVRPs-5D, respectively. We will use 'aug.10' and 'aug.12' to denote the DRL-based method featuring the two augmentation schemes on an MDVRPs-3D and an MDVRPs-5D instance, respectively. We also denote 'aug.8' to indicate a DRL-based method featuring only coordinate rotation augmentation. Those augmented instances are different

Algorithm 1 MD-MTA Network Inference

```
Input: evaluation set $h$, the number of trajectories $m$, policy
    $\pi_\theta$, the number of augmentations $g$ (i.e., 7+d)
Output: route
1: $h' \leftarrow \{h_1, h_2, \dots, h_g\} \leftarrow$ instance augment (h)
2: for $i = 1$ to $g$ do
3:     for $j = 1$ to $m$ do
4:         $\tau_j^i \leftarrow$ rollout $(h', \pi_\theta)$
5:     end for
6: end for
7: route = argmax $R(\tau_j^i)$ $\forall i \in \{1, \dots, g\}$ $\forall j \in \{1, \dots, m\}$
```

from the original ones in the view of the policy network, while they are essentially regarded as the same ones in the view of humans, and they exactly are. Thus, the optimal solution may be obtained from the augmented instances. We retrieve the best solution found across all augments.

Greedy Strategy selects the node with the highest probability to visit, which means that the same results will be obtained for each instance even if the inference is repeated multiple times. For a more concise representation, we use 'g.' to denote the DRL-based method featuring the greedy strategy for rollout.

Sampling Strategy samples a node according to the probabilities, which means that there is a chance to obtain different solutions. For a more concise representation, we use 's.' to denote the DRL-based method featuring the sampling strategy for rollout, and we use 's.200' to indicate the case of sampling 200 solutions per (augmented) instance.

Deployment of Rollout Strategy. The inference of MD-MTA is summarised in Algorithm 1.

V. Experiments Result

We conducted a series of experiments to verify the following assumptions:

- MD-MTA outperforms baselines on synthetic datasets (see Tables III, IV, V, and VI).
- MD-MTA showcases better generalization performance on benchmark datasets (see Tables VII and VIII).
- Each design contributes to performance, respectively (see Table IX).

A. Experiments Setup

In this section, we introduce the experiment setup. To highlight the efficiency, we set a time limit of 60 seconds, ensuring that the experiments are feasible in real life. We use an 8-core AMD R7 CPU to evaluate the performance of CPU-based methods and a single RTX4090 GPU to evaluate that of DRL-based solvers. Our code for MD-MTA is available.[1]

Hyper-Parameters: The dimension of embedding is set as $dim_E = 256$. The number of encoder sub-layers is set as $U = 6$. The dimensions of query, key, and value are set as $dim_Q = 16$. The number of heads in the MHA mechanism is set as $Y = 16$. The size of the tan clip is set as CLIP = 10, and the dimensions of the feed-forward layer are set as $dim_F = 512$. In all training, the number of epochs is 1,000, and each epoch consists of 10,000 synthetic problem instances.

Baselines: We compare our MD-MTA with many different baselines: **POMO** [9], a state-of-the-art DRL-based construction method. **TAOA** [22], an improved transformer model for solving the low-carbon MDVRP. **GAT-RL** a reinforcement learning method based on graph attention network to solve MDVRP with time windows. **OR-Tools** [25], a heuristic solver widely used for CO problems. We tested the performance of OR-Tools on both MDVRP and MDOVRP. We set the parameters in accordance with the official recommendations. **Gurobi** [26], the fastest exact solver. We didn't compare other exact methods since their aims are to search for the optimal solution regardless the computational cost, which is contrary to our goals, i.e., obtaining decent solutions through negligible time spent. So, we only use Gurobi to reflect the effect of exact methods. **HGA** [15], a hybrid genetic algorithm **MNS-TS** [16], a multiple neighborhood search coupled with a tabu search algorithm. **VTNS** [17], a variable tabu neighborhood search algorithm. We set the parameters of the traditional baselines following their original papers.

Training: We trained each DRL-based method on the synthetic dataset with three different numbers of customers $\{20, 50, 100\}$ and two different numbers of depots $\{3, 5\}$, respectively. We maintain the capacity of vehicles constant at 50 and the demand of customers randomly drawn from the set $\{1, 2, 3, \dots, 10\}$. We use the same Adam Optimizer [27] during training. The training time varied with the number of nodes. Table I and Table II summarise the training time for MDVRP and MDOVRP separately. Notice that MDOVRP requires less time compared to MDVRP due to the shorter solution lengths in MDOVRP.

