CAMP: Collaborative Attention Model with Profiles for Vehicle Routing Problems

Chuanbo Hua*
KAIST, Omelet
Daejeon, South Korea
cbhua@kaist.ac.kr

Federico Berto*
KAIST, Omelet
Daejeon, South Korea
fberto@kaist.ac.kr

Jiwoo Son*
Omelet
Busan, South Korea
jiwoo.son@omelet.ai

Seunghyun Kang
Omelet
Daejeon, South Korea
seunghyun.kang@omelet.ai

Changhyun Kwon
KAIST, Omelet
Daejeon, South Korea
chkwon@kaist.ac.kr

Jinkyoo Park
KAIST, Omelet
Daejeon, South Korea
jinkyoo.park@kaist.ac.kr

ABSTRACT

The profiled vehicle routing problem (PVRP) is a generalization of the heterogeneous capacitated vehicle routing problem (HCVRP) in which the objective is to optimize the routes of vehicles to serve client demands subject to different vehicle profiles, with each having a preference or constraint on a per-client basis. While existing learning methods have shown promise for solving the HCVRP in real-time, no learning method exists to solve the more practical and challenging PVRP. In this paper, we propose a Collaborative Attention Model with Profiles (CAMP), a novel approach that learns efficient solvers for PVRP using multi-agent reinforcement learning. CAMP employs a specialized attention-based encoder architecture to embed profiled client embeddings in parallel for each vehicle profile. We design a communication layer between agents for collaborative decision-making across profiled embeddings at each decoding step and a batched pointer mechanism to attend to the profiled embeddings to evaluate the likelihood of the next actions. We evaluate CAMP on two variants of PVRPs: PVRP with preferences, which explicitly influence the reward function, and PVRP with zone constraints with different numbers of agents and clients, demonstrating that our learned solvers achieve competitive results compared to both classical state-of-the-art neural multi-agent models in terms of solution quality and computational efficiency. We make our code openly available at https://github.com/ai4co/camp.

KEYWORDS

Neural Combinatorial Optimization, Vehicle Routing Problems, Reinforcement Learning, Attention network

ACM Reference Format:
Chuanbo Hua*, Federico Berto*, Jiwoo Son*, Seunghyun Kang, Changhyun Kwon, and Jinkyoo Park. 2025. CAMP: Collaborative Attention Model with Profiles for Vehicle Routing Problems. In *Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Detroit, Michigan, USA, May 19 – 23, 2025*, IFAAMAS, 10 pages.

*These authors contributed equally to this work. Work made with contributions from the AI4CO open research community.

Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Y. Vorobeychik, S. Das, A. Nowé (eds.), May 19 – 23, 2025, Detroit, Michigan, USA. © 2025 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

1 INTRODUCTION

Vehicle Routing Problems (VRPs) are a class of well-known combinatorial optimization (CO) problems where the objective is to determine the most efficient set of routes for a fleet of vehicles to deliver goods to a set of clients. A particularly challenging variant is the Heterogeneous Capacitated Vehicle Routing Problem (HCVRP), where the fleet consists of vehicles with varying capacities and operational costs [17]. In many real-world scenarios, different vehicles might not only have capacity constraints but also profile-specific preferences or operational constraints that can vary per client. This problem, which we term the Profiled Vehicle Routing Problem (PVRP), generalizes HCVRP by incorporating vehicle profiles with client-specific preferences or constraints [8]. These profiles might affect routing decisions based on factors like vehicle access to particular areas, preferred client relationships, or regulatory requirements for specific vehicle-client combinations [1, 39, 58, 74]. Solving the PVRP with exact solution methods, such as Branch and Bound, becomes impractical for large instances due to its NP-hard nature [49]. Thus, heuristic approaches like genetic algorithms are used in practical scenarios offering approximate solutions trying to balance solution quality and computational efficiency, but they may struggle in larger scales [45] and require considerable manual design and tuning [27].

Recent advances in reinforcement learning (RL) for combinatorial optimization provide a promising alternative with several benefits. RL can automatically learn effective solutions without supervision by interacting with relatively inexpensive simulated environments; thus, designing RL methods requires little to no domain expertise. RL methods can find faster and possibly better solutions than heuristics approaches, particularly in improving scalability to more complex real-world problem instances. Most works follow the pointer network paradigm [2, 62], an encoder-decoder architecture that encodes input data into a shared latent space, which is then used for fast autoregressive decoding [56].

Although recent learning-based approaches have been applied successfully to various VRPs [29, 32, 33, 46], including rich and highly constrained variants [5, 6, 24, 34, 42, 75, 76], multi-agent [11, 14, 15, 50, 54, 71–73, 78] and heterogenous agent (i.e., fleet) VRPs [4, 36, 44], such approaches cannot model heterogeneous agents on a *per-client* basis, in which each agent has a different internal representation for each node.

In this paper, we propose a novel learning approach, the Collaborative Attention Model with Profiles (CAMP), designed to address PVRP using multiagent reinforcement learning (MARL). Our approach builds upon the attention-based encoder-decoder framework [32], integrating vehicle and client profiles into a collaborative decision-making architecture. Unlike previous methods, CAMP leverages a specialized attention-based communication encoder to embed profiled client representations in parallel for each vehicle. During the decoding phase, we employ a specialized communication layer between agents to enable cooperative decisions even among a heterogeneous profiled latent space. Finally, a parallel pointer mechanism [4] is utilized to attend to these profiled embeddings, allowing the model to efficiently evaluate possible next actions for each vehicle based on its profile.

We evaluate CAMP on the PVRP with Preferences (PVRP-P), where various distributions of vehicle profiles explicitly affect the reward function, and PVRP with Zone Constraints (PVRP-ZC), a scenario that imposes zone-based operational constraints on vehicles. Our experiments demonstrate that CAMP achieves competitive results in terms of solution quality and computational efficiency compared to classical methods and modern neural multi-agent models, showcasing CAMP as a valuable tool for research and practitioners in solving complex routing problems in real-time.

2 RELATED WORK

Recent advancements in neural combinatorial optimization (NCO) have introduced promising end-to-end solutions for vehicle routing problems (VRP) [3, 67]. Learning-based VRP approaches can broadly be divided into *construction* [7, 12, 13, 16, 18, 22, 25, 26, 29, 32, 33, 38, 41, 46, 65, 77] and *improvement/search* methods [9, 19, 21, 28, 47, 48, 55, 57, 69, 70]. Construction methods learn to generate a solution, while improvement/search methods iteratively refine them. We focus on learning construction methods due to their lower reliance on handcrafted heuristics and faster inference speed. Most constructive methods derive from the seminal work of Vinyals et al. [62], refined by Bello et al. [2] with deep reinforcement learning, and improved by Kool et al. [32] using transformers in the widely adopted Attention Model (AM). More practical multi-agent VRPs have been formulated by Son et al. [54]and Zheng et al. [72] as sequential decision-making, while Zhang et al. [71] and Zong et al. [78] and Liu et al. [44] employ simultaneous solution construction. For the heterogeneous VRP (HCVRP), approaches have evolved from Vera and Abad [60]'s policy network with A2C method to Qin et al. [53]'s reinforcement learning-enhanced heuristics and Li et al. [35]'s two-decoder framework emphasizing vehicle and node selection. However, these methods often lack fleet generalization due to fixed vehicle number assumptions, and while they attempt simultaneous solutions for various agents, they typically sequentially limit fast parallel decoding and collaboration. PARCO [4] addresses these limitations with a flexible approach using a general communication framework, making it effective for multi-agent and heterogeneous VRP scenarios, enhancing solutions by allowing for dynamic agent counts and improved conflict resolution strategies, representing a significant advancement in addressing the complexities of HCVRPs within the NCO framework. However, no neural approach has yet been proposed to tackle the more practical PVRP.

3 PROBLEM FORMULATION

3.1 Mathematical Formulation of PVRP

Consider a set of nodes $N = \{0, 1, \ldots, n\}$, where node 0 represents the depot and nodes $\{1, \ldots, n\}$ represent clients. Let $C = N \setminus \{0\}$ denote the set of client nodes. The set of vehicles is represented by $K = \{1, \ldots, m\}$. Each node $i \in N$ is characterized by its location $s_i \in \mathbb{R}^2$ and demand d_i (with $d_0 = 0$ for the depot). Each vehicle $k \in K$ is defined by its capacity Q_k, speed s_k, and a profile parameter $p_k = (p_{1k}, \ldots, p_{nk})$, where p_{ik} represents the profile parameter for vehicle k serving client i. Let r denote index for delivery trips, $r \in R$, where R is the maximum number of trips for each vehicle.

