A Mean Field Reinforcement Learning Approach to Large-Scale Vehicle Routing Problems

Anonymous authors
Paper under double-blind review

Abstract

Solving large-scale vehicle routing problems (VRPs) is NP-hard and poses a computational challenge in numerous applications such as logistics. Meanwhile, mean field control (MFC) provides a tractable and rigorous approach to controlling many agents. We provide a solution to pickup-and-delivery VRPs via scalable MFC. In combination with reinforcement learning (RL) and clustering, our MFC approach efficiently scales to large-scale VRPs. We perform a theoretical analysis of our MFC-based approximation, giving convergence results for large VRP instances and error bounds for clustering-based approximations. We verify our algorithms on different datasets and compare them against solutions such as OR-Tools, PyVRP and heuristics, showing scalability in terms of speed for mean-field methods, for the first time in discrete optimization. Overall, our work establishes a novel synthesis of MFC-based RL techniques, vehicle routing problems and clustering approximations, to solve a hard discrete optimization problem of practical use in a scalable way.

1 Introduction

Solving large-scale vehicle routing problems (VRPs) is an area of great practical interest in operations research. There are numerous applications, ranging from dial-a-ride (Cordeau & Laporte, 2007; Ho et al., 2018) over shared use of autonomous vehicles (Narayanan et al., 2020; Lin et al., 2012), or delivery logistics (Bortfeldt & Yi, 2020; Wang et al., 2021; Arunapuram et al., 2003; Zhang & Lygeros, 2023) to agriculture (Bochtis & Sørensen, 2009; 2010). Due to the large number of vehicles and customers, scalable solutions to VRPs remain an active area of research (Konstantakopoulos et al., 2022; Zhang et al., 2022).

Motivated by complex applications with many vehicles and tasks, there has been growing interest in solving large-scale VRPs via machine learning. Learning approaches for finding scalable VRP solutions are manifold and differ in their conceptual design. Some learning techniques split the initial VRP into smaller and more tractable subproblems and delegate their solution to subsolvers (Li et al., 2021b) to reduce the overall computational complexity. Other research (Lei et al., 2022) uses transformers to increase the scalability of learning algorithms. For recent overviews on learning VRPs, see Li et al. (2022); Bogyrbayeva et al. (2024).

1.1 Contribution

Scaling solutions for very large NP-hard problems can be difficult. For this reason, our approach is based on mean field control (MFC) (Fornasier & Solombrino, 2014; Bensoussan et al., 2013) as the cooperative form of mean field games (Lasry & Lions, 2007; Saldi et al., 2018). This will allow our method to scale to arbitrarily large problem sizes, with constant problem complexity in the limiting problem, and linear complexity in the number of agents / objectives for the actual finite problem, by sampling the agent routes. The idea of MFC is to abstract a large number of small, indistinguishable agents – in our case vehicles – into their *mean field* (MF) probability distribution. Working with the MF distribution entails both solid theoretical properties as well as reduced computational complexity which, in turn, paves the way for scalable learning algorithms. In contrast to existing works, our MF-based solution allows to solve arbitrarily large problems at constant computational complexity. We propose MFC-based reinforcement learning (RL) for solving large-scale VRPs with pickup and delivery, where our learning approach scales to arbitrarily large numbers of vehicles and deliveries.

Our contributions are summarized as: (i) We present a first formulation of vehicle routing – a hard discrete optimization problem – as MFC & RL; (ii) We give general and special fine-tuned algorithms for solving large-scale vehicle routing problems, considering both general pretraining as well as fine-tuning to problems; (iii) We prove convergence to the MF limit for the case of large-scale VRPs with many vehicles and locations. Moreover, we show approximate probable optimality of MFC solutions and give error bounds for general metric spaces with clustering approximations; (iv) We empirically evaluate by comparing against OR-Tools and heuristics, showing potential generalization capabilities and speeds of MFC for different large-scale settings, where existing methods for discrete optimization can be expensive.

1.2 Related Work

In comparison to our work, a majority of recent related work does not consider pickup-delivery problems. We also find that few other RL works allow sub-minute training times, which we provide. Moreover, very few works consider large-scale VRPs with thousands of nodes and vehicles.

For example, (Zhou et al., 2024) considers different variants of VRPs, but no pickup-delivery, focusing on multi-task settings with generalization and presented for up to 100 nodes. Similar smaller settings are found in the following works, such as DPN Zheng et al. (2024), which considers min-max VRPs via sequential RL-based partition and navigation methods. In contrast, we consider near-arbitrarily large VRP. In Liu et al. (2024), VRP with heterogeneous capacities are considered via RL-based solutions, and evaluated for small fleet and customer sizes of up to 7 and 100. As for other recent work, Wu et al. (2024) studies multi-objective VRPs using collaborative RL, Xiao et al. (2024) demonstrates fast transformer-based solutions, and Kikuta et al. (2024) considers smaller variants of VRP for electric vehicles. On the other hand, Min et al. (2024) proposes graph-learning methods for traveling salesman problems (TSP), and Jiang et al. (2023) focuses on out-of-distribution generalization. Furthermore, Xia et al. (2024) explores heuristics for the special case of TSPs, highlighting the scalability issues with current solutions and proposing scalable Monte Carlo tree search. Our contribution similarly is a method that provides an automatically scaling solution for another special case of VRPs.

Lastly, most related to our setting, also known as pickup-delivery problems (PDP), Li et al. (2021a) and Zong et al. (2022) solve single-agent and multi-agent PDP using deep attention-based (multi-agent) reinforcement learning methods. As for literature specifically on large-scale VRP, recent works include meta-learning and reinforcement learning for combinatorial optimization (Qiu et al., 2022), graph-based diffusion solvers (Sun & Yang, 2023), and methods that generalize pre-trained models or learned heuristics to arbitrarily large TSP or VRP instances (Fu et al., 2021; Hou et al., 2023). Other neural combinatorial optimization approaches also aim for large-scale generalization via encoder-decoder architectures (Luo et al., 2023). In contrast, our MFC-based RL method provides a methodology for reducing the multi-agent RL (MARL) problem to a single-agent RL problem of constant complexity, regardless of the number of agents.

Modelling agent MFs is done in analogy to general MFC (Angiuli et al., 2022; Carmona et al., 2023; Cui et al., 2021), with policy gradients (Angiuli et al., 2023; Carmona et al., 2019; Cui et al., 2024) as a solution for the resulting high-dimensional RL problem. However, VRPs require a generalized formulation with *delivery* distributions: We first suitably rewrite between VRP and multi-agent control with MFs. Moreover, we have two MFs that cannot be modelled by existing MFC-based MARL, as the two MFs of agents & objects create a double limit, which we resolve by introducing the ratio between objects and agents. Importantly, we also obtain probable approximate optimality of any random realization in MFVRP that is not obtained from results in general MFC, via embedding finite MFVRP into the limiting system. See also Section 2.5 for a discussion.

2 Mathematical Framework

In this section, we connect VRP with MARL and MFC. Proofs for theoretical results may be found in the Appendix.