B. Performance Comparisons on Synthetic Datasets

We first compare the performance on synthetic datasets. The details of setups, baselines, results on MDVRP, and results on MDOVRP are presented and discussed in this section.

Setup: We initiate our evaluation on synthetic datasets, where the settings of the datasets are the same as the training ones.[2] All methods were evaluated on the same sets of 2,000 synthetic test instances for each problem size.

Results on MDVRP: Table III and Table IV report the mean objective values as 'Obj.', mean gaps as 'Gap',[3] and total solving time as Time(m) on MDVRP. We make an effort to test the Gurobi without a time limit, but it is not feasible to obtain each solution with an acceptable time, i.e., no solution may be found even over 10 minutes on instances of 50C or 100C,

[1]https://github.com/Good9T/MD_MTA

[2]The generation of synthetic instances is display as follows: All coordinates are real numbers and follow the uniform distribution in the range of [0,1], whether the X coordinates or the Y coordinates; all demands are positive integers and follow the uniform distribution in the range of [1, 10].

[3]In Table III, Table IV, Table V, and Table VI, the gaps with boldface indicate that the corresponding method is the best among all methods, and the gaps with * indicate that the corresponding method is the best among DRL-based methods.

TABLE I
THE TRAINING TIME(H) OF DRL-BASED METHODS ON MDVRP

Method	MDVRP					
	20C3D	50C3D	100C3D	20C5D	50C5D	100C5D
POMO	2.47	5.13	12.04	2.59	5.41	12.85
MD-MTA	3.64	7.81	19.24	3.87	8.23	20.98

TABLE II
THE TRAINING TIME(H) OF DRL-BASED METHODS ON MDOVRP

Method	MDOVRP					
	20C3D	50C3D	100C3D	20C5D	50C5D	100C5D
POMO	1.98	4.59	10.53	2.12	4.44	11.18
MD-MTA	3.36	6.97	17.89	3.82	7.18	19.43

TABLE III
THE MDVRP-3D RESULTS ON SYNTHETIC DATASETS

Method	MDVRP-20C3D			MDVRP-50C3D			MDVRP-100C3D		
	Obj.	Gap	Time(m)	Obj.	Gap	Time(m)	Obj.	Gap	Time(m)
Gurobi	4.13	**0.00%**	2.97	-	-	-	-	-	-
Gurobi(60s)	4.13	0.00%	2.97	7.39	0.40%	857.46	12.38	3.14%	2001.62
OR-Tools(60s)	4.43	7.31%	1982.36	7.86	6.79%	2000.96	12.50	4.11%	2001.23
HGA(60s)	4.15	0.62%	83.46	7.39	0.40%	347.36	12.40	3.31%	2001.31
VTNS(60s)	4.14	0.37%	320.95	7.41	0.69%	1032.72	12.31	2.51%	2000.97
TAOA(aug.8, g.)	4.20	1.76%	0.03	7.57	2.89%	0.09	12.41	3.36%	0.52
TAOA(aug.8, s.400)	4.16	0.79%	10.86	7.46	1.40%	34.12	12.23	1.86%	162.74
GAT-RL(aug.8, g.)	4.19	1.52%	0.04	7.53	2.35%	0.11	12.36	2.94%	0.56
GAT-RL(aug.8, s.400)	4.15	0.55%	12.32	7.43	0.99%	36.52	12.19	1.53%	153.48
POMO (aug.8, g.)	4.18	1.19%	0.03	7.47	1.57%	0.09	12.26	2.12%	0.47
POMO (aug.8, s.400)	4.14	0.36%	10.19	7.39	0.51%	33.92	12.15	1.17%	151.84
MD-MTA (aug.10, g.)	4.16	0.78%	0.07	7.42	0.90%	0.20	12.17	1.34%	0.80
MD-MTA (aug.10, s.200)	4.13	*0.02%	10.28	7.36	***0.00%**	32.27	12.01	***0.00%**	156.18