Let c_{ij} denote the travel cost from node i to node j, typically representing the Euclidean distance between the nodes. The decision variable is a boolean x^r_{ijk} that equals 1 if vehicle k travels directly from node i to node j in trip r, and 0 otherwise. We also introduce three auxiliary variables to aid in problem formulation and solution: y^r_{ik} is a binary variable that equals 1 if client i is served by vehicle k in trip r, and 0 otherwise. z_k is a binary variable representing the vehicle k is selected or not. Let w_i be a continuous variable used to avoid subtours.

The PVRP can thus be formulated as follows:

$$\min \sum_{k\in K}\sum_{i\in N}\sum_{j\in N}\sum_{r\in R}\left(\frac{c_{ij}}{s_k} - \alpha p_{ik}\right) x^r_{ijk} \tag{1}$$

subject to the following constraints:

$$\sum_{k\in K}\sum_{r\in R} y^r_{ik} = 1, \qquad \forall i \in C \tag{2}$$

$$\sum_{j\in C} x^r_{0jk} \le z_k, \qquad \forall k \in K, r \in R \tag{3}$$

$$\sum_{j\in N} x^r_{ijk} \le y^r_{ik}, \qquad \forall i \in N, k \in K, r \in R \tag{4}$$

$$x^r_{iik} = 0, \qquad \forall k \in K, r \in R \tag{5}$$

$$\sum_{i\in N} x^r_{ihk} - \sum_{j\in N} x^r_{hjk} = 0, \qquad \forall h \in N, k \in K, r \in R \tag{6}$$

$$\sum_{j\in C} x^r_{0jk} = \sum_{i\in C} x^r_{i0k}, \qquad \forall k \in K, r \in R \tag{7}$$

$$\sum_{i\in C}\sum_{j\in C} d_j x^r_{ijk} \le Q_k, \qquad \forall i, j \in N, k \in K, r \in R \tag{8}$$

$$w_j \ge w_i + 1 - N(1 - \sum_k x^r_{ijk}), \qquad \forall i \in C, j \in C, i \ne j, r \in R \tag{9}$$

$$\sum_{j\in C} x^r_{0jk} >= \sum_{j\in C} x^{r+1}_{0jk}, \qquad \forall k \in K, r \le R-1 \tag{10}$$

$$x^r_{ijk}, y^r_{ik}, z_k \in \{0, 1\}, \qquad \forall i, j \in N, k \in K, r \in R \tag{11}$$

The objective function Eq. (1) minimizes the total adjusted travel cost while maximizing preference scores, with α as a weight parameter to balance these two factors. Constraint (2) ensures that each client is visited exactly once. Constraint (3) counts utilized vehicle while leaving from depot. Constraint (4) links the route variables (x^r_{ijk}) with the assignment variables (y^r_{ik}). Constraint (5) bans traveling in the same location. Constraint (6) maintains flow

conservation for each vehicle, while constraint (7) guarantees that each vehicle forces that if a vehicle leaves the depot, it must return to the depot. Constraint (8) ensures that vehicle capacities are respected. Constraint (9) eliminates subtours. Constraints (10) establishes trip sequences. Constraints (11) define the binary for the decision variables.

3.1.1 PVRP with Preferences. The PVRP-P defines the profile parameter p_{ik} as a *preference score*. The objective function Eq. (1) can be simply rewritten as follows:

$$\max \sum_{k\in K}\sum_{i\in N}\sum_{j\in N}\left(\alpha p_{ik} - \frac{c_{ij}}{s_k}\right)x_{ijk}, \tag{12}$$

i.e., maximize the preferences while minimizing the total tour duration with a multi-objective balancing parameter α.

3.1.2 PVRP with Zone Constraints. The PVRP-ZC incorporates zone constraints, where certain vehicles may not serve specific clients. One way to model this problem is to introduce an additional constraint:

$$y_{ik} = 0, \quad \forall (i,k) \text{ where } p_{ik} = 0. \tag{13}$$

In practice, this constraint can be modeled by Eq. (1) by introducing a large negative value, effectively $-\infty$[1], for p_{ik} when vehicle k is not allowed to serve client i:

$$p_{ik} = \begin{cases} -\infty & \text{if vehicle } k \text{ is not allowed to serve client } i \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

and setting α to 1. Using this definition of p_{ik}, the objective function Eq. (12) naturally enforces zone constraints. When $p_{ik} = -\infty$, the term $-\alpha p_{ik}$ becomes ∞, ensuring that the corresponding x_{ijk} variables are set to 0 in any optimal solution, effectively preventing vehicle k from serving client i.

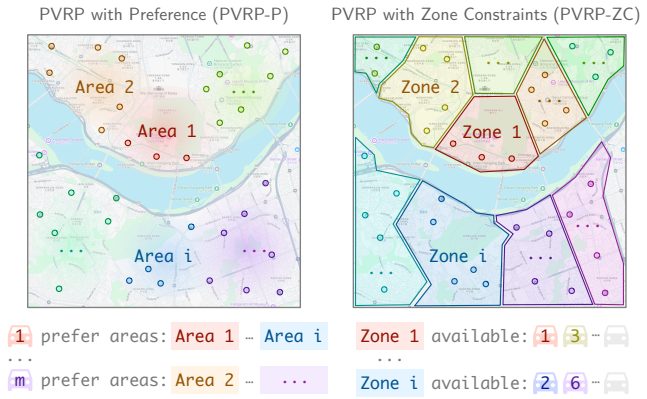

Figure 1: Practical examples of PVRPs. [Left] PVRP-P has preference zones for each vehicle profile. [Right] PVRP-ZC has (hard) zone constraints for certain vehicles.

We show practical examples of PVRP-P (Section 3.1.1) and PVRP-ZC (Section 3.1.2) in Fig. 1.

[1]In practice, a sufficiently large (negative) constant M can be chosen to avoid numerical overflows. For instance, OR-Tools sets $|M| = 2^{48}$.

4 METHODOLOGY

4.1 Modeling PVRP with MARL

We reformulate the Profiled Vehicle Routing Problem (PVRP) as a Markov Decision Process (MDP) defined by the 4-tuple $M = \{S, A, \tau, r\}$, where S is the state space, A is the action space, τ is the state transition function, and r is the reward function.

State Space S. Each state $s_t = (V_t, X_t) \in S$ consists of two parts:

1. Vehicle state $V_t = \{v_t^1, v_t^2, \ldots, v_t^m\}$, where each $v_t^k = (o_t^k, T_t^k, G_t^k, p_k)$ represents:

- o_t^k: Remaining capacity of vehicle k at step t
- T_t^k: Accumulated travel time of vehicle k at step t
- $G_t^k = \{g_0^k, g_1^k, \ldots, g_t^k\}$: Partial route of vehicle k at step t
- $p_k = (p_{1k}, \ldots, p_{nk})$: Profile parameter of vehicle k

The interpretation of p_k differs between PVRP-P and PVRP-ZC:

- For PVRP-P: p_{ik} represents the preference score for vehicle k serving node i
- For PVRP-ZC: p_{ik} is binary, where 1 indicates vehicle k is allowed to serve node i, and 0 indicates it is not allowed

2. Node state $X_t = \{x_t^0, x_t^1, \ldots, x_t^n\}$, where each $x_t^i = (s^i, d_t^i)$ represents:

- s^i: 2D vector representing the location of node i
- d_t^i: Remaining demand of node i at step t

Action Space A. Without loss of generality, we consider an action $a_t \in A$ is defined as selecting a vehicle and a node to visit. Specifically, $a_t = (v_t^k, x_t^j)$, where vehicle k is selected to visit node j at step t.

State Transition Function τ. The transition function τ updates the state based on the chosen action: $s_{t+1} = (V_{t+1}, X_{t+1}) = \tau(V_t, X_t, a_t)$. For PVRP-P, the elements are updated as follows:

$$o_{t+1}^k = \begin{cases} o_t^k - d_t^j, & \text{if } k \text{ is the selected vehicle} \\ o_t^k, & \text{otherwise} \end{cases} \tag{15}$$

$$T_{t+1}^k = \begin{cases} T_t^k + \frac{c_{g_t^k, j}}{s_k}, & \text{if } k \text{ is the selected vehicle} \\ T_t^k, & \text{otherwise} \end{cases} \tag{16}$$

$$G_{t+1}^k = \begin{cases} [G_t^k, j], & \text{if } k \text{ is the selected vehicle} \\ G_t^k, & \text{otherwise} \end{cases} \tag{17}$$

$$d_{t+1}^i = \begin{cases} 0, & \text{if } i \text{ is the selected node} \\ d_t^i, & \text{otherwise} \end{cases} \tag{18}$$

For PVRP-ZC, we modify the transition function to incorporate zone constraints:

$$\tau_{ZC}(V_t, X_t, a_t) = \begin{cases} (V_{t+1}, X_{t+1}), & \text{if } p_{jk} = 1 \\ (V_t, X_t), & \text{if } p_{jk} = 0 \end{cases} \tag{19}$$

i.e., actions for all vehicle k which would lead $p_{jk} = 0$ are directly masked in the environment.