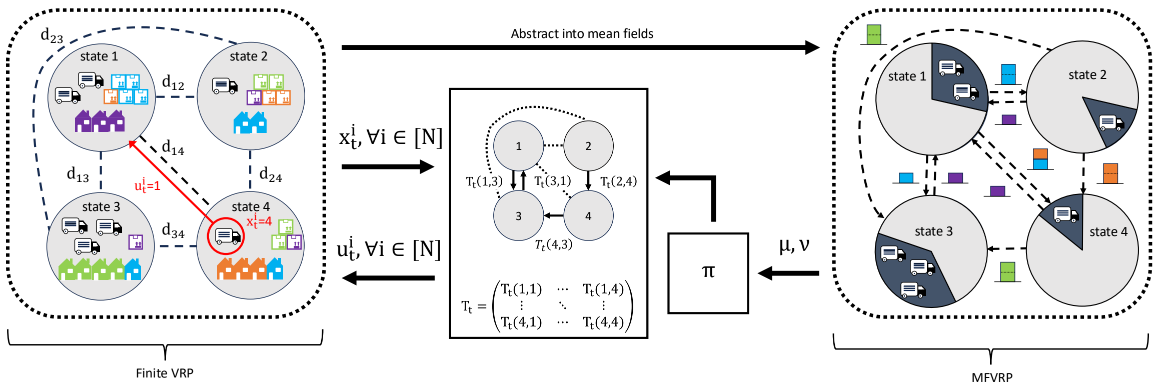

Figure 1: Simplified overview of the relation between finite VRP and MFVRP system. Examplary system with four states, seven trucks and 14 packages and destinations. Orange (blue / green / purple) packages should be delivered to orange (blue / green / purple) destinations / houses. In the case of Euclidean locations, each location is assigned to one state via clustering, see Section 3.

Notation: For any integers k, l, let $[k] \coloneqq \{1, 2, \dots, k\}$ and $[k] + l \coloneqq \{l+1, \dots, l+k\}$. For any metric space A, let $\mathcal{B}_1(A)$ denote Borel measures bounded by 1 over A, with probability measures $\mathcal{P}(A) \subseteq \mathcal{B}_1(A)$.

2.1 Capacitated VRP with Pickup and Delivery

As a finite problem of interest, one may consider vehicle routing problems with pickup and delivery. We consider a large number of N vehicles and M objects, each of which needs to be transported from one position to another in some metric space $\mathcal{X}$. The methodology applies to any metric space. For example, $\mathcal{X}$ can be a subset of Euclidean space, or it could be nodes on a road graph with appropriate node distances. We index vehicles and objects by indices in $\mathcal{N} \coloneqq [N]$ and $\mathcal{M} \coloneqq [M] + N$ respectively. We denote distances between objects or vehicles at $x, y \in \mathcal{X}$ as $d(x, y)$. For example, we consider Euclidean spaces $\mathcal{X} \subseteq \mathbb{R}^2$ with the Euclidean distance $d(x, y) = \|x - y\|_2$ as well as finite spaces embedded into the Euclidean space. More generally, we also include positions on a road network or graph. Each object or transport mission $j \in \mathcal{M}$ is specified by its source $s_j \in \mathcal{X}$ and destination $d_j \in \mathcal{X}$, while each vehicle $i \in \mathcal{N}$ starts at an initial depot $s_i \in \mathcal{X}$.

For simplicity of exposition, consider vehicles with carrying capacity of one object. A more general setting with larger capacities can be considered analogously, see also Section 2.5, though the state-action complexity will be exponential in the transport capacity. The goal is to optimize *which* vehicles transport *which* objects in *which* order, such that an overall global transport cost is minimized. Formally, the overall cost is given as the total distances travelled by all vehicles. In other words, we define the set of all indices $\mathcal{V} \coloneqq [N + M] = \mathcal{N} \cup \mathcal{M}$ and optimize the *objective* $J \coloneqq \sum_{k \in \mathcal{N}} \sum_{i,j \in \mathcal{V}} d_{ij} b_{ij}^k$ for binary decision variables $\mathbf{b} \coloneqq (b_{ij}^k)_{a \in \mathcal{N}, i,j \in \mathcal{V}}$, where b_{ij}^k is one whenever vehicle k moves from depot or transport mission i to j with induced travel cost between depots or mission sources $i, j \in \mathcal{V}$

$$d_{ij} \coloneqq d(s_i, d_i) + d(d_i, s_j), \tag{1}$$

where the depot's source and destination are equal to the depot's position.

We consider a standard tour length constraint $C \in \mathbb{N}$ as the maximum number of missions to be performed by any single vehicle due to time or energy constraints. At least for a carrying capacity of one, the above problem can also be solved via capacitated VRP (CVRP) by considering each mission as one location, but with asymmetric non-Euclidean distances according to (1) due to differing pickup and delivery locations. Solving the optimization is well known to be NP-hard. We introduce MF-approximations for the important case of many missions and vehicles.

2.2 VRP as Mean-Field Multi-Agent Control

We consider a sequential solution of VRPs formulated as a multi-agent control problem, i.e. performing one delivery per vehicle in each "time step" $t \in \mathbb{N}$ (decision epoch, not real time). The multi-agent control formulation of VRP is obtained by assuming an underlying distribution of vehicle and object locations, and introducing both current vehicle states and remaining transportation missions. That is, we assume each location x_0^i of vehicle $i \in \mathcal{N}$ – the vehicle state – is sampled from some initial distribution $\mu_0 \in \mathcal{P}(\mathcal{X})$, $x_0^i \sim \mu_0$. Analogously, the source-destination tuples (s_j, d_j) of objects $j \in [M]$ are sampled from an object distribution $\nu_0 \in \mathcal{P}(\mathcal{X}^2)$, $(s_j, d_j) \sim \nu_0$. Consider a fixed ratio $R = M/N$ of objects per vehicle to transport. Then, due to indiscernibility of objects and vehicles at the same locations, the entire information of a VRP is contained in the parameter R and the empirical distribution of vehicle states $\mu_0^N = \frac{1}{N}\sum_i \delta_{x_0^i}$ and source and destination locations $\nu_0^N = \frac{1}{M}\sum_j \delta_{(s_j,d_j)}$ – the empirical MFs at time zero. Here, the MFs can be understood as "histograms" over $\mathcal{X}$ and $\mathcal{X}^2$ respectively, with Dirac measures δ. For a visualization of the scenario, see the left side of Fig. 1, which is the original VRP and is transformed into a problem of moving around fractions of vehicles and objects.

The current intermediate state of a VRP solution during execution at any time t is given by counting the remaining objects to be transported at each source and destination, as well as the current positions of vehicles. Indeed, the current state is contained in $\nu_t^N = \frac{1}{M}\sum_{j \text{ not transported yet}} \delta_{(s_j,d_j)}$ and $\mu_t^N = \frac{1}{N}\sum_i \delta_{x_t^i}$. Here, the states of vehicles x_t^i are moved by vehicle actions $u_t^i \in \mathcal{X}$ to $x_{t+1}^i = u_t^i$. When moving, each vehicle i transports a random object from x_t^i to u_t^i, if one exists. The reward is then given by the negative distance travelled by all vehicles plus a reward for each delivered package, $r_t^N = c_\mathrm{d} \cdot \#_t - \sum_i d(x_t^i, u_t^i)$, where $\#_t$ is the number of delivered objects in epoch t and constant $c_\mathrm{d} > \frac{2}{M}\operatorname{diam}(\mathcal{X})$ with the following reason: The choice of c_d ensures that an optimal undiscounted control maximizing $J^N = \mathbb{E}\left[\sum_t r_t^N\right]$ is also an optimal VRP solution: For optimality, vehicles always deliver all objects to obtain positive rewards by $c_\mathrm{d} > \frac{2}{M}\operatorname{diam}(\mathcal{X})$ instead of avoiding deliveries, while also minimizing travelled distances. Analogously, we add a final cost for moving back to the depot and award a larger final reward r_bonus whenever all objects have been transported, and a cost of c_miss times fraction of objects not delivered at the end of an episode. Here, we terminate an episode when all objects are delivered or after reaching an episode time horizon of $t_\mathrm{max} = C$ to respect the maximum tour length constraint $C \in \mathbb{N}$.