TABLE IV
THE MDVRP-5D RESULTS ON SYNTHETIC DATASETS

Method	MDVRP-20C5D			MDVRP-50C5D			MDVRP-100C5D		
	Obj.	Gap	Time(m)	Obj.	Gap	Time(m)	Obj.	Gap	Time(m)
Gurobi	3.78	**0.00%**	3.22	-	-	-	-	-	-
Gurobi(60s)	3.78	0.00%	3.22	6.60	0.10%	895.18	10.79	1.57%	2000.95
OR-Tools(60s)	4.08	7.98%	2000.15	7.01	6.37%	2000.25	11.00	3.57%	2001.31
HGA(60s)	3.86	1.96%	89.59	6.72	1.95%	369.81	10.91	2.76%	2000.58
VTNS(60s)	3.79	0.21%	339.27	6.62	0.44%	1123.52	10.80	1.73%	2001.23
TAOA(aug.8, g.)	3.86	2.05%	0.04	6.77	2.69%	0.12	10.98	3.40%	0.65
TAOA(aug.8, s.500)	3.84	1.53%	15.48	6.72	1.93%	42.12	10.82	1.89%	209.69
GAT-RL(aug.8, g.)	3.85	1.79%	0.05	6.73	2.09%	0.14	10.93	2.93%	0.69
GAT-RL(aug.8, s.500)	3.82	1.00%	16.26	6.67	1.18%	43.87	10.79	1.61%	216.74
POMO (aug.8, g.)	3.83	1.17%	0.04	6.70	1.61%	0.11	10.90	2.62%	0.62
POMO (aug.8, s.500)	3.80	0.45%	15.23	6.63	0.56%	40.86	10.75	1.19%	204.68
MD-MTA (aug.12, g.)	3.82	0.89%	0.10	6.65	0.90%	0.25	10.74	1.09%	1.12
MD-MTA (aug.12, s.200)	3.79	*0.10%	15.91	6.59	***0.00%**	40.21	10.62	***0.00%**	208.68

so we only record the results without time limits on 20C. As reported, DRL-based methods show their overwhelming advantage in time spent. We compare the MD-MTA (g.) and the POMO (g.) to testify that MD-MTA outperforms POMO when both of them are using the greedy strategy; we compare the MD-MTA (s.200) to the POMO (s.200) to testify that the

TABLE V
THE RESULT OF SYNTHETIC DATASET ON MDOVRP-3D

Method	MDOVRP-20C3D			MDOVRP-50C3D			MDOVRP-100C3D		
	Obj.	Gap	Time(m)	Obj.	Gap	Time(m)	Obj.	Gap	Time(m)
Gurobi	2.95	**0.00%**	2.28	-	-	-	-	-	-
Gurobi(60s)	2.95	**0.00%**	2.28	5.15	0.20%	793.24	8.31	2.30%	2000.84
OR-Tools(60s)	3.18	7.52%	1238.35	5.43	5.60%	2001.25	8.47	4.21%	2001.65
HGA(60s)	2.98	0.82%	78.62	5.18	0.82%	325.35	8.41	3.55%	2001.28
VTNS(60s)	2.96	0.28%	159.63	5.18	0.75%	487.84	8.24	1.47%	1336.86
TAOA(aug.8, g.)	3.04	2.90%	0.03	5.29	2.92%	0.09	8.42	3.62%	0.51
TAOA(aug.8, s.400)	3.01	1.89%	10.86	5.22	1.56%	33.52	8.26	1.66%	154.51
GAT-RL(aug.8, g.)	3.02	2.23%	0.04	5.26	2.34%	0.11	8.37	3.01%	0.53
GAT-RL(aug.8, s.400)	3.00	1.55%	12.32	5.20	1.17%	35.73	8.23	1.29%	151.61
POMO (aug.8, g.)	2.99	1.19%	0.03	5.26	2.34%	0.08	8.29	2.04%	0.44
POMO (aug.8, s.400)	2.97	0.47%	9.18	5.18	0.75%	30.12	8.22	1.12%	142.14
MD-MTA (aug.10, g.)	2.99	1.05%	0.07	5.21	1.28%	0.19	8.20	0.96%	0.78
MD-MTA (aug.10, s.200)	2.96	*0.05%	9.88	5.14	***0.00%**	28.60	8.13	***0.00%**	147.24

MD-MTA outperforms POMO while both of them sample the same number of solutions; and we compare the MD-MTA (s.200) to the POMO (s.400) and to the POMO (s.500) to testify that the MD-MTA outperforms POMO while using a similar time on MDVRP-100C. Compared to traditional baselines, our MD-MTA (g.) outperforms the OR-Tools in all cases and outperforms all other traditional baselines on the hardest 100C problems. With prolonged run time, our MD-MTA (s.200) consistently outperforms all traditional baselines in all cases, except slightly inferior to the Gurobi on MDVRP-20C instances, as the exact solver is more suitable to solve small-scale problems but unable to solve larger ones. With the increase in problem size, the advantage between the MD-MTA and other baselines also increases. For instance, the mean gap in Table III between MD-MTA (s.200) and POMO (s.400) increases from 0.34% to 0.51% and from 0.51% to 0.75% as the number of customers increases from 20 to 50 and from 50 to 100. Similar observations can be found when comparing MD-MTA to Gurobi, HGA, and VTNS. These results demonstrate that MD-MTA can yield a high-quality solution with a small computational cost.