Where (V_{t+1}, X_{t+1}) is calculated using the same update rules as in PVRP-P. This ensures that the state remains unchanged if an invalid action is attempted.

Reward Function r. The reward function is defined as follows for both PVRP-P and PVRP-ZC:

$$r_t = \begin{cases} 0, & \text{if episode is not complete} \\ R(s_T), & \text{if episode is complete (at final step T)} \end{cases} \tag{20}$$

Where $R(s_T)$ is the final reward calculated at the end of the episode:

for PVRP-P:

$$R(s_T) = \sum_{k \in K} \sum_{(i,j) \in G_T^k} \left(\alpha p_{ik} - \frac{c_{ij}}{s_k} \right) \tag{21}$$

for PVRP-ZC:

$$R(s_T) = - \sum_{k \in K} \sum_{(i,j) \in G_T^k} \frac{c_{ij}}{s_k} \tag{22}$$

Where G_T^k is the final route of vehicle k, c_{ij} is the travel cost between nodes i and j, s_k is the speed of vehicle k, p_{ik} is the preference score for vehicle k serving node i, and α is a weight parameter balancing travel cost and preference score (for PVRP-P only).

4.1.1 Optimization Objective. The goal is to find the optimal policy parameters θ^* that maximize the expected cumulative reward. Formally, we aim to solve:

$$\theta^* = \arg\max_{\theta} \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{T} r_t \right] \tag{23}$$

4.1.2 Construction Methods. We note that there are several ways to construct solutions, starting from the formulated MDP, which takes into account the most general case. In particular, we identify the following cases:

(1) *Autoregressive Sequential*: such construction method considers a single vehicle and one action at a time; a vehicle is switched only when its route is complete [54, 72].
(2) *Autoregressive Alternating*: similar to the above but more flexible: in this case, the vehicle can be switched at any time [36, 44].
(3) *Parallel Autoregressive*: in this case, the steps are performed for all agents in parallel simultaneously, reducing the total T, which makes optimization and inference faster [4].

Without loss of generality, we adopt the Parallel Autoregressive approach in CAMP because of its speed and flexibility.

4.2 Collaborative Attention Model with Profiles

Fig. 2 shows an illustration of CAMP model. We first provide an overview of the overall solution construction (Section 4.2.1), then of the encoder (Section 4.2.2) and decoder (Section 4.2.3).

4.2.1 Overall solution construction. We formulate the solution construction of CAMP in a parallel autoregressive approach [4]. We define the solution construction $\boldsymbol{a}$ given instance $\boldsymbol{x}$ as follows:

$$\boldsymbol{h} = f_\theta(x) \tag{24}$$

$$\pi_\theta(\boldsymbol{a}|\boldsymbol{x}) = \prod_{t=0}^{T} \prod_{k=1}^{m} g_\theta(a_t^k | a_{t-1}^k, \ldots, a_0^k, \boldsymbol{h}) \tag{25}$$

where $f_\theta(\cdot)$ is the *encoder* f mapping $\boldsymbol{x}$ to its (vehicle-specific) latent embeddings $\boldsymbol{h}$, $g_\theta(\cdot)$ is the autoregressive *decoder*, and π_θ represents the full CAMP encoder-decoder model mapping $\boldsymbol{x}$ to $\boldsymbol{a}$.

4.2.2 Encoder.

Encoder. The encoder in CAMP is designed to handle multiple vehicle profiles, each with distinct embeddings. Unlike previous approaches that utilize a single shared embedding for all vehicles [4, 54, 72], CAMP encodes each vehicle profile individually, generating profile-specific embeddings. The encoding process involves three key steps: (1) obtaining initial embeddings, (2) applying attention to compute vehicle-specific profile embeddings, and (3) integrating these profile embeddings using bipartite graph message passing.

First, the encoder maps the raw features of the graph instance into an embedding space. Specifically, it transforms the feature vectors of vehicles x^v, clients x^c, and vehicle preference profiles p_{ij} into their respective embedding spaces using the learned embedding matrices W^v_{init}, W^c_{init}, and W^p_{init}. The resulting embeddings are computed as follows:

$$h_i^v = W^v_{\text{init}} \cdot x_i^v$$
$$h_j^c = W^c_{\text{init}} \cdot x_j^c$$
$$h_{ij}^p = W^p_{\text{init}} \cdot p_{ij}$$

Next, the vehicle, client, and preference embeddings are concatenated to form a combined profile embedding, integrating both node and edge features in the graph:

$$h_{ij} = W_{\text{combine}} \cdot \text{concat}(h_i^v, h_j^c, h_{ij}^p)$$

This profile embedding h_{ij}, which captures the interaction between a vehicle and clients, is processed using multi-head attention (MHA) to incorporate profile-specific information. The overall profile embedding $\boldsymbol{h}$ is constructed by concatenating the embeddings of all vehicle-client pairs: $\boldsymbol{h} = \text{concat}(h_1, h_2, \ldots, h_m)$. MHA is then applied separately to each profile:

$$h' = \text{MHA}(h_i, h_i, h_i) \tag{26}$$

where MHA represents the multi-head attention defined as:

$$\text{MHA}(Q, K, V) = \left(\Big\Vert_{i=1}^{n_{\text{heads}}} \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \right) W^O$$

with

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}} \right) V$$

where $\Vert$ denotes concatenation; $W^O \in \mathbb{R}^{d_k \times d_h}$ is used to combine the outputs from different attention heads where $d_k = d_h / n_{\text{heads}}$ represents the dimension per heads; and $W_i^Q, W_i^K, W_i^V \in \mathbb{R}^{d_h \times d_k}$ are the learnable parameter matrices for queries Q, keys K, and values V.

After obtaining the vehicle-specific profile embeddings, the encoder in CAMP employs a bipartite graph message passing framework, which is crucial for integrating information across individual vehicle profile embeddings. By facilitating the exchange of information between vehicle and client nodes, this framework allows each vehicle-specific profile embedding to benefit from the broader

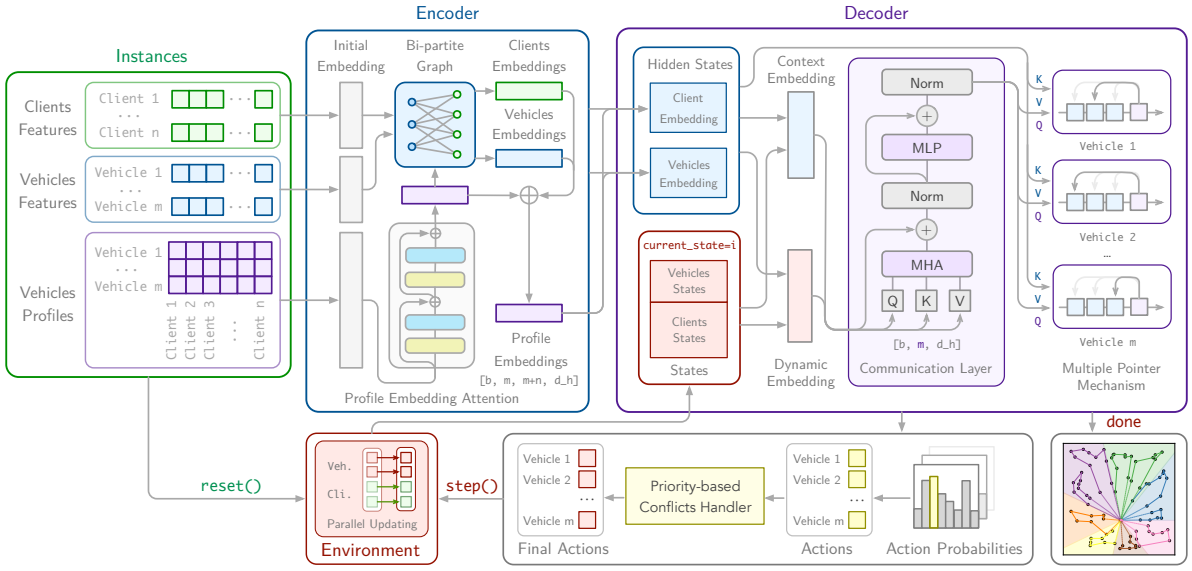

Figure 2: Overview of CAMP.

context, enabling a richer, more interconnected representation. The bipartite graph structure enables two-directional message passing:

- **Vehicle-to-Client Update**: The vehicle embeddings are propagated to the client nodes they are connected to, aggregating vehicle-specific information to update customer embeddings:

$$h_j'^c = \Phi(h_i^v, h_j^c, h_{ij}')$$

- **Client-to-Vehicle Update**: The customer embeddings are subsequently propagated back to the vehicles, allowing vehicles to gather information from multiple customers:

$$h_i'^v = \Phi(h_j^c, h_i^v, h_{ji}')$$

This bidirectional message passing (Φ) enables information to flow across all vehicle-customer pairs, allowing vehicle-specific profile embeddings to interact and integrate with each other.