2.3 Mean Field VRP

To obtain tractable MFVRP, in the infinite vehicle-object limit $N \to \infty$ with $M = R \cdot N$, $R > 0$ the initial empirical MFs are given by μ_0 and ν_0 via the law of large numbers. It suffices to describe the system by the limiting deterministic MFs $\mu_t \in \mathcal{P}(\mathcal{X})$, $\nu_t \in \mathcal{B}_1(\mathcal{X})$ at all times t. Here, at each time t, the MFVRP action is one transportation by each of the infinitely many vehicles. That is, we assign all vehicles from their current to their next location and transport objects on the way. In general, the action at time t is therefore a transport map T_t, a Markov kernel on $\mathcal{X}$.

For finite $\mathcal{X}$, the action can be numerically represented as a transition matrix $T_t \colon \mathcal{X} \times \mathcal{X} \to [0,1]$ with $\sum_d T_t(s,d) = 1$ for all $s \in \mathcal{X}$, encoding which fraction $T_t(s,d)$ of vehicles at $s \in \mathcal{X}$ will move to $d \in \mathcal{X}$. Since the ratio between vehicles and objects is taken as some constant $R > 0$, the fraction of remaining transportation tasks decreases according to

$$\nu_{t+1}(s,d) = \max(0, \nu_t(s,d) - \mu_t(s)T_t(s,d)/R) \tag{2}$$

while the MF of vehicles evolves as

$$\mu_{t+1}(d) = \sum_{s \in \mathcal{X}} \mu_t(s)T_t(s,d). \tag{3}$$

Hence, the MFC problem is a single-agent Markov decision problem, with states (μ_t, ν_t) and actions T_t, where the reward is given by delivered objects minus distance travelled (on average per agent, which is equivalent to optimizing the total travelled distance)

$$r_t = \sum_{s,d \in \mathcal{X}} c_{\mathrm{d}} \min(\nu_t(s,d), \mu_t(s) T_t(s,d)/R) - d(s,d)\mu_t(s) T_t(s,d). \tag{4}$$

In other words, we learn an "upper-level" MFC policy $\hat{\pi}$ via RL, such that $T_t \sim \hat{\pi}(T_t \mid \mu_t, \nu_t)$. See also Fig. 1 for a visualization. Final rewards and costs are handled analogously. We choose $c_{\mathrm{d}} > 2\,\mathrm{diam}(\mathcal{X})$ to ensure optimality of delivering all objects instead of waiting until termination.

2.4 Theoretical Guarantees

To show optimality of MFC in VRP, the three introduced models are rigorously analyzed. We first sketch how the VRP objective is exactly given by the finite MARL system objective, and how the MARL objective is approximately solved by the limiting MFVRP MDP.

The former connection is made by noticing that vehicle trajectories in the MARL system $(x_0^i, u_0^i, x_1^i, u_1^i, x_2^i, \ldots)_{i \in \mathcal{N}}$ correspond to a solution of the VRP. Any objects not transported by the trajectory are handled by postprocessing, by transporting objects by random available vehicles at the end. Assuming optimal behavior of vehicles in the MARL problem and therefore that all objects are transported, the cost of the VRP as total distance travelled is exactly given by the negative sum of rewards J, up to a constant episodic reward for delivering all objects. Therefore, optimally solving the MARL problem optimally solves the VRP. In other words, for any vehicle $i \in \mathcal{N}$, we assign $b_{jk}^i = 1$ whenever it transports objects j and then k, as well as $b_{ij}^i = 1$ for moving from depot i to the first object j and $b_{ji}^i = 1$ for the last object, thereby removing any unnecessary in-between movements of each vehicle that does not transport objects.

The latter connection is shown in the following. A MFC solution solves the infinite-vehicle model, and approximates optimal solutions in the finite MARL system by allowing the sampling from T_t – a random realization of the deterministic MFC system (2) and (3) constitutes a MARL solution and therefore also VRP solution. Vehicles take actions according to deterministic, or hence equivalently constant, $\pi_t \equiv T_t$ at all times $t \leq t_{\max}$ up to termination at time $t_{\max}$, after which we randomly transport any remaining objects. For a visualization see Fig. 1.

An error analysis of finite VRP gives us estimates of the asymptotic error of MFC assuming an infinitude of vehicles and objects, instead of finitely many. We then learn a deterministic MFC policy $\hat{\pi}$, e.g., after the convergence of stochastic policy gradient methods (Schulman et al., 2017), or by using deterministic policy gradients (Silver et al., 2014), which can be provably near-optimal in large finite problems. The proofs can be found in the Appendix.

Theorem 2.1 (Propagation of Chaos). *Let $\mathcal{X}$ be finite and consider some deterministic $\hat{\pi}$. Then, at any time $t \in \mathbb{N}$,*

$$\|\mu_t^N - \mu_t\|_\infty \to 0 \quad \text{and} \quad \|\nu_t^N - \nu_t\|_\infty \to 0$$

in probability as $N \to \infty$.

Corollary 2.2 (Probable Approximate Optimality). *Let $\mathcal{X}$ be finite. Then, an optimal MFC solution is probably approximately optimal in the finite N systems, i.e., for any $\varepsilon, \delta > 0$, there is $N' \in \mathbb{N}$ s.t. for any $N \geq N'$, the sampled MARL solution is ε-optimal with probability at least $1 - \delta$.*

The advantage is constant RL problem size regardless of number of agents / clients around each cluster.

2.5 Discussion

The optimal MFC solution in Corollary 2.2 requires no additional assumptions because we may replace it by an optimal *constant* MFC policy, leading to deterministic T_t. This works only for our MFVRP setting and not for general MFC, since we show that any optimal VRP objective is also obtained exactly in the limiting MFC by a suitable MFC policy, see the third inequality in the proof of Corollary 2.2.

Generality. Even for unit-capacity, MFC-based RL is well-justified by virtue of scaling to arbitrarily many agents. Moreover, we can naturally generalize the system to higher-than-unit-capacity problems by exponentiating the state space of agents to model picked-up objects, and generalizing transition matrices T_t with additional probabilities of picking up objects with particular destination, i.e. the state space is exponential in the number of parcels per vehicle. The sequential formulation can also allow dynamic variants of VRP which are not addressed by standard VRP solvers, e.g., with random arrival of new missions, or stochastic failure of vehicles. MFVRP can also model heterogeneous vehicles, such as heterogeneous fleets with different capacities or speeds, by duplicating the state space for each separate type of vehicle (i.e. integrating the type into the state).

The methodology also applies to any metric space including non-Euclidean such as on graphs, in practice by using generalized clustering methods. Finally, in case of uncountable $\mathcal{X}$, such as a convex subset of $\mathbb{R}^2$, transport maps T_t are replacable by couplings on $\mathcal{X} \times \mathcal{X}$, but the resulting MFVRP MDP is of infinite dimension. Instead, for algorithmic tractability, in the following we consider a discretization-based approach, with guarantees of near optimality for sufficiently fine discretization.