Results on MDOVRP: On MDOVRP, the baselines we compare to are similar to those on MDVRP. Table V and Table VI report the mean objective values as Obj., the mean gaps as Gap and the total solving time as Time(m) on MDOVRP. Conclusions similar to those in MDVRP can be drawn.

C. Generalization on Benchmark Datasets

We tested the generalization performance on benchmark datasets. Following baselines in Section V-B, we use POMO, GAT-RL, and TAOA as baselines. We chose the benchmark dataset following various previous works [2], [15], [16], [17]. We use names {p01, p02, p03, p04, p05, p06, p07, p08, p09, p10, p11} to represent each instance in the benchmark dataset sourced from Renaud et al. [2], where {p01, p02, p03, p04, p05, p06, p07} is designed by Christofides and Eilon [1], and {p08, p09, p10, p11} is designed by Gillett and Miller [28].

Setup: All the DRL-based methods were trained on a 100C3D synthetic dataset and used the same settings. We compare our MD-MTA (s.200) to POMO (s.400), GAT-RL (s.400), and TAOA (s.400) to test the relative generalization performance while using a comparable total solving time. All methods evaluate each instance 10 times, and the objective value is their average value. We also compare the results to the best-known solutions (BKS) to indicate the absolute performance of DRL-based methods. We note that those benchmark instances differ a lot from the ones used for training. In Table VII and VIII, the settings of instances reported include the number of depots as 'Dep.', the number of customers as 'Cus.', and the best-known solutions (BKS) references from Sadati et al. [17] as 'BKS', while the results reported include objective values along with gaps to the BKS.

Results on MDVRP: Table VII reports the generalization results on the MDVRP, including the best objective values of 10 solution as 'B.O.', the gaps[4] from 'B.O.' to the BKS as 'B.G.', the average objective values of 10 solution as 'A.O.', and the gaps from 'A.O.' to the BKS as 'A.G.'. Our MD-MTA outperforms other DRL-based baselines consistently in generalization experiments on all 11 MDVRP instances. It is worth noting that the instances {p08, p09, p10, p11} contain an exquisitely different number of customers (249) than the training instances (100), so DRL-based methods would get worse performance than other instances. The generalization results of MDVRP show that the largest 'A.G.' of our MD-MTA to the BKS is 7.63% on p11, while the 'A.G.' to the BKS of other DRL-based baselines on p11 is at least 10.34%. Besides, the average 'A.G.' of our MD-MTA to the BKS is 2.60%, while the average 'A.G.' to the BKS of other DRL-based baselines is at least 4.70%. Our MD-MTA is able to achieve the BKS except on those instances with a large different number of customers, and the largest 'B.G.' to the BKS is 4.15%, while the 'B.G.' to the BKS of other DRL-based baselines on p11

[4]In TableVII and Table VIII, the gaps with boldface indicates that the corresponding method is the best among all methods.

TABLE VI
THE RESULT OF SYNTHETIC DATASET ON MDOVRP-5D

Method	MDOVRP-20C5D			MDOVRP-50C5D			MDOVRP-100C5D		
	Obj.	Gap	Time(m)	Obj.	Gap	Time(m)	Obj.	Gap	Time(m)
Gurobi	2.68	**0.00%**	2.39	-	-	-	-	-	-
Gurobi(60s)	2.68	**0.00%**	2.39	4.68	0.13%	810.23	7.49	1.57%	2000.78
OR-Tools(60s)	2.86	6.67%	1267.21	4.89	4.67%	2000.48	7.64	3.49%	2001.23
HGA(60s)	2.72	1.13%	82.36	4.71	0.80%	352.91	7.61	3.14%	2001.35
VTNS(60s)	2.70	0.69%	173.15	4.70	0.50%	523.64	7.50	1.64%	1347.93
TAOA(aug.8, g.)	2.76	2.80%	0.04	4.82	3.10%	0.11	7.62	3.27%	0.54
TAOA(aug.8, s.500)	2.72	1.31%	5.81	4.74	1.39%	18.23	7.54	2.19%	203.47
GAT-RL(aug.8, g.)	2.74	2.05%	0.05	4.79	2.46%	0.12	7.59	2.87%	0.59
GAT-RL(aug.8, s.500)	2.71	0.93%	5.96	4.73	1.18%	18.96	7.52	1.92%	206.12
POMO (aug.8, g.)	2.73	1.56%	0.04	4.76	1.81%	0.10	7.53	2.00%	0.53
POMO (aug.8, s.500)	2.70	0.62%	13.21	4.71	0.66%	38.51	7.47	1.22%	197.34
MD-MTA (aug.12, g.)	2.72	1.13%	0.09	4.74	1.44%	0.23	7.48	1.40%	1.03
MD-MTA (aug.12, s.200)	2.69	*0.08%	12.86	4.67	***0.00%**	37.40	7.38	***0.00%**	198.60