Residual connections are employed to maintain stability in the learning process, which add the initial node embeddings to the updated embeddings:

$$h_i''^v = h_i^v + h_i'^v, \quad h_j''^c = h_j^c + h_j'^c$$

Finally, the edge embeddings are refined by combining the updated vehicle and customer embeddings, ensuring that the edge representations reflect the fully integrated node information:

$$h_{ij}'' = \text{concat}(h_i^v, h_j^c) + h_{ij}'$$

The aggregated information is then passed through multiple transformer-style layers [59] with bidirectional message passing similarly to Eq. (28). The final output of the encoder is a set of embeddings $\boldsymbol{h} = [\boldsymbol{h}^1, \ldots, \boldsymbol{h}^m]$, where each $\boldsymbol{h}^k \in \mathbb{R}^{(m+n)\times d_h}$ represents the encoded graph of vehicles and clients for vehicle profile k. This approach allows CAMP to capture profile-specific information and relationships between vehicle and client, which is crucial for solving the PVRP.

4.2.3 Decoder. The decoder transforms multi-profile embeddings $\boldsymbol{h}$ into a probability distribution for action selection. We calculate the m queries Q_t for the multiple pointer mechanism [4] as follows:

$$Q_t = W_{\text{proj}}(\text{concat}(\boldsymbol{h}_k^k, \boldsymbol{h}_i^k, W_{\text{context}} \cdot x_t)) \tag{27}$$

where k is the vehicle index, i is the node index, x_t are context features at the current step t; W_{context} is a learnable parameter matrix to project the context features while W_{proj} is a learnable parameter matrix to project back to hidden space d_h the concatenated static embeddings and dynamic features. Then, we apply a communication layer using a transformer block to capture intra-vehicle *dynamic* relationships:

$$\begin{aligned} H' &= \text{Norm}(\text{MHA}(Q_t, Q_t, Q_t) + Q_t) \\ Q_t' &= \text{Norm}(\text{FFN}(H') + H') \end{aligned} \tag{28}$$

where Norm denotes a normalization layer, and FFN represents the multi-layer perceptron. The self-attention in the MHA layer enables message passing between vehicles based on their current embeddings.

After this communication step, we perform cross-attention between the updated vehicle queries and the profile-specific node embeddings. Thus:

$$\text{MHA}(Q_t' W_i^Q, \boldsymbol{h} W_i^K, \boldsymbol{h} W_i^V) \tag{29}$$

where $\boldsymbol{h} = [\boldsymbol{h}^1, \ldots, \boldsymbol{h}^m]$ are the profile-specific node embeddings from the encoder. This formulation allows the attention mechanism to consider the profile-specific encoded information for each vehicle.

The outputs of these attention heads are then combined to form $U \in \mathbb{R}^{m\times d_h}$, which is used in our Multiple Pointer Mechanism:

$$Z = C \cdot \tanh\left(\frac{UL^\top}{\sqrt{d_h}}\right) \tag{30}$$

Here, L is a projection of the node embeddings $\boldsymbol{h}$, and C is a scale parameter to control the entropy of Z ($C = 10$ in our work according to Bello et al. [2]). The resulting $Z \in \mathbb{R}^{m \times N}$ contains logits for each vehicle-node pair. Finally, the logit vector Z of Eq. (30) is masked based on the solution construction feasibility. The probability of selecting action i for all vehicles is given by:

$$P_i = \frac{e^{Z_i}}{\sum_i e^{Z_i}} \tag{31}$$

allowing us to sample a vector A containing m actions for each vehicle in parallel. If a conflict arises between actions, i.e., two or more vehicles select an action of index i, we prioritize the vehicle with the largest action probability P_i and set the action of all other conflicting vehicles equivalent to their current position as in Berto et al. [4].

4.3 Training

CAMP is a centralized multi-agent parallel decision-making model, with a shared policy π_θ for all agents and a global reward R. As such, CAMP can be trained using any of the training algorithms from single-agent NCO literature. We train CAMP using the REINFORCE gradient estimator [63] with a shared baseline [29, 33]:

$$\nabla_\theta \mathcal{L} \approx \frac{1}{B \cdot L} \sum_{i=1}^{B} \sum_{j=1}^{L} G_{ij} \nabla_\theta \log p_\theta(A_{ij} | \boldsymbol{x}_i). \tag{32}$$

Here, B is the mini-batch size and G_{ij} is the advantage $R(A_{ij}, \boldsymbol{x}_i) - b^{\text{shared}}(\boldsymbol{x}_i)$ of a solution A_{ij} w.r.t. to the shared baseline $b^{\text{shared}}(\boldsymbol{x}_i)$ of problem instance $\boldsymbol{x}_i$, in our case obtained through symmetric transformations [29].

For PVRP-P, different preference distributions can lead to rewards of varying scales in the reward function. To mitigate potential biases during learning, we propose applying reward balancing across these distributions, which has been successfully applied in multi-task routing problems [5]. We employ reward balancing techniques to calculate the normalized reward $R^{(k)}_{\text{norm},t}$ for all preference distributions $k \in [1, \cdots, K]$ at each training step $t \geq 1$. This normalization is achieved by dividing by the exponentially smoothed mean. We calculate the average reward $\hat{R}^{(k)}_t$ up to training step t, starting from the average reward $\bar{R}^{(k)}_t$ at step t. For the exponential moving average, we set $\hat{R}^{(k)}_1 = \bar{R}^{(k)}_1$ and compute subsequent values for $t > 1$ based on [23], using a smoothing factor β:

$$\hat{R}^{(k)}_t = (1 - \beta) \cdot \hat{R}^{(k)}_{t-1} + \beta \cdot \bar{R}^{(k)}_t, \quad 0 < \beta < 1, \quad t > 1. \tag{33}$$

We then calculate the normalized reward as $R^{(k)}_{\text{norm},t} = R^{(k)}_t / |\hat{R}^{(k)}_t|$ to ensure fairness in reward distribution.

5 EXPERIMENTS

5.1 Experimental Setup

Classical Baselines. We employ Google's OR-Tools [51], which is both an exact and heuristic solver that utilizes constraint programming. OR-Tools is highly regarded within the operations research community for its flexibility in handling a broad spectrum of VRP variants and other problems. We also utilize PyVRP [64], an open-source heuristic VRP solver that represents the latest advancements and is based on HGS-CVRP [61]. PyVRP is capable of addressing vehicle profile constraints by accordingly modifying vehicle-wise cost matrices as we detail in Section 3. Both baseline approaches are applied to solve each problem instance using a single CPU core.

Models. We evaluate the following neural methods:

- `ET` [54] is an advanced solution for multi-agent TSP and PDP, focusing on generating sequential actions and distributing workloads equitably.
- `DPN` [72] improves ET by using a unique encoder that improves route partitioning and navigation, significantly enhancing efficiency over previous methods.
- `2D-Ptr` [44] dynamically adapts to varying scenarios in HCVRP using a dual-encoder system to optimize vehicle and client routes efficiently.
- `PARCO` [4] is a recent method that accelerates multi-agent combinatorial optimization using a novel decoding mechanism and communication layers, achieving fast and competitive results.
- `CAMP(-EC)` is our proposed model without the Encoder Communication (EC), i.e., without the bipartite graph message passing of Section 4.2.2.
- `CAMP` is our full model.

For fairness of comparison, we try to match the number of training steps to be the same and adjust the batch size accordingly. Specifically, we train models for 100 epochs as in Kool et al. [32] using the Adam optimizer [30] with an initial learning rate (LR) of 10^{-4} with a decay factor of 0.1 after the 80th and 95th epochs with the SymNCO training scheme [29]. Every epoch, we sample 10^5 samples. The policy gradients shared baseline is made of 8 augmented symmetrical solutions out of a 128-batch size. We train by sampling locations and random preference scores from uniform distributions. We set the embedding dimension to 128, the number of attention heads to 8, and the feedforward hidden dimension to 512 across all models and employ 3 encoder layers.