Limitations. We note that MFC is not without limitations for solving discrete optimization problems, which are discussed in the following. In particular, MFC remains hard to apply for non-pickup-delivery VRPs: First, the special case of multiple traveling salesmen problem (mTSP) under MFC would boil down to a clustered version of standard mTSP, since distances between points in the same cluster are zero so they are immediately served by a single same agent. While this case can be considered, there is therefore no methodological novelty in applying MFC to mTSP beyond clustering all the locations and replacing them with their centroid. Meanwhile, constant finite capacities without pickup-delivery structure disappear in the limit of infinite agents, and are therefore also difficult to model. For scaling capacities, we obtain back a standard VRP with finite number of agents and not the RL problem we discuss. Meanwhile, time constraints are hard to model for MFC. Heterogeneous-capacity VRP may be considered when considering non-unit capacities, but we point out that the capacity scales the MFC problem complexity exponentially.

3 Mean Field Vehicle Routing RL

We present our algorithmic solution, visualized in Fig. 1. The idea is to solve the MFVRP as a single-agent MDP, via single-agent RL. At the same time, the implementation is multi-agent RL if we directly learn on the finite VRPs. Indeed, this gives us two approaches of implementing algorithms for MFVRP. But first, we describe the clustering procedure required for handling general metric $\mathcal{X}$, with a brief theoretical error analysis as a motivation.

3.1 Discretization and Clustering

For locations on a road network or graph, a finite state space $\mathcal{X}$ suffices and we can avoid infinite-dimensional MFC state-actions, but in general this is not the case. Therefore, one can use Voronoi partitions to trade-off between accuracy of MF approximation and size of MFC problem. This is particularly useful when the objects and vehicles are clustered around a few most relevant locations such as cities, which are not located on a grid. Consider $k \in [K]$ cell centroids $c_k \in \mathcal{X}$. Each centroid induces a cell $C_k \coloneqq \{x \in \mathcal{X} \mid d(x, c_k) \in \min_j d(x, c_j)\} \subseteq \mathcal{X}$ consisting of all points in $\mathcal{X}$ which are closest to that particular centroid out of all centroids. Then, to minimize the error stemming from MFC approximation, we may use the empirical histogram with cells instead of bins, giving us the approximate MF as fractions of objects to be delivered from one cell to another. As a result, we produce an assignment $\rho \colon \mathcal{M} \to [K]^2$ such that each object i is assigned to the centroids closest to its source and destination location, $\rho(i) \in \arg\min_k d(s_i, c_k) \times \arg\min_l d(d_i, c_l)$, where $\times$ denotes the Cartesian product. In practice, for simplicity we use k-means to produce a clustering of all source and destination locations, assigning each location to one of the clusters. In general one can use any clustering method, e.g., instead minimizing the non-squared Euclidean distances via k-medoids (Kaufman & Rousseeuw, 2008). See Fig. 2 for a visualization.

We can then bound the maximum error of the MFC approximation by the maximum sizes of each cluster.

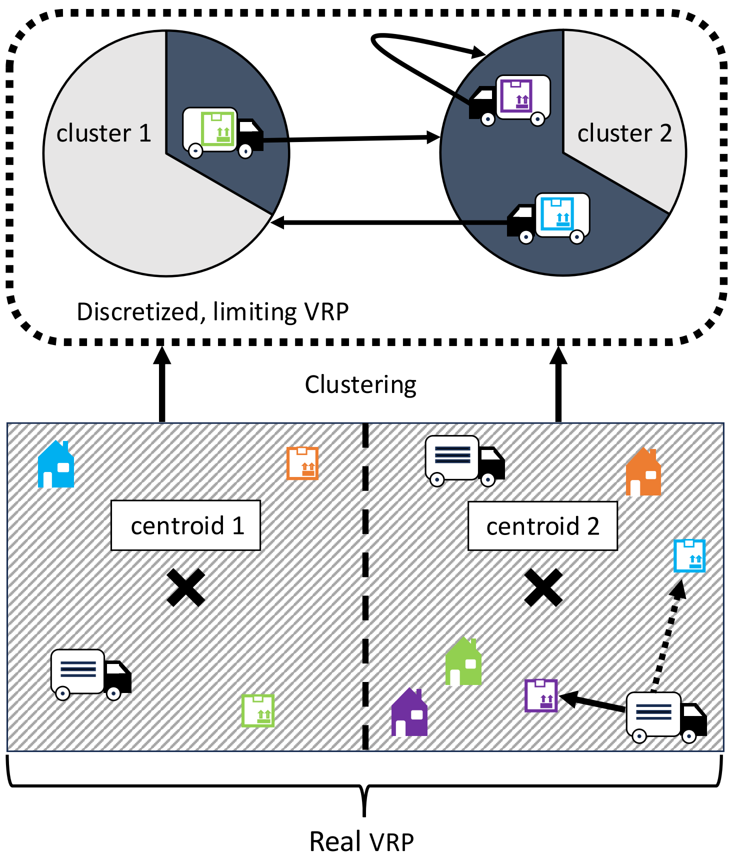

Figure 2: Simplified overview of the clustering approach for Euclidean VRP in a toy example. Observe the correspondence between limiting equations and real VRP, with clustering-based discretization.

Theorem 3.1. *An optimal discretized MFC solution is probably approximately optimal in the finite N systems, i.e., for any $\varepsilon, \delta > 0$, there is $N' \in \mathbb{N}$ s.t. the sampled MARL solution for any $N \geq N'$ is $(\varepsilon + \varepsilon')$-optimal with probability at least $1 - \delta$, with additional discretization error*

$$\varepsilon' \leq (4M + 2N) \max_k \operatorname{diam}(C_k).$$

Our methodology is hence near-optimal whenever RL converges to an optimal MFC solution, for enough agents and good discretization. Note that the time complexity becomes linear in the number of agents or objects N, M, because we only need to count how many fall into each bin. In contrast, complexity is shifted to quadratic in the number of clusters K. Thus, as shown in the experiments, we can achieve high scalability in N, M.

3.2 Algorithm and Fine-Tuning

With the problem of general representation solved, RL can now solve the otherwise difficult-to-analyze optimization or multi-agent control problem. The straightforward approach is to use the limiting equations (2), (3) and (4) to define an environment, and apply single-agent policy gradients to the problem, see Alg. 1.

As discussed in Section 2.4, we obtain a valid VRP solution by sampling MARL actions for each vehicle, and performing a postprocessing step where objects not delivered are appended to the route of some random vehicles. Preprocessing is performed via k-means with negligible run time of less than a second per VRP instance, i.e., we assume the data is already clustered into K clusters. Moreover, the depot is appended as an

Algorithm 1 MFVRP-G

```
1: Input: $\mu_0$, $\nu_0$, distance matrix.
2: for iterations $n = 1, 2, \dots$ do
3:     for time steps $t = 0, \dots, B_{\text{len}} - 1$ do
4:         Sample MFC action $T_t \sim \hat{\pi}^\theta(T_t \mid \mu_t, \nu_t)$.
5:         Compute reward $r(\mu_t)$ and next MFs $\mu_{t+1}$, $\nu_{t+1}$ as in Eqs. (2), (3), (4), and termination / reset flag $d_{t+1} \in \{0, 1\}$.
6:     end for
7:     Update $\hat{\pi}^\theta$ on minibatch sampled from data $\{((\mu_t, \nu_t), T_t, r_t, d_{t+1}, (\mu_{t+1}, \nu_{t+1}))\}_{t \geq 0}$.
8: end for
```

additional cluster centroid at runtime. Whenever a vehicle transports an object from cluster i to cluster j, we randomly iterate through vehicles and assign the nearest not yet transported object from i to j, see Fig. 2.