TABLE VII
THE RESULT OF GENERALIZATION EXPERIMENTS ON MDVRP

Ins.	Dep.	Cus.	BKS	MD-MTA (aug.9-12,s.200)				POMO (aug.8, s.400)				GAT-RL (aug.8, s.400)				TAOA (aug.8, s.400)			
				B.O.	B.G.	A.O.	A.G.	B.O.	B.G.	A.O.	A.G.	B.O.	B.G.	A.O.	A.G.	B.O.	B.G.	A.O.	A.G.
p01	4	50	577	577	**0.00%**	581	**0.68%**	581	0.69%	589	2.06%	584	1.30%	591	2.52%	589	2.03%	594	2.89%
p02	4	50	474	474	**0.00%**	475	**0.24%**	477	0.63%	478	0.94%	482	1.84%	486	2.69%	486	2.68%	490	3.53%
p03	5	75	641	641	**0.00%**	650	**1.33%**	648	1.09%	661	3.09%	652	1.73%	663	3.44%	665	3.66%	680	6.00%
p04	2	100	1001	1001	**0.00%**	1013	**1.17%**	1011	1.00%	1020	1.86%	1017	1.56%	1040	3.85%	1042	4.12%	1056	5.52%
p05	2	100	750	750	**0.00%**	755	**0.66%**	760	1.33%	771	2.85%	767	2.32%	772	2.98%	773	3.11%	781	4.18%
p06	3	100	877	877	**0.00%**	879	**0.33%**	883	0.68%	894	1.97%	887	1.24%	897	2.38%	894	1.96%	900	2.65%
p07	4	100	882	882	**0.00%**	885	**0.31%**	897	1.70%	911	3.29%	905	2.59%	913	3.49%	909	3.03%	927	5.07%
p08	2	249	4370	4467	**2.22%**	4599	**5.25%**	4587	4.97%	4721	8.02%	4661	6.67%	4796	9.76%	4675	6.97%	4770	9.15%
p09	3	249	3859	3919	**1.55%**	4026	**4.33%**	3985	3.27%	4124	6.88%	4064	5.33%	4196	8.75%	4037	4.63%	4186	8.49%
p10	4	249	3630	3770	**3.86%**	3874	**6.72%**	3820	5.23%	4008	10.43%	3934	8.39%	4079	12.39%	4013	10.56%	4105	13.09%
p11	5	249	3545	3692	**4.15%**	3816	**7.63%**	3723	5.02%	3912	10.34%	3903	10.09%	4053	14.32%	3996	12.71%	4040	13.95%
average			1873	1914	**1.07%**	1959	**2.60%**	1943	2.33%	2008	4.70%	1987	3.91%	2044	6.05%	2007	5.04%	2048	6.77%