Environment Settings. We evaluate the trained models under four types of environment settings:

- `Random`: each vehicle has a random preference score between 0 and 1 for each client.
- `Angle`: based on the depot's location, the region is divided into sectors by angle. Each vehicle is randomly assigned to one sector, and all clients within that sector are given a preference score of 1 for this vehicle, while clients outside the sector have a score of 0.
- `Cluster`: a number of vehicle cluster centers are randomly placed in the region. Each vehicle is assigned to one cluster. The preference score for each vehicle for each client is the Euclidean distance from the client to the respective cluster center.
- `Zone`: cluster centers are randomly placed in the region from the number of vehicles to three times that number. Each client belongs to the zone of the nearest cluster center. Each zone is randomly designated as available or unavailable to each vehicle.

Evaluation Settings. We randomly generate 1280 instances for each type of environment setting, with varying numbers of vehicles

Table 1: Benchmarks and results for PVRP-P (top) and PVRP-ZC (bottom) at varying sizes and agent numbers. Highlighting cost (↓) and average gaps (↓) to the sota HGS-PyVRP. Single instance solution time in (·).

PVRP-P (Preferences)										
N	60			80			100			Gap(%)
m	3	5	7	3	5	7	3	5	7	avg.
OR-Tools	7.34 (10m)	7.64 (10m)	7.87 (10m)	8.42 (12m)	9.02 (12m)	9.19 (12m)	10.15 (15m)	10.32 (15m)	10.62 (15m)	10.68
HGS-PyVRP	6.44 (10m)	6.83 (10m)	7.10 (10m)	7.66 (12m)	8.17 (12m)	8.47 (12m)	8.93 (15m)	9.50 (15m)	9.80 (15m)	0.00
ET (*g.*)	7.62(0.17s)	8.12(0.17s)	8.41(0.18s)	9.04(0.23s)	9.63(0.24s)	9.96(0.23s)	10.50(0.28s)	11.18(0.30s)	11.63(0.30s)	18.13
DPN (*g.*)	7.52(0.17s)	8.00(0.18s)	8.27(0.18s)	8.93(0.22s)	9.63(0.24s)	10.05(0.24s)	10.50(0.29s)	11.08(0.29s)	11.42(0.30s)	17.15
2D-Ptr (*g.*)	7.34(0.15s)	7.75(0.15s)	8.03(0.16s)	8.70(0.20s)	9.30(0.19s)	9.59(0.20s)	10.10(0.25s)	10.82(0.25s)	11.18(0.25s)	13.86
PARCO (*g.*)	7.31(0.14s)	7.73(0.15s)	8.08(0.15s)	8.69(0.21s)	9.19(0.22s)	9.57(0.22s)	10.14(0.25s)	10.78(0.24s)	11.10(0.25s)	13.21
CAMP(-EC) (*g.*)	7.30(0.16s)	7.67(0.15s)	7.90(0.16s)	8.68(0.22s)	9.10(0.21s)	9.49(0.21s)	9.98(0.25s)	10.72(0.25s)	11.00(0.26s)	12.53
CAMP (*g.*)	**7.16**(0.17s)	**7.66**(0.18s)	**7.88**(0.18s)	**8.49**(0.25s)	**9.07**(0.25s)	**9.44**(0.25s)	**9.87**(0.33s)	**10.60**(0.33s)	**10.90**(0.33s)	**11.23**
ET (*s.*)	7.16(0.25s)	7.62(0.24s)	7.87(0.25s)	8.51(0.35s)	9.05(0.37s)	9.41(0.36s)	9.93(0.45s)	10.55(0.44s)	10.88(0.46s)	11.23
DPN (*s.*)	7.13(0.27s)	7.55(0.26s)	7.87(0.26s)	8.46(0.34s)	9.04(0.37s)	9.36(0.36s)	9.85(0.43s)	10.52(0.42s)	10.88(0.48s)	10.69
2D-Ptr (*s.*)	6.89(0.16s)	7.31(0.17s)	7.57(0.17s)	8.23(0.20s)	8.72(0.21s)	9.09(0.22s)	9.57(0.26s)	10.14(0.26s)	10.45(0.27s)	6.81
PARCO (*s.*)	6.87(0.21s)	7.25(0.23s)	7.55(0.22s)	8.13(0.33s)	8.66(0.35s)	9.02(0.34s)	9.44(0.42s)	10.12(0.41s)	10.49(0.41s)	6.44
CAMP(-EC) (*s.*)	6.82(0.22s)	7.20(0.23s)	7.54(0.23s)	8.12(0.33s)	8.63(0.32s)	8.95(0.34s)	9.42(0.40s)	10.05(0.42s)	10.32(0.41s)	5.72
CAMP (*s.*)	**6.75**(0.33s)	**7.18**(0.35s)	**7.45**(0.33s)	**8.05**(0.43s)	**8.57**(0.42s)	**8.90**(0.42s)	**9.38**(0.51s)	**9.96**(0.54s)	**10.29**(0.54s)	**5.02**
PVRP-ZC (Zone Constraints)										
N	60			80			100			Gap(%)
m	3	5	7	3	5	7	3	5	7	avg.
OR-Tools	13.17(10m)	13.15(10m)	13.18(10m)	16.67(12m)	16.69(12m)	16.70(12m)	20.18(15m)	20.22(15m)	20.20(15m)	8.35
HGS-PyVRP	12.13(10m)	12.16(10m)	12.16(10m)	15.36(12m)	15.40(12m)	15.42(12m)	18.59(15m)	18.66(15m)	18.70(15m)	0.00
ET (*g.*)	13.87(0.17s)	14.04(0.17s)	13.91(0.17s)	17.55(0.23s)	17.54(0.23s)	17.48(0.24s)	21.28(0.28s)	21.20(0.29s)	21.36(0.29s)	14.15
DPN (*g.*)	13.98(0.17s)	13.94(0.17s)	13.81(0.17s)	17.50(0.22s)	17.62(0.22s)	17.57(0.23s)	21.24(0.28s)	21.31(0.28s)	21.37(0.28s)	14.14
2D-Ptr (*g.*)	13.42(0.15s)	13.49(0.15s)	13.37(0.15s)	16.98(0.20s)	16.95(0.20s)	17.04(0.20s)	20.53(0.25s)	20.58(0.24s)	20.57(0.25s)	10.06
PARCO (*g.*)	13.39(0.14s)	13.42(0.13s)	13.32(0.13s)	16.87(0.21s)	16.95(0.20s)	16.82(0.20s)	20.33(0.25s)	20.52(0.25s)	20.33(0.25s)	9.55
CAMP(-EC) (*g.*)	13.05(0.17s)	13.19(0.16s)	**13.05**(0.16s)	**16.58**(0.22s)	16.73(0.23s)	16.65(0.22s)	20.10(0.25s)	**20.07**(0.25s)	20.07(0.26s)	7.69
CAMP (*g.*)	**12.93**(0.19s)	**13.07**(0.20s)	13.14(0.18s)	16.59(0.25s)	**16.38**(0.25s)	**16.52**(0.24s)	**20.00**(0.33s)	20.18(0.32s)	**20.07**(0.33s)	**7.62**
ET (*s.*)	13.18(0.25s)	13.17(0.25s)	13.18(0.25s)	16.63(0.34s)	16.73(0.34s)	16.58(0.34s)	20.19(0.46s)	20.47(0.45s)	20.14(0.45s)	8.78
DPN (*s.*)	13.13(0.27s)	13.30(0.27s)	13.24(0.26s)	16.62(0.33s)	16.67(0.33s)	16.78(0.34s)	20.23(0.44s)	20.18(0.44s)	20.34(0.43s)	8.69
2D-Ptr (*s.*)	12.86(0.15s)	12.93(0.15s)	12.99(0.15s)	16.33(0.21s)	16.31(0.21s)	16.17(0.22s)	19.76(0.25s)	19.72(0.25s)	19.90(0.25s)	5.92
PARCO (*s.*)	12.85(0.21s)	12.81(0.22s)	12.86(0.22s)	16.28(0.33s)	16.46(0.33s)	16.31(0.33s)	19.64(0.42s)	19.72(0.42s)	19.79(0.41s)	5.86
CAMP(-EC) (*s.*)	**12.49**(0.22s)	12.58(0.23s)	12.56(0.23s)	15.87(0.33s)	15.92(0.32s)	**15.86**(0.33s)	19.20(0.41s)	**19.19**(0.41s)	19.18(0.40s)	3.34
CAMP (*s.*)	12.54(0.33s)	**12.50**(0.33s)	**12.52**(0.34s)	**15.77**(0.42s)	**15.84**(0.43s)	15.88(0.42s)	**19.06**(0.50s)	19.26(0.49s)	**19.18**(0.49s)	**3.31**

and clients. All baselines are evaluated across settings of α ranging from 0.0 to 0.2 for PVRP-P. We calculate the average cost among different preference distributions as the final cost for PVRP-P.