For learning a general solution under fixed K, we dynamically generate every possible scenario by sampling ν_0 and cluster centroids randomly according to flat Dirichlet and uniform distributions respectively. This can be understood as "pretraining" on all possible limiting MFC scenarios for given K. Alternatively, consider learning on a large fixed data set, e.g. if real data is available and clustered.

Datasets Fine-Tuning Beyond learning to solve the general case, an improved approach is to specifically consider and train on a fixed VRP instance. We can still take the pretrained general solution as a warm start and fine-tune it on this particular instance to achieve improved results. Here, there are again two possibilities: We can discretize the fixed VRP instance and perform further training on the limiting MFVRP MDP (MFVRP-F). An alternative approach (MFVRP-FD) is to directly train on the VRP instance by the procedure of obtaining a VRP solution, as discussed earlier. This would possibly avoid errors stemming from the MFC approximation in smaller finite systems. However, we find that the latter gives similar results and takes too much time. We thus mainly focus on the former MFVRP-F method. Overall, the discussed approaches result in Algorithm 2. The advantage is possibly improved results, at the cost of time-expensive training, which is ameliorated by parallelized and fast training code.

Algorithm 2 MFVRP-F/FD

```
1: Input: $\mu_0^N$, $\nu_0^N$, distance matrix, pretrained MFVRP-G model.
2: for iterations $n = 1, 2, \dots$ do
3:     for time steps $t = 0, \dots, B_{\text{len}} - 1$ do
4:         Sample MFC action $T_t \sim \hat{\pi}^\theta(T_t \mid \mu_t, \nu_t)$.
5:         (F): Compute reward $r(\mu_t)$ and next MFs $\mu_{t+1}$, $\nu_{t+1}$ and flag $d_{t+1} \in \{0, 1\}$ as in Alg. 2 for specific VRP instance.
6:         for vehicle $i = 1, \dots, N$ do
7:             (FD): Sample per-vehicle action $u_t^i \sim T_t(u_t^i \mid x_t^i)$.
8:         end for
9:         (FD): Perform actions, observe reward $r_t$, next MF $\mu_{t+1}$, termination flag $d_{t+1} \in \{0, 1\}$
10:    end for
11:    Update $\hat{\pi}^\theta$ on minibatch sampled from data $\{((\mu_t, \nu_t), T_t, r_t, d_{t+1}, (\mu_{t+1}, \nu_{t+1}))\}_{t \geq 0}$.
12: end for
```

In experiments, we consider datasets "finite", "uniform" and "cities". The former consists of a finite state space $|\mathcal{X}| = K$, with distances from randomly sampled locations on $[-1, 1]^2$. The latter takes the latitude and longitude of the 80 largest German cities and randomly samples K cities, around which source and destination locations are normal distributed. For both of the above, we randomly sample a joint distribution over source and destinations in $[-1, 1]^4$ from a flat Dirichlet. Each object is then sampled from this joint distribution, and in the case of "cities" they are additionally perturbed around the location of the city. Lastly, for "uniform" we sample source and destination locations uniformly at random from $[-1, 1]^2$, which gives us a "worst-case" for clustering and MFVRP.

4 Numerical Experiments

In the following, we perform an experimental evaluation of our algorithms on large problem instances. The implementation can be found in the supplementary materials.

4.1 Implementation Details

For the implementation, we use JAX with PPO (Schulman et al., 2017) based on PureJaxRL (Lu et al., 2022). We compare against the OR-Tools routing problem solver (Perron & Furnon, 2023), the recent state-of-the-art PyVRP solver Wouda et al. (2024) and a greedy heuristic where vehicles always fly to the nearest next mission. In the latter, we iterate through each agent until their maximum tour length is exhausted. For OR-Tools and PyVRP, we solve the problem as a CVRP using scaled distances as described in Eq. (1) by multiplying by 10^6 and rounding to the nearest integer.

For our training, we used a GPU with 40 GB VRAM and over 19 TFLOPS of FP32 compute performance for a single hour for each fixed K in MFVRP-G (pretraining), and around two to three minutes per MFVRP-F or MFVRP-FD run for a particular problem instance. We learned policies with two hidden layers of 256 nodes and tanh activations. The discount factor is set to $\gamma = 0.99$, and with GAE $\lambda = 0.95$. The minibatch sizes are 10000 for MFVRP-F, as well as annealed from 10000 to 250000 for MFVRP-G, with 8 PPO steps per training batch. The clip parameter is set to 0.2. The learning rate was linearly annealed from 0.00003 down to zero. As usual in RL, our implementation uses discounted objectives and stationary time-independent policies (Guo et al., 2022), and we also transform the current time t into a one-hot observation. Unless stated otherwise, we use the parameters $K = 5$, $R = 5$, $C = 15$, $c_{\mathrm{d}} = 50$, $c_{\mathrm{miss}} = 30$, $r_{\mathrm{bonus}} = 5$ and datasets for clustered and uniform scenarios with all locations in $\mathcal{X} = [-1, 1]^2$, generated as described in the Appendix. We perform experiments on the aforementioned 3 types of datasets using the parameter K with 50 instances each, to be able to assess the performance and generalization of our algorithms in different scenarios. For uniform datasets, in the following K is merely a hyperparameter of the algorithm.

4.2 Experimental Results

In Fig. 3, we show the training curves for the general MFVRP-G trained on $K = 3$ and $K = 5$. The curve is smooth as we chose a sufficiently high batch size (10000 to 250000), making gradient estimates less noisy. We observe that the methodology converges in terms of episodic returns, and in turn that the maximization of episodic returns indeed minimizes the travelled distances of vehicles while completing all episodes. The results reach a close to 100% completion rate of deliveries while optimizing the distances travelled. This verifies our reward function design.

We can also see that training takes longer as K increases, and uses a great number of time steps, as common for RL methods. In particular, the batch sizes were chosen large to obtain stable training behavior, since the MFVRP MDP has the unique RL challenge of high-dimensional continuous action spaces. Nevertheless, we

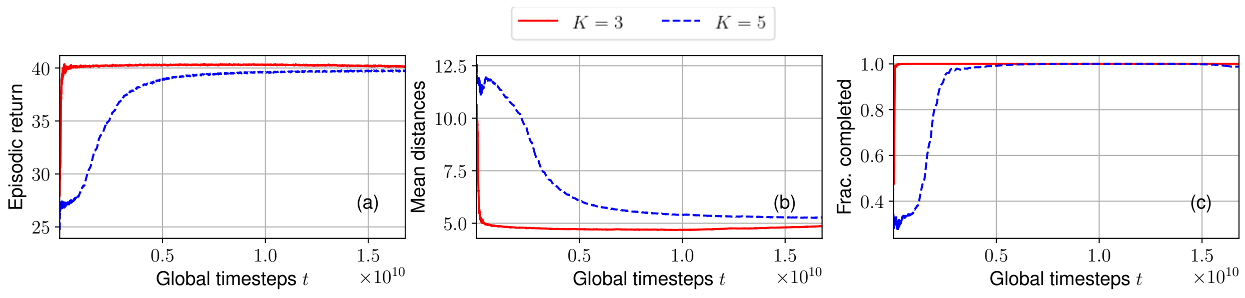

Figure 3: Training curves for MFVRP-G. Comparison of returns, travelled distances, and percentage of completed episodes for MFVRP-G.