TABLE VIII
THE RESULT OF GENERALIZATION EXPERIMENTS ON MDOVRP

Ins.	Dep.	Cus.	BKS	MD-MTA (aug.9-12, s.200)				POMO (aug.8, s.400)				GAT-RL (aug.8, s.400)				TAOA (aug.8, s.400)			
				B.O.	B.G.	A.O.	A.G.	B.O.	B.G.	A.O.	A.G.	B.O.	B.G.	A.O.	A.G.	B.O.	B.G.	A.O.	A.G.
p01	4	50	386	386	**0.00%**	388	**0.35%**	387	0.22%	388	0.48%	391	1.31%	396	2.61%	394	1.96%	400	3.51%
p02	4	50	376	376	**0.00%**	378	**0.60%**	377	0.39%	379	0.92%	384	2.03%	390	3.63%	388	3.13%	394	4.72%
p03	5	75	475	475	**0.00%**	481	**1.38%**	480	1.06%	484	1.91%	485	2.28%	489	3.12%	486	2.49%	492	3.76%
p04	2	100	662	662	**0.00%**	670	**1.16%**	668	0.93%	672	1.53%	679	2.60%	689	4.11%	683	3.18%	693	4.69%
p05	2	100	608	608	**0.00%**	608	**0.06%**	618	1.78%	618	1.78%	622	2.37%	634	4.34%	632	3.96%	643	5.77%
p06	3	100	612	612	**0.00%**	612	**0.08%**	619	1.18%	628	2.65%	624	1.94%	628	2.60%	625	2.07%	630	2.89%
p07	4	100	608	608	**0.00%**	610	**0.26%**	613	0.78%	621	2.09%	617	1.51%	630	3.65%	626	2.95%	637	4.76%
p08	2	249	2776	2853	**2.76%**	2940	**5.90%**	2921	5.23%	3024	8.94%	2993	7.81%	3064	10.36%	3042	9.56%	3087	11.18%
p09	3	249	2578	2638	**2.32%**	2676	**3.79%**	2710	5.11%	2720	5.49%	2746	6.51%	2813	9.11%	2815	9.17%	2835	9.94%
p10	4	249	2482	2550	**2.72%**	2636	**6.18%**	2648	6.67%	2737	10.26%	2714	9.34%	2763	11.31%	2735	10.17%	2815	13.39%
p11	5	249	2468	2558	**3.64%**	2628	**6.48%**	2632	6.63%	2681	8.61%	2712	9.86%	2760	11.80%	2723	10.30%	2753	11.51%
average			1276	1302	**1.04%**	1330	**2.39%**	1334	2.72%	1359	4.06%	1361	4.32%	1387	6.06%	1377	5.36%	1398	6.92%

is at least 5.02%. Besides, the average 'B.G.' to the BKS of our MD-MTA is 1.07%, while the average 'B.G.' to the BKS of other DRL-based baselines is at least 2.33%. Thus, our MD-MTA performs well on generalization experiments of MDVRP.

Results on MDOVRP: Table VIII reports the results of the MDOVRP. Similar observations can be found in MDOVRP. Our MD-MTA performs well on MDOVRP.

D. Ablation Studies

To highlight network design and depot rotation augmentation, we conducted ablation studies.

Setup: We uniformly adopt a sampling strategy and set a total sample size of 2000. For example, the network with 'aug.10' augmentation will sample 2000 / 10 = 200 solutions, the network with 'aug.8' augmentation will sample 2000 / 8 = 250 solutions, and the network without augmentation will sample 2000 solutions.

Results: Table IX displays the results of objective values as 'Obj.' and the gaps as 'Gap'. Setting 1 and setting 4 show that the network of MD-MTA can improve the solutions by 0.63%. Setting 1 and setting 2 show that such depot rotation improves the objective value by 0.39%; setting 1 and setting 3 show that the performance of deploying both depot rotation

TABLE IX
THE RESULT OF ABLATION STUDIES

Setting	Network	Augment	MDVRP		MDOVRP	
			Obj.	Gap	Obj.	Gap
1	MD-MTA	aug.10, s.200	12.007	0.00%	8.126	0.00%
2	MD-MTA	aug.8, s.250	12.053	0.39%	8.162	0.45%
3	MD-MTA	s.2000	12.116	0.91%	8.220	1.17%
4	POMO	aug.10, s.400	12.112	0.88%	8.195	0.85%
5	POMO	aug.8, s.250	12.147	1.17%	8.217	1.12%
6	POMO	s.2000	12.205	1.65%	8.298	2.12%

augmentation and coordinate rotation augmentation simultaneously can further improve the objective value by 0.91%. All the above observations validate the contributions of the designs.

VI. CONCLUSION AND FUTURE WORK

In this paper, we solve the MDVRP and MDOVRP through a DRL-based approach. We formulate these problems as both mathematical programming and RL models and propose a DRL-based method called MD-MTA to learn a policy for route construction. Specifically, the MD-MTA consists of a multi-type embedding encoder and a multi-type attention decoder. We also propose depot rotation augmentation to further improve the performance. Comparison results show that our MD-MTA outstrips the existing baselines on synthetic and benchmark datasets. Generalization results show that our MD-MTA network is robust. Ablation results show that each design in our MD-MTA can contribute to the overall performance enhancement. For future work, we will look into the generalization performance.