Evaluation runs are conducted on an AMD Ryzen Threadripper 3960X 24-core CPU with a single RTX 3090 GPU. We value open reproducibility and provide source code on Github [2].

5.2 Experiment Results

Table 1 compares CAMP against previously discussed baselines for PVRP-P and PVRP-ZC, with inference times shown in parentheses (·); with (*g.*) referring to greedy performance while (*s.*) refers to sampling 1280 solutions. We observe that CAMP consistently outperforms all other neural solver baselines in experiments across all client and vehicle sizes for both PVRP-P and PVRP-ZC. Fig. 3 displays the Pareto curve for all models at various α values under different preference distributions. The higher the α, the greater the weight given to preferences relative to costs. It is evident that CAMP offers a significant advantage in optimizing both duration and preference compared to all other baselines across all preference distributions. As the weight of the preference increases, CAMP improves more effectively in optimizing preference compared to other models while maintaining robust performance in duration. This indicates our model's superior capability in capturing vehicle preference information.

Fig. 4 qualitatively visualizes the solutions constructed by CAMP for instances of PVRP-P with angle and cluster preference distributions. Each color represents a vehicle's preferred area and its corresponding route. It is noteworthy that CAMP effectively constructs solutions that respect the preferences of vehicles while maintaining optimal duration. This results in a superior solution that optimizes both duration and preference.

[2] https://github.com/ai4co/camp

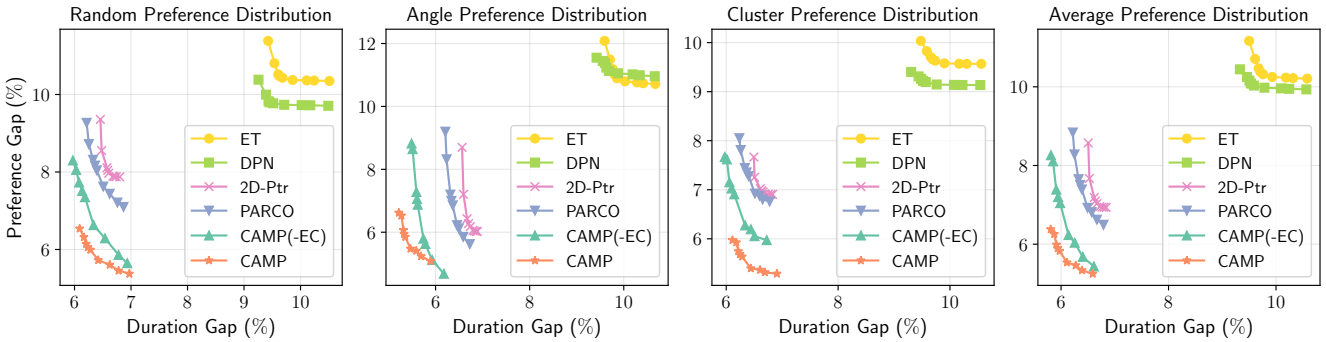

Figure 3: Pareto plot of VRP-P gaps at varying values of α for different preference matrix distribution. The bottom left is better.

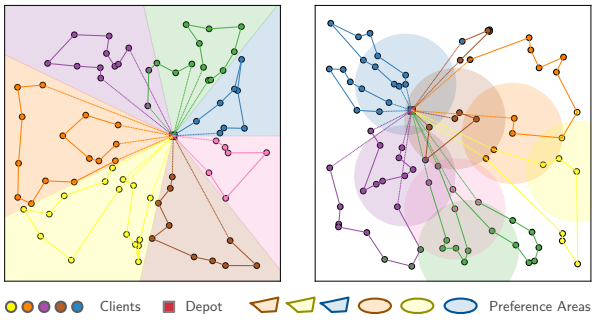

Figure 4: Visualization of CAMP solutions of PVRP-P with angle and cluster preference distribution. CAMP successfully routes according to the preference distribution.

5.3 Ablation Studies

Table 2 Illustrates the main ablation studies for the CAMP components on PVRP-P with instance size $N = 100$. We compare the performance of the full CAMP model against its variants with specific components ablated one at a time.

Table 2: Ablation study for CAMP model showing gaps (↓) removing different components.

Model	Dur. Gap	Pref. Gap
CAMP (full)	**6.52%**	**4.98%**
- Encoder Communication	6.55%	5.68%
- Balanced Reward Training	6.63%	5.88%
- Vehicle-specific Profile Embedding	6.85%	6.35%

Encoder Communication. This ablation (CAMP-EC in Table 1) underpins the effectiveness of encoder communication due to the bipartite graph, which significantly enhances the representation capability of CAMP in capturing the preference relationships, leading to improved optimization of the preference gap.

Balanced Reward Training. Comparing CAMP without Encoder Communication and Balanced Reward, we observe that the latter successfully equilibrates the rewards across different preference settings, making it easier for the model to adapt and thereby enhancing performance in managing preferences.

Vehicle-specific Profile Embedding. This setting removes all our contributed components in CAMP and corresponds to PARCO [4]. This overall performs worst, although still better than other baselines such as the sequential autoregressive ones.

6 CONCLUSION

In this work, we introduced a formulation for the Profiled Vehicle Routing Problem (PVRP), an extension of the Heterogeneous Capacitated Vehicle Routing Problem (HCVRP) that incorporates client-specific preferences and operational constraints. To tackle this complex problem, we proposed the Collaborative Attention Model with Profiles (CAMP), a novel multi-agent reinforcement learning (MARL) approach that leverages an attention-based encoder-decoder framework with agent communication to enable collaborative decision-making among heterogeneous vehicles with different profiles for each client. Our extensive evaluations on both synthetic, across two types of PVRP variants—PVRP with Preferences (PVRP-P) and PVRP with Zone Constraints (PVRP-ZC)—show that CAMP consistently delivers competitive performance in terms of solution quality and computational efficiency, outperforming traditional heuristics and other neural-based methods. These results position CAMP as a powerful tool for solving complex routing problems in dynamic, real-time settings.

Limitations and Future Work. CAMP represents an early attempt at solving the PVRP. While it shows promising results in solution time and performance – including outperforming OR-Tools – it can be improved in several ways to beat the final performance of SOTA heuristic HGS. Promising future works include integrating end-to-end construction and improvement methods [31], learning to guide (local) search algorithms [20, 37, 66], multi-objective learning at different preference values α [10, 40], extending recent foundation models for VRPs [5, 34] with CAMP's *agentic* representations including agent communication and heterogenous learned representations, and obtaining better heuristics for resolving decoding conflicts which could be achieved by automated LLM algorithmic discovery [43, 52, 68].

REFERENCES

[1] Satomi AIKO, Phathinan THAITHATKUL, and Yasuo Asakura. 2018. Incorporating user preference into optimal vehicle routing problem of integrated sharing transport system. *Asian Transport Studies* 5, 1 (2018), 98–116.

[2] Irwan Bello, Hieu Pham, Quoc V Le, Mohammad Norouzi, and Samy Bengio. 2016. Neural combinatorial optimization with reinforcement learning. *arXiv preprint arXiv:1611.09940* (2016).

[3] Yoshua Bengio, Andrea Lodi, and Antoine Prouvost. 2021. Machine learning for combinatorial optimization: a methodological tour d'horizon. *European Journal of Operational Research* 290, 2 (2021), 405–421.

[4] Federico Berto, Chuanbo Hua, Laurin Luttmann, Jiwoo Son, Junyoung Park, Kyuree Ahn, Changhyun Kwon, Lin Xie, and Jinkyoo Park. 2024. PARCO: Learning Parallel Autoregressive Policies for Efficient Multi-Agent Combinatorial Optimization. *arXiv preprint arXiv:2409.03811* (2024). https://github.com/ai4co/parco.

[5] Federico Berto, Chuanbo Hua, Nayeli Gast Zepeda, André Hottung, Niels Wouda, Leon Lan, Kevin Tierney, and Jinkyoo Park. 2024. RouteFinder: Towards Foundation Models for Vehicle Routing Problems. In *ICML 2024 Workshop on Foundation Models in the Wild (Oral)*. https://openreview.net/forum?id=hCiaiZ6e4G https://github.com/ai4co/routefinder.

[6] Jieyi Bi, Yining Ma, Jianan Zhou, Wen Song, Zhiguang Cao, Yaoxin Wu, and Jie Zhang. 2024. Learning to Handle Complex Constraints for Vehicle Routing Problems. *arXiv preprint arXiv:2410.21066* (2024).