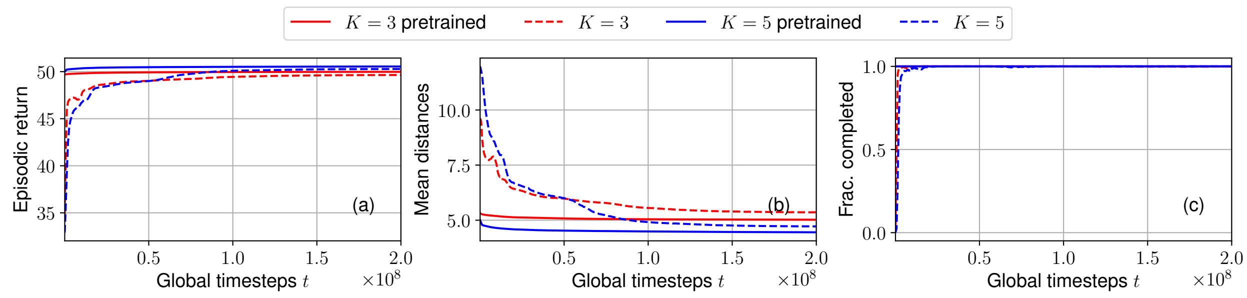

Figure 4: Training curves for MFVRP-F (single trial). Comparison of returns, travelled distances, episode lengths and percentage of completed episodes for MFVRP-F on a particular VRP instance, for $K = 3, M = 500$ and $K = 5, M = 1000$.

observe that RL begins to reduce the return and fraction of completed episodes when training for too long, and we apply early stopping.

Moreover, Fig. 4 contains the results for MFVRP-F with and without warm-starting from the pretrained solution of MFVRP-G. For the training steps seen in the figure, around 3 minutes of GPU training time were used, see Tables 1 and 2. The pretrained solution provides significant increases in performance for the fine-tuned MFVRP-F solution, in contrast to cold-starting training on a given VRP instance.

As a result of the fast training time using a JAX-based learning implementation, our approach is applicable even in time-critical scenarios despite the very large scale of vehicle routing problems, and therefore may prove advantageous over classical methods such as OR-Tools.

Quantitative comparison of speed. In Table 1 and Table 2, we quantitatively compare the results of our approach against OR-Tools and PyVRP as strong practical baselines.

We observe that both the general MFVRP-G and our fine-tuned MFVRP-F (including training time) are faster than OR-Tools in large scenarios, which was configured with maximum computation times of 1 to over 120 minutes for small and larger problem configurations respectively. Training was performed on a GPU, but

Table 1: Comparison and evaluation of VRP methods on various datasets, each consisting of 50 problems for each configuration (M, K). In the finite dataset, $K = |\mathcal{X}|$, whereas in uniform it is a hyperparameter of our methods, and in cities it is also the number of selected city locations. The confidence intervals for the objectives are in the range of < 1.

	Method	$M = 100,\ K = 3$			$M = 1000,\ K = 3$			$M = 5000,\ K = 3$			$M = 50000,\ K = 3$		
		Obj.	Gap	Time	Obj.	Gap	Time	Obj.	Gap	Time	Obj.	Gap	Time
Finite	Greedy	154.35	55.64 %	0.06 s	1548.87	61.20 %	1.02 s	7715.41	58.73 %	5.52 s	73700.7	52.16 %	276.79 s
	OR-Tools	100.76	1.60 %	49.98 s	1056.95	9.99 %	300.00 s	5311.167	9.27 %	8001.01 s	–	–	–
	PyVRP	99.17	*	180.03 s	960.86	*	180.53 s	4860.81	*	192.88 s	–	–	–
	MFVRP-G	107.78	8.68 %	5.09 s	1056.68	9.97 %	12.24 s	5250.19	7.99 %	32.69 s	49692.9	2.59 %	502.47 s
	MFVRP-F	105.72	6.61 %	173.2 s	1030.66	7.26 %	176.30 s	5076.18	4.43 %	199.82 s	48438.2	*	788.39 s
Cities	Greedy	79.43	55.99 %	0.06 s	795.35	49.26 %	0.78 s	3958.64	49.49 %	4.62 s	44404.7	49.83 %	278.56 s
	OR-Tools	50.92	*	60.00 s	532.91	*	300.00 s	2647.75	*	8000.01 s	–	–	–
	PyVRP	58.63	15.14 %	180.02 s	576.70	8.22 %	181.02 s	2879.56	8.75 %	201.64 s	–	–	–
	MFVRP-G	58.09	14.08 %	6.6 s	551.25	3.44 %	14.39 s	2721.67	2.79 %	34.45 s	30743.2	3.73 %	523.10 s
	MFVRP-F	56.90	11.74 %	173.03 s	544.26	2.13 %	180.66 s	2687.12	1.49 %	223.08 s	29635.9	*	775.41 s
Uniform	Greedy	217.76	73.09 %	0.06 s	2206.73	87.12 %	1.02 s	11032.74	89.65 %	5.62 s	110313.7	72.30 %	275.73 s
	OR-Tools	131.35	4.40 %	60.00 s	1230.68	4.35 %	300.00 s	5919.15	1.74 %	8000.01 s	–	–	–
	PyVRP	125.81	*	180.03 s	1179.32	*	180.63 s	5817.59	*	194.11 s	–	–	–
	MFVRP-G	164.99	31.14 %	4.92 s	1406.12	19.23 %	9.73 s	6648.92	14.29 %	31.77 s	64394.3	0.58 %	494.12 s
	MFVRP-F	164.71	30.92 %	159.38 s	1406.39	19.26 %	172.63 s	6633.06	14.02 %	193.26 s	64024.5	*	727.07 s

Table 2: Comparison and evaluation of VRP methods for $K = 5$ as in Table 1.

	Method	$M = 100,\ K = 5$			$M = 1000,\ K = 5$			$M = 5000,\ K = 5$			$M = 50000,\ K = 5$		
		Obj.	Gap	Time	Obj.	Gap	Time	Obj.	Gap	Time	Obj.	Gap	Time
Finite	Greedy	182.28	78.62 %	0.06 s	1849.05	84.11 %	1.02 s	9208.56	83.27 %	5.64 s	86675.4	68.63 %	258.47 s
	OR-Tools	113.00	10.73 %	57.12 s	1174.35	16.93 %	300.00 s	5978.828	18.99 %	8001.01 s	–	–	–
	PyVRP	102.05	*	180.03 s	1004.32	*	180.49 s	5024.67	*	191.90 s	–	–	–
	MFVRP-G	130.91	28.28 %	10.43 s	1222.37	21.72 %	14.99 s	6029.62	20.00 %	42.79 s	55017.1	7.03 %	507.06 s
	MFVRP-F	128.68	26.10 %	215.68 s	1172.54	16.75 %	217.8 s	5766.29	14.76 %	241.85 s	51401.0	*	859.22 s
Cities	Greedy	105.87	68.69 %	0.08 s	1064.14	74.58 %	0.76 s	5333.94	75.25 %	4.32 s	51858.1	61.84 %	258.02 s
	OR-Tools	64.70	3.09 %	60.00 s	670.29	9.97 %	300.00 s	3326.19	9.29 %	8000.01 s	–	–	–
	PyVRP	62.76	*	180.02 s	609.53	*	180.66 s	3043.34	*	198.39 s	–	–	–
	MFVRP-G	79.61	26.84 %	13.29 s	729.45	19.67 %	18.76 s	3581.63	17.69 %	53.36 s	34455.7	7.53 %	513.36 s
	MFVRP-F	78.92	25.75 %	202.17 s	731.32	19.98 %	207.97 s	3563.57	17.09 %	228.8 s	32042.1	*	798.19 s
Uniform	Greedy	217.76	73.09 %	0.06 s	2206.73	87.13 %	1.02 s	11032.74	89.65 %	5.58 s	110313.7	72.28 %	262.77 s
	OR-Tools	131.35	4.40 %	60.00 s	1230.67	4.36 %	300.00 s	5919.67	1.74 %	8000.01 s	–	–	–
	PyVRP	125.81	*	180.02 s	1179.24	*	180.73 s	5818.09	*	194.29 s	–	–	–
	MFVRP-G	171.04	35.95 %	9.64 s	1458.49	23.68 %	14.74 s	6836.91	17.51 %	43.35 s	66644.9	4.08 %	488.32 s
	MFVRP-F	171.91	36.64 %	190.95 s	1453.92	23.30 %	174.96 s	6759.07	16.17 %	192.39 s	64029.3	*	806.69 s