APPENDIX A
MATHEMATICAL FORMULATION

An MDVRP or an MDOVRP instance contains a set $X = \{x_i\}_{i=0}^{n-1}$ with n overall nodes. The subset $X_d = \{x_i\}$ $(i \in [0, d-1], i \in \mathbb{R}^+)$ comprises d depots, and the other subset $X_c = \{x_i\}$ $(i \in [d, n-1], i \in \mathbb{R}^+)$ comprises c customers, with $d + c = n$. Each node x_i is featured by (G_i, D_i), where G_i represents the two-dimensional location coordinates of x_i and D_i denotes the demand x_i. Let C_{max} denote a fixed maximum capacity for each vehicle, $Z(x_i, x_j)$ denote the Euclidean distance from node x_i to node x_j, and C_{ij} denote the remaining capacity of the vehicle before travelling from node x_i to node x_j (the initial capacity equals C_{max}). The binary variable $v_{ij} = 1$ indicates that the vehicle directly travelled from x_i to x_j, and $v_{ij} = 0$ otherwise. The objective of both MDVRP and MDOVRP is to minimize the total route length of the vehicles for serving all customers, which is defined as

$$\min \sum_{i\in X}\sum_{j\in X} Z(x_i, x_j)v_{ij}. \tag{25}$$

The two problems should satisfy the constraints as follows,

$$\sum_{i\in X} v_{ij} = 1, j \in X_c \tag{26}$$

$$\sum_{j\in X} v_{ij} = 1, i \in X_c \tag{27}$$

$$\sum_{i\in X} v_{ij} - \sum_{k\in X} v_{jk} = 0, j \in X_c \tag{28}$$

$$\sum_{i\in X} C_{ij} - \sum_{k\in X} C_{jk} = D_j, j \in X_c \tag{29}$$

$$v_{ij} = \{0, 1\}, i \in X, j \in X \tag{30}$$

$$C_{ij} \geq 0, D_i \geq 0, i \in X, j \in X \tag{31}$$

$$Z(x_i, x_j) = 0, i \in X_c, j \in X_d. \tag{32}$$

Constraints (26) and (27) ensure that each customer can be visited exactly once. Constraint (28) ensures that each route is continuous. Constraint (29) guarantees that the difference in remaining capacity before and after the vehicle serves a customer equals its demand. Constraint (30) defines the binary decision variable and constraint (31) defines the non-negativity of the capacity and the demand. Constraint (32) is only applicable to MDOVRP, which ensures that the vehicles do not return to the depot.

REFERENCES

[1] N. Christofides and S. Eilon, "An algorithm for the vehicle-dispatching problem," *J. Oper. Res. Soc.*, vol. 20, no. 3, pp. 309–318, Sep. 1969.

[2] J. Renaud, G. Laporte, and F. F. Boctor, "A Tabu search heuristic for the multi-depot vehicle routing problem," *Comput. Oper. Res.*, vol. 23, no. 3, pp. 229–235, Mar. 1996.

[3] P. P. Repoussis, C. D. Tarantilis, O. Bräysy, and G. Ioannou, "A hybrid evolution strategy for the open vehicle routing problem," *Comput. Oper. Res.*, vol. 37, no. 3, pp. 443–455, Mar. 2010.

[4] B. Ombuki-Berman and F. T. Hanshar, "Using genetic algorithms for multi-depot vehicle routing," in *Bio-inspired Algorithms for the Vehicle Routing Problem.* Berlin, Germany: Springer, 2009, pp. 77–99.

[5] K. Braekers, K. Ramaekers, and I. Van Nieuwenhuyse, "The vehicle routing problem: State of the art classification and review," *Comput. Ind. Eng.*, vol. 99, pp. 300–313, Sep. 2016.

[6] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 6000–6010.

[7] D. Silver et al., "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, Jan. 2016.

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

[9] Y.-D. Kwon, J. Choo, B. Kim, I. Yoon, Y. Gwon, and S. Min, "Pomo: Policy optimization with multiple optima for reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 21188–21198.

[10] Y. Wu, W. Song, Z. Cao, J. Zhang, and A. Lim, "Learning improvement heuristics for solving routing problems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 9, pp. 5057–5069, Sep. 2022.

[11] Y. Ma et al., "Learning to iteratively solve routing problems with dual-aspect collaborative transformer," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 11096–11107.

[12] Y. Ma et al., "Efficient neural neighborhood search for pickup and delivery problems," in *Proc. 31st Int. Joint Conf. Artif. Intell.*, Jul. 2022, pp. 4776–4784.

[13] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, nos. 3–4, pp. 229–256, 1992.

[14] A. Lim and F. Wang, "Multi-depot vehicle routing problem: A one-stage approach," *IEEE Trans. Autom. Sci. Eng.*, vol. 2, no. 4, pp. 397–402, Oct. 2005.

[15] R. Liu, Z. Jiang, and N. Geng, "A hybrid genetic algorithm for the multi-depot open vehicle routing problem," *OR Spectr.*, vol. 36, no. 2, pp. 401–421, Mar. 2014.