[7] Aigerim Bogyrbayeva, Taehyun Yoon, Hanbum Ko, Sungbin Lim, Hyokun Yun, and Changhyun Kwon. 2023. A deep reinforcement learning approach for solving the traveling salesman problem with drone. *Transportation Research Part C: Emerging Technologies* 148 (2023), 103981.

[8] Kris Braekers, Katrien Ramaekers, and Inneke Van Nieuwenhuyse. 2016. The vehicle routing problem: State of the art classification and review. *Computers & industrial engineering* 99 (2016), 300–313.

[9] Federico Julian Camerota Verdù, Lorenzo Castelli, and Luca Bortolussi. 2025. Scaling Combinatorial Optimization Neural Improvement Heuristics with Online Search and Adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

[10] Jinbiao Chen, Zizhen Zhang, Zhiguang Cao, Yaoxin Wu, Yining Ma, Te Ye, and Jiahai Wang. 2024. Neural multi-objective combinatorial optimization with diversity enhancement. *Advances in Neural Information Processing Systems* 36 (2024).

[11] Elija Deineko and Carina Kehrt. 2024. Learn to Solve Vehicle Routing Problems ASAP: A Neural Optimization Approach for Time-Constrained Vehicle Routing Problems with Finite Vehicle Fleet. *arXiv preprint arXiv:2411.04777* (2024).

[12] Darko Drakulic, Sofia Michel, and Jean-Marc Andreoli. 2024. GOAL: A Generalist Combinatorial Optimization Agent Learning. *arXiv preprint arXiv:2406.15079* (2024).

[13] Darko Drakulic, Sofia Michel, Florian Mai, Arnaud Sors, and Jean-Marc Andreoli. 2024. Bq-nco: Bisimulation quotienting for efficient neural combinatorial optimization. *Advances in Neural Information Processing Systems* 36 (2024).

[14] Jonas K Falkner and Lars Schmidt-Thieme. 2020. Learning to solve vehicle routing problems with time windows through joint attention. *arXiv preprint arXiv:2006.09100* (2020).

[15] Ricardo Gama, Daniel Fuertes, Carlos R del Blanco, and Hugo L Fernandes. 2024. Multi-Agent Environments for Vehicle Routing Problems. *arXiv preprint arXiv:2411.14411* (2024).

[16] Chengrui Gao, Haopu Shang, Ke Xue, Dong Li, and Chao Qian. 2023. Towards Generalizable Neural Solvers for Vehicle Routing Problems via Ensemble with Transferrable Local Policy. *arXiv preprint arXiv:2308.14104* (2023).

[17] Bruce Golden, Arjang Assad, Larry Levy, and Filip Gheysens. 1984. The fleet size and mix vehicle routing problem. *Computers & Operations Research* 11, 1 (1984), 49–66.

[18] Nathan Grinsztajn, Daniel Furelos-Blanco, Shikha Surana, Clément Bonnet, and Tom Barrett. 2023. Winner takes it all: Training performant RL populations for combinatorial optimization. *Advances in Neural Information Processing Systems* 36 (2023), 48485–48509.

[19] André Hottung, Yeong-Dae Kwon, and Kevin Tierney. 2021. Efficient active search for combinatorial optimization problems. *arXiv preprint arXiv:2106.05126* (2021).

[20] André Hottung and Kevin Tierney. 2020. Neural large neighborhood search for the capacitated vehicle routing problem. In *ECAI 2020*. IOS Press, 443–450.

[21] André Hottung, Paula Wong-Chung, and Kevin Tierney. 2025. Neural Deconstruction Search for Vehicle Routing Problems. *arXiv preprint arXiv:2501.03715* (2025).

[22] Qingchun Hou, Jingwei Yang, Yiqiang Su, Xiaoqing Wang, and Yuming Deng. 2023. Generalize learned heuristics to solve large-scale vehicle routing problems in real-time. In *The Eleventh International Conference on Learning Representations*.

[23] J. Stuart Hunter. 1986. The Exponentially Weighted Moving Average. *Journal of Quality Technology* 18, 4 (1986), 203–210. https://doi.org/10.1080/00224065.1986.11979014 arXiv:https://doi.org/10.1080/00224065.1986.11979014

[24] Xia Jiang, Yaoxin Wu, Yuan Wang, and Yingqian Zhang. 2024. Unco: Towards unifying neural combinatorial optimization through large language model. *arXiv preprint arXiv:2408.12214* (2024).

[25] Yuan Jiang, Zhiguang Cao, Yaoxin Wu, Wen Song, and Jie Zhang. 2023. Ensemble-based Deep Reinforcement Learning for Vehicle Routing Problems under Distribution Shift. In *Thirty-seventh Conference on Neural Information Processing Systems*.

[26] Yan Jin, Yuandong Ding, Xuanhao Pan, Kun He, Li Zhao, Tao Qin, Lei Song, and Jiang Bian. 2023. Pointerformer: Deep Reinforced Multi-Pointer Transformer for the Traveling Salesman Problem. *arXiv preprint arXiv:2304.09407* (2023).

[27] David S Johnson and Lyle A McGeoch. 1997. The traveling salesman problem: a case study. *Local search in combinatorial optimization* (1997), 215–310.

[28] Minsu Kim, Sanghyeok Choi, Jiwoo Son, Hyeonah Kim, Jinkyoo Park, and Yoshua Bengio. 2024. Ant Colony Sampling with GFlowNets for Combinatorial Optimization. *arXiv preprint arXiv:2403.07041* (2024).

[29] Minsu Kim, Junyoung Park, and Jinkyoo Park. 2022. Sym-nco: Leveraging symmetricity for neural combinatorial optimization. *Advances in Neural Information Processing Systems* 35 (2022), 1936–1949.

[30] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[31] Detian Kong, Yining Ma, Zhiguang Cao, Tianshu Yu, and Jianhua Xiao. 2024. Efficient Neural Collaborative Search for Pickup and Delivery Problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2024).

[32] Wouter Kool, Herke Van Hoof, and Max Welling. 2018. Attention, learn to solve routing problems! *arXiv preprint arXiv:1803.08475* (2018).

[33] Yeong-Dae Kwon, Jinho Choo, Byoungjip Kim, Iljoo Yoon, Youngjune Gwon, and Seungjai Min. 2020. Pomo: Policy optimization with multiple optima for reinforcement learning. *Advances in Neural Information Processing Systems* 33 (2020), 21188–21198.

[34] Han Li, Fei Liu, Zhi Zheng, Yu Zhang, and Zhenkun Wang. 2024. CaDA: Cross-Problem Routing Solver with Constraint-Aware Dual-Attention. *arXiv preprint arXiv:2412.00346* (2024).

[35] Jingwen Li, Yining Ma, Ruize Gao, Zhiguang Cao, Andrew Lim, Wen Song, and Jie Zhang. 2022. Deep reinforcement learning for solving the heterogeneous capacitated vehicle routing problem. *IEEE Transactions on Cybernetics* 52, 12 (2022), 13572–13585.

[36] Jingwen Li, Liang Xin, Zhiguang Cao, Andrew Lim, Wen Song, and Jie Zhang. 2021. Heterogeneous attentions for solving pickup and delivery problem via deep reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems* 23, 3 (2021), 2306–2315.

[37] Sirui Li, Zhongxia Yan, and Cathy Wu. 2021. Learning to delegate for large-scale vehicle routing. *Advances in Neural Information Processing Systems* 34 (2021), 26198–26211.

[38] Yang Li, Jinpei Guo, Runzhong Wang, and Junchi Yan. 2024. From distribution learning in training to gradient search in testing for combinatorial optimization. *Advances in Neural Information Processing Systems* 36 (2024).

[39] Yifu Li, Chenhao Zhou, Peixue Yuan, and Thi Tu Anh Ngo. 2023. Experience-based territory planning and driver assignment with predicted demand and driver present condition. *Transportation research part E: logistics and transportation review* 171 (2023), 103036.

[40] Xi Lin, Zhiyuan Yang, and Qingfu Zhang. 2022. Pareto set learning for neural multi-objective combinatorial optimization. *arXiv preprint arXiv:2203.15386* (2022).

[41] Zhuoyi Lin, Yaoxin Wu, Bangjian Zhou, Zhiguang Cao, Wen Song, Yingqian Zhang, and Senthilnath Jayavelu. 2024. Cross-Problem Learning for Solving Vehicle Routing Problems. *IJCAI* (2024).

[42] Fei Liu, Xi Lin, Zhenkun Wang, Qingfu Zhang, Tong Xialiang, and Mingxuan Yuan. 2024. Multi-task learning for routing problem with cross-problem zero-shot generalization. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1898–1908.