evaluation was performed without. For MFVRP-G the time is the feedforward evaluation time on a standard CPU, whereas for MFVRP-F the times also include the training time on the GPU. Meanwhile, we have slight variation in training and postprocessing time taken, due to minor performance fluctuations. Moreover, OR-Tools failed to produce a solution for very large scales, $M = 5000$, even when the computation time was set to 30 minutes. Similarly, both PyVRP and OR-Tools failed to produce any solution for $M = 50000$ within more than 120 minutes (including pre-processing time).

Overall, we conclude that in terms of speed and scalability, the MF approach can be advantageous for very large scales. Moreover, we would like to point out that most of the increase in time over increasing M for MFVRP solutions is due to inefficient Python postprocessing code where objects (also leftover objects) are assigned to vehicles. This increase in time can be made significantly more efficient, as can the Python loops required for adding all M^2 edges between locations in PyVRP. We point out that MFVRP therefore scales to near-arbitrarily many vehicles if required, with speeds nearly matching simple heuristics.

Quantitative comparison of objectives. As for the quality of objectives given by the total distance travelled by all vehicles, we find that MFVRP-F is usually better than MFVRP-G as a result of fine-tuning on the particular VRP instance of interest. In a separate test, we verify that MFVRP-FD gives results in-between MFVRP-F and MFVRP-G (106.72 for $M = 100, K = 3$).

In comparison to baselines, we find that MFVRP usually outperforms simple heuristics but is behind OR-Tools and PyVRP in smaller-sized problem instances. As the size of problems increases however, the gap to the best method grows increasingly small. We also find that MFVRP can outperform OR-Tools and PyVRP under the right circumstances ($M = 5000$, "finite" and "cities" respectively). Moreover, the method scales automatically to arbitrary numbers of vehicles and objects, whereas OR-Tools and PyVRP take very long to produce results at very large scales. Moreover, we find that for $M = 1000$ and $M = 5000$ in the finite dataset, our approach achieves an increase in performance over OR-Tools. This shows that a mean-field-based methodology to optimization problems can in fact outperform more standard methods in the case of (a) accurate clustering ("finite" dataset), and (b) high numbers of vehicles.

Comparison between datasets. Comparing between different scenarios and parameters, we find an increasingly good approximation of the MFVRP solution as the number of vehicles and objects becomes large, $M \to \infty$. In general, the gap between MFVRP and baseline performance becomes increasingly small or disappears as M increases.

Additionally, we can see that the suboptimality of MFVRP is strongly associated with the clusteredness of the dataset as expected. In the "finite" dataset, we find significantly better comparative performance due to the finite space of locations $|\mathcal{X}| = K$, whereas in "cities" the sources and destinations are scattered around the city centroids. Finally, in "uniform" we essentially have the worst-case for clustering-based approximations, with accordingly worse gaps in performance. To further improve the gap stemming from discretization however, we have to increase K, which in turn increases the hardness of the associated RL problem, since the action space is of dimension K^2. Scaling up RL towards very high-dimensional continuous action spaces is difficult and left as future work.

5 Discussion and Conclusion

In this work, we have proposed a scalable RL methodology based on MFC for solving large-scale CVRP with pickup and delivery. The proposed methods were not only theoretically motivated, but also empirically evaluated and compared against existing baselines as a proof of concept. While the general case with many clusters is yet difficult due to the unique challenge of high-dimensional actions, the solution of particular cases remains useful. In particular, existing standard methods such as OR-Tools may take too long to find a solution for high numbers of vehicles and locations (see Table 2), for which our method provides a fast solution. While solving high-dimensional continuous RL such as the MFVRP remains difficult, we still obtain RL complexities independent of the number of clients (or linear by sampling a VRP solution and routes for each agent from the limiting solution, which is the minimal asymptotic complexity). Overall, we have investigated the applicability of mean-field methods to discrete optimization settings and found that it is at the very least feasible.

At the same time, we have also seen the limitations in applying mean field methods in standard discrete optimization settings (e.g., Section 2.5 and experiments). In future work, one could consider how to apply MFC-based solutions for general non-pickup-delivery VRP and other hard combinatorial optimization problems. One could also scale up to more clusters, e.g. through the usage of curriculum learning and using energy-based actor critic methods (Haarnoja et al., 2018). Further, it remains to be studied whether one can apply appropriate inductive biases and representations of T_t, such as via graph neural networks to generalize to any number of clusters. Finally, one may consider taking inspiration from related classical complexity reduction such as the fast multipole method (Rokhlin, 1985), or avoiding discretization altogether by suitable kernel-based parametrizations.

References

Andrea Angiuli, Jean-Pierre Fouque, and Mathieu Laurière. Unified reinforcement Q-learning for mean field game and control problems. *Math. Control Signals Syst.*, 34(2):217–271, 2022.

Andrea Angiuli, Jean-Pierre Fouque, Ruimeng Hu, and Alan Raydan. Deep reinforcement learning for infinite horizon mean field problems in continuous spaces. *arXiv:2309.10953*, 2023.

Sundararajan Arunapuram, Kamlesh Mathur, and Daniel Solow. Vehicle routing and scheduling with full truckloads. *Transportation Science*, 37(2):170–182, 2003.

Alain Bensoussan, Jens Frehse, Phillip Yam, et al. *Mean field games and mean field type control theory*, volume 101. Springer, 2013.

DD Bochtis and Claus G Sørensen. The vehicle routing problem in field logistics: Part I. *Biosystems engineering*, 104(4):447–457, 2009.

DD Bochtis and Claus G Sørensen. The vehicle routing problem in field logistics: Part II. *Biosystems engineering*, 105(2):180–188, 2010.

Aigerim Bogyrbayeva, Meraryslan Meraliyev, Taukekhan Mustakhov, and Bissenbay Dauletbayev. Machine learning to solve vehicle routing problems: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 2024.

Andreas Bortfeldt and Junmin Yi. The split delivery vehicle routing problem with three-dimensional loading constraints. *European Journal of Operational Research*, 282(2):545–558, 2020.

René Carmona, François Delarue, et al. *Probabilistic theory of mean field games with applications I-II*. Springer, 2018.

René Carmona, Mathieu Laurière, and Zongjun Tan. Linear-quadratic mean-field reinforcement learning: convergence of policy gradient methods. *arXiv:1910.04295*, 2019.

René Carmona, Mathieu Laurière, and Zongjun Tan. Model-free mean-field reinforcement learning: mean-field MDP and mean-field Q-learning. *Ann. Appl. Probab.*, 33(6B):5334–5381, 2023.

Jean-François Cordeau and Gilbert Laporte. The dial-a-ride problem: models and algorithms. *Annals of Operations Research*, 153:29–46, 2007.

Kai Cui, Anam Tahir, Mark Sinzger, and Heinz Koeppl. Discrete-time mean field control with environment states. In *Proc. CDC*, pp. 5239–5246, 2021.

Kai Cui, Sascha Hauck, Christian Fabian, and Heinz Koeppl. Learning decentralized partially observable mean field control for artificial collective behavior. In *Proc. ICLR*, 2024.

Massimo Fornasier and Francesco Solombrino. Mean-field optimal control. *ESAIM: Control, Optimisation and Calculus of Variations*, 20(4):1123–1152, 2014.

Zhang-Hua Fu, Kai-Bin Qiu, and Hongyuan Zha. Generalize a small pre-trained model to arbitrarily large TSP instances. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 7474–7482, 2021.

Xin Guo, Anran Hu, and Junzi Zhang. Theoretical guarantees of fictitious discount algorithms for episodic reinforcement learning and global convergence of policy gradient methods. In *Proc. AAAI*, volume 36, pp. 6774–6782, 2022.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proc. ICML*, pp. 1861–1870, 2018.

Sin C Ho, Wai Yuen Szeto, Yong-Hong Kuo, Janny MY Leung, Matthew Petering, and Terence WH Tou. A survey of dial-a-ride problems: Literature review and recent developments. *Transportation Research Part B: Methodological*, 111:395–421, 2018.

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

Yuan Jiang, Zhiguang Cao, Yaoxin Wu, Wen Song, and Jie Zhang. Ensemble-based deep reinforcement learning for vehicle routing problems under distribution shift. In *Proc. NeurIPS*, 2023.

Leonard Kaufman and Peter J. Rousseeuw. *Partitioning Around Medoids (Program PAM)*, pp. 68–125. John Wiley & Sons, Inc., 2008. ISBN 9780470316801.

Daisuke Kikuta, Hiroki Ikeuchi, Kengo Tajiri, Yuta Toyama, Masaki Nakamura, and Yuusuke Nakano. Electric vehicle routing for emergency power supply with deep reinforcement learning. In *Proc. AAMAS*, pp. 2336–2338, 2024.

Grigorios D Konstantakopoulos, Sotiris P Gayialis, and Evripidis P Kechagias. Vehicle routing problem and related algorithms for logistics distribution: A literature review and classification. *Operational Research*, 22 (3):2033–2062, 2022.

Jean-Michel Lasry and Pierre-Louis Lions. Mean field games. *Japanese J. Math*, 2:229–260, 2007.

Kun Lei, Peng Guo, Yi Wang, Xiao Wu, and Wenchao Zhao. Solve routing problems with a residual edge-graph attention neural network. *Neurocomputing*, 508:79–98, 2022.

Bingjie Li, Guohua Wu, Yongming He, Mingfeng Fan, and Witold Pedrycz. An overview and experimental study of learning-based optimization algorithms for the vehicle routing problem. *IEEE/CAA Journal of Automatica Sinica*, 9(7):1115–1138, 2022.

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

Sirui Li, Zhongxia Yan, and Cathy Wu. Learning to delegate for large-scale vehicle routing. In *Advances in Neural Information Processing Systems*, volume 34, pp. 26198–26211, 2021b.

Yeqian Lin, Wenquan Li, Feng Qiu, and He Xu. Research on optimization of vehicle routing problem for ride-sharing taxi. *Procedia-Social and Behavioral Sciences*, 43:494–502, 2012.

Qidong Liu, Chaoyue Liu, Shaoyao Niu, Cheng Long, Jie Zhang, and Mingliang Xu. 2D-Ptr: 2D array pointer network for solving the heterogeneous capacitated vehicle routing problem. In *Proc. AAMAS*, pp. 1238–1246, 2024.

Chris Lu, Jakub Kuba, Alistair Letcher, Luke Metz, Christian Schroeder de Witt, and Jakob Foerster. Discovered policy optimisation. In *Proc. NeurIPS*, volume 35, pp. 16455–16468, 2022.

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

Yimeng Min, Yiwei Bai, and Carla P Gomes. Unsupervised learning for solving the travelling salesman problem. In *Proc. NeurIPS*, volume 36, 2024.

Santhanakrishnan Narayanan, Emmanouil Chaniotakis, and Constantinos Antoniou. Shared autonomous vehicle services: A comprehensive review. *Transportation Research Part C: Emerging Technologies*, 111: 255–293, 2020.

Laurent Perron and Vincent Furnon. OR-Tools, 2023. URL `https://developers.google.com/optimization/`.

Ruizhong Qiu, Zhiqing Sun, and Yiming Yang. Dimes: A differentiable meta solver for combinatorial optimization problems. In *Advances in Neural Information Processing Systems*, volume 35, pp. 25531–25546, 2022.

Vladimir Rokhlin. Rapid solution of integral equations of classical potential theory. *Journal of computational physics*, 60(2):187–207, 1985.

Naci Saldi, Tamer Basar, and Maxim Raginsky. Markov–Nash equilibria in mean-field games with discounted cost. *SIAM J. Control Optim.*, 56(6):4256–4287, 2018.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv:1707.06347*, 2017.

David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *Proc. ICML*, pp. 387–395, 2014.

Zhiqing Sun and Yiming Yang. Difusco: Graph-based diffusion solvers for combinatorial optimization. In *Advances in Neural Information Processing Systems*, volume 36, pp. 3706–3731, 2023.

Aad W Van der Vaart. *Asymptotic Statistics*, volume 3. Cambridge University Press, 2000.

Yong Wang, Qin Li, Xiangyang Guan, Jianxin Fan, Maozeng Xu, and Haizhong Wang. Collaborative multi-depot pickup and delivery vehicle routing problem with split loads and time windows. *Knowledge-Based Systems*, 231:107412, 2021.

Niels A. Wouda, Leon Lan, and Wouter Kool. PyVRP: a high-performance VRP solver package. *INFORMS Journal on Computing*, 36(4):943–955, 2024.

Yaoxin Wu, Mingfeng Fan, Zhiguang Cao, Ruobin Gao, Yaqing Hou, and Guillaume Sartoretti. Collaborative deep reinforcement learning for solving multi-objective vehicle routing problems. In *Proc. AAMAS*, pp. 1956–1965, 2024.

Yifan Xia, Xianliang Yang, Zichuan Liu, Zhihao Liu, Lei Song, and Jiang Bian. Position: Rethinking post-hoc search-based neural approaches for solving large-scale traveling salesman problems. In *Proc. ICML*, pp. 54178–54190, 2024.

Yubin Xiao, Di Wang, Boyang Li, Mingzhao Wang, Xuan Wu, Changliang Zhou, and You Zhou. Distilling autoregressive models to obtain high-performance non-autoregressive solvers for vehicle routing problems with faster inference speed. In *Proc. AAAI*, volume 38, pp. 20274–20283, 2024.

Haifei Zhang, Hongwei Ge, Jinlong Yang, and Yubing Tong. Review of vehicle routing problems: Models, classification and solving algorithms. *Archives of Computational Methods in Engineering*, pp. 1–27, 2022.

Kenan Zhang and John Lygeros. Routing and charging game in ride-hailing service with electric vehicles. In *Proc. CDC*, pp. 3116–3121, 2023.

Zhi Zheng, Shunyu Yao, Zhenkun Wang, Tong Xialiang, Mingxuan Yuan, and Ke Tang. DPN: Decoupling partition and navigation for neural solvers of min-max vehicle routing problems. In *Proc. ICML*, pp. 61559–61592, 2024.

Jianan Zhou, Zhiguang Cao, Yaoxin Wu, Wen Song, Yining