[16] M. Soto, M. Sevaux, A. Rossi, and A. Reinholz, "Multiple neighborhood search, Tabu search and ejection chains for the multi-depot open vehicle routing problem," *Comput. Ind. Eng.*, vol. 107, pp. 211–222, May 2017.

[17] M. E. H. Sadati, B. Çatay, and D. Aksen, "An efficient variable neighborhood search with Tabu shaking for a class of multi-depot vehicle routing problems," *Comput. Oper. Res.*, vol. 133, Sep. 2021, Art. no. 105269.

[18] I. Bello, H. Pham, Q. V. Le, M. Norouzi, and S. Bengio, "Neural combinatorial optimization with reinforcement learning," 2016, *arXiv:1611.09940*.

[19] O. Vinyals, M. Fortunato, and N. Jaitly, "Pointer networks," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, vol. 28. Cambridge, MA, USA: MIT Press, 2015, pp. 2692–2700.

[20] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27, 2014, pp. 3104–3112.

[21] V. Konda and J. N. Tsitsiklis, "Actor-critic algorithms," Ph.D. dissertation, Massachusetts Inst. Technol., Cambridge, MA, USA, 2002.

[22] Y. Zou, H. Wu, Y. Yin, L. Dhamotharan, D. Chen, and A. K. Tiwari, "An improved transformer model with multi-head attention and attention to attention for low-carbon multi-depot vehicle routing problem," *Ann. Oper. Res.*, pp. 1–20, Jun. 2022.

[23] K. Zhang, X. Lin, and M. Li, "Graph attention reinforcement learning with flexible matching policies for multi-depot vehicle routing problems," *Phys. A, Stat. Mech. Appl.*, vol. 611, Feb. 2023, Art. no. 128451.

[24] R. Bellman, "A Markovian decision process," *J. Math. Mech.*, vol. 6, pp. 679–684, Jan. 1957.

[25] S. Kruk, *Practical Python AI Projects Mathematical Models of Optimization Problems With Google Or-Tools*. Berlin, Germany: Springer, 2018.

[26] L. Gurobi Optimization, *Gurobi Optimizer Reference Manual*. Beaverton, OR, USA: Gurobi Optimization, LLC, 2020.

[27] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

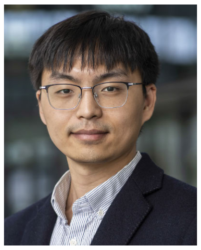
[28] B. E. Gillett and L. R. Miller, "A heuristic algorithm for the vehicle-dispatch problem," *Oper. Res.*, vol. 22, no. 2, pp. 340–349, Apr. 1974.

Jinqi Li received the B.Eng. degree in computer science and technology from China University of Geosciences (Beijing), Beijing, China, in 2021, where he is currently pursuing the M.Eng. degree with the School of Information Engineering. His research interests include deep reinforcement learning for routing problems.

Bing Tian Dai received the Ph.D. degree from the National University of Singapore in 2011. He is currently an Assistant Professor with the School of Computing and Information Systems, Singapore Management University. He is also the Director of the MITB (Artificial Intelligence) Program. Prior to his faculty position, he was a Research Scientist with the Living Analytics Research Centre, working on graph mining and machine learning problems on social media and social networks. His research interests include applying machine learning techniques to solve database problems for data mining applications.

Yunyun Niu received the Ph.D. degree in systems analysis and integration from the Huazhong University of Science and Technology, Wuhan, China, in 2012. She is currently a Professor with the School of Information Engineering, China University of Geosciences, Beijing, China. Her research interests include artificial intelligence and rout planning.

Jianhua Xiao received the Ph.D. degree in system engineering from the Huazhong University of Science and Technology, China, in 2008. He is currently a Professor with the Research Centre of Logistics, Nankai University, Tianjin, China. His current research interests include combinatorial optimization, bio-inspired computation, and logistics system optimization.



Yaoxin Wu received the B.Eng. degree in traffic engineering from Wuyi University, Jiangmen, China, in 2015, the M.Eng. degree in control engineering from Guangdong University of Technology, Guangzhou, China, in 2018, and the Ph.D. degree in computer science from Nanyang Technological University, Singapore, in 2023. He was a Research Associate with the Singtel Cognitive and Artificial Intelligence Laboratory for Enterprises (SCALE@NTU). He is currently an Assistant Professor with the Department of Industrial Engineering and Innovation Sciences, Eindhoven University of Technology, The Netherlands. His research interests include deep learning, combinatorial optimization, and integer programming.