[43] Fei Liu, Tong Xialiang, Mingxuan Yuan, Xi Lin, Fu Luo, Zhenkun Wang, Zhichao Lu, and Qingfu Zhang. 2024. Evolution of Heuristics: Towards Efficient Automatic Algorithm Design Using Large Language Model. In *Forty-first International Conference on Machine Learning*.

[44] Qidong Liu, Chaoyue Liu, Shaoyao Niu, Cheng Long, Jie Zhang, and Mingliang Xu. 2024. 2D-Ptr: 2D Array Pointer Network for Solving the Heterogeneous Capacitated Vehicle Routing Problem. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems*. 1238–1246.

[45] Manuel Lozano, Daniel Molina, and Francisco Herrera. 2011. Editorial scalability of evolutionary algorithms and other metaheuristics for large-scale continuous optimization problems. *Soft computing* 15 (2011), 2085–2087.

[46] Fu Luo, Xi Lin, Fei Liu, Qingfu Zhang, and Zhenkun Wang. 2023. Neural combinatorial optimization with heavy decoder: Toward large scale generalization. *Advances in Neural Information Processing Systems* 36 (2023), 8845–8864.

[47] Yining Ma, Zhiguang Cao, and Yeow Meng Chee. 2024. Learning to search feasible and infeasible regions of routing problems with flexible neural k-opt. *Advances in Neural Information Processing Systems* 36 (2024).

[48] Yining Ma, Jingwen Li, Zhiguang Cao, Wen Song, Le Zhang, Zhenghua Chen, and Jing Tang. 2021. Learning to iteratively solve routing problems with dual-aspect collaborative transformer. *Advances in Neural Information Processing Systems* 34

(2021), 11096–11107.
[49] Christos H Papadimitriou and Kenneth Steiglitz. 1998. *Combinatorial optimization: algorithms and complexity*. Courier Corporation.
[50] Junyoung Park, Changhyun Kwon, and Jinkyoo Park. 2023. Learn to Solve the Min-max Multiple Traveling Salesmen Problem with Reinforcement Learning. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*. 878–886.
[51] Laurent Perron and Vincent Furnon. 2023. OR-Tools. Google.
[52] Vu Tuan Dat Pham, Long Doan, and Thi Thanh Binh Huynh. 2025. HSEvo: Elevating Automatic Heuristic Design with Diversity-Driven Harmony Search and Genetic Algorithm Using LLMs. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*. Association for the Advancement of Artificial Intelligence (AAAI).
[53] Wei Qin, Zilong Zhuang, Zizhao Huang, and Haozhe Huang. 2021. A novel reinforcement learning-based hyper-heuristic for heterogeneous vehicle routing problem. *Computers & Industrial Engineering* 156 (2021), 107252.
[54] Jiwoo Son, Minsu Kim, Sanghyeok Choi, Hyeonah Kim, and Jinkyoo Park. 2024. Equity-Transformer: Solving NP-Hard Min-Max Routing Problems as Sequential Generation with Equity Context. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 20265–20273.
[55] Jiwoo Son, Minsu Kim, Hyeonah Kim, and Jinkyoo Park. 2023. Meta-sage: Scale meta-learning scheduled adaptation with guided exploration for mitigating scale shift on combinatorial optimization. In *International Conference on Machine Learning*. PMLR, 32194–32210.
[56] Zhiqing Sun, Zhuohan Li, Haoqing Wang, Di He, Zi Lin, and Zhihong Deng. 2019. Fast structured decoding for sequence models. *Advances in Neural Information Processing Systems* 32 (2019).
[57] Zhiqing Sun and Yiming Yang. 2023. Difusco: Graph-based diffusion solvers for combinatorial optimization. In *Thirty-seventh Conference on Neural Information Processing Systems*.
[58] Team Locus. 2020. Zone-Based Routing is the Need of the Hour. https://www.locus.sh/blog/zone-based-routing-is-the-need-of-the-hour. *Locus Blog* (12 May 2020). https://www.locus.sh/blog/zone-based-routing-is-the-need-of-the-hour Accessed: 2024-10-17.
[59] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).
[60] José Manuel Vera and Andres G Abad. 2019. Deep reinforcement learning for routing a heterogeneous fleet of vehicles. In *2019 IEEE Latin American Conference on Computational Intelligence (LA-CCI)*. IEEE, 1–6.
[61] Thibaut Vidal. 2022. Hybrid genetic search for the CVRP: Open-source implementation and SWAP* neighborhood. *Computers & Operations Research* 140 (2022), 105643.
[62] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. *Advances in neural information processing systems* 28 (2015).
[63] Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* 8 (1992), 229–256.
[64] Niels A Wouda, Leon Lan, and Wouter Kool. 2024. PyVRP: A high-performance VRP solver package. *INFORMS Journal on Computing* (2024).
[65] Yubin Xiao, Di Wang, Huanhuan Chen, Boyang Li, Wei Pang, Xuan Wu, Hao Li, Dong Xu, Yanchun Liang, and You Zhou. 2023. Reinforcement learning-based non-autoregressive solver for traveling salesman problems. *arXiv preprint arXiv:2308.00560* (2023).
[66] Zhongxia Yan and Cathy Wu. 2024. Neural Neighborhood Search for Multi-agent Path Finding. In *The Twelfth International Conference on Learning Representations*.
[67] Yunhao Yang and Andrew Whinston. 2023. A survey on reinforcement learning for combinatorial optimization. In *2023 IEEE World Conference on Applied Intelligence and Computing (AIC)*. IEEE, 131–136.
[68] Haoran Ye, Jiarui Wang, Zhiguang Cao, Federico Berto, Chuanbo Hua, Haeyeon Kim, Jinkyoo Park, and Guojie Song. 2024. ReEvo: Large Language Models as Hyper-Heuristics with Reflective Evolution. In *Advances in Neural Information Processing Systems*. https://github.com/ai4co/reevo.
[69] Haoran Ye, Jiarui Wang, Zhiguang Cao, Helan Liang, and Yong Li. 2023. DeepACO: Neural-enhanced Ant Systems for Combinatorial Optimization. In *Advances in Neural Information Processing Systems*.
[70] Haoran Ye, Jiarui Wang, Helan Liang, Zhiguang Cao, Yong Li, and Fanzhang Li. 2024. GLOP: Learning Global Partition and Local Construction for Solving Large-scale Routing Problems in Real-time. In *Proceedings of the AAAI Conference on Artificial Intelligence*.
[71] Ke Zhang, Fang He, Zhengchao Zhang, Xi Lin, and Meng Li. 2020. Multi-vehicle routing problems with soft time windows: A multi-agent reinforcement learning approach. *Transportation Research Part C: Emerging Technologies* 121 (2020), 102861.
[72] Zhi Zheng, Shunyu Yao, Zhenkun Wang, Tong Xialiang, Mingxuan Yuan, and Ke Tang. 2024. DPN: Decoupling Partition and Navigation for Neural Solvers of Min-max Vehicle Routing Problems. In *Forty-first International Conference on Machine Learning*. https://openreview.net/forum?id=ar174skI9u
[73] Zhi Zheng, Changliang Zhou, Tong Xialiang, Mingxuan Yuan, and Zhenkun Wang. 2024. UDC: A unified neural divide-and-conquer framework for large-scale combinatorial optimization problems. *arXiv preprint arXiv:2407.00312* (2024).
[74] Hongsheng Zhong, Randolph W Hall, and Maged Dessouky. 2007. Territory planning and vehicle dispatching with driver learning. *Transportation Science* 41, 1 (2007), 74–89.
[75] Jianan Zhou, Zhiguang Cao, Yaoxin Wu, Wen Song, Yining Ma, Jie Zhang, and Chi Xu. 2024. MVMoE: Multi-Task Vehicle Routing Solver with Mixture-of-Experts. *arXiv preprint arXiv:2405.01029* (2024).
[76] Jianan Zhou, Yaoxin Wu, Zhiguang Cao, Wen Song, Jie Zhang, and Zhenghua Chen. 2023. Learning large neighborhood search for vehicle routing in airport ground handling. *IEEE Transactions on Knowledge and Data Engineering* (2023).
[77] Jianan Zhou, Yaoxin Wu, Zhiguang Cao, Wen Song, Jie Zhang, and Zhiqi Shen. 2024. Collaboration! Towards Robust Neural Methods for Routing Problems. *arXiv preprint arXiv:2410.04968* (2024).
[78] Zefang Zong, Meng Zheng, Yong Li, and Depeng Jin. 2022. Mapdp: Cooperative multi-agent reinforcement learning to solve pickup and delivery problems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 9980–9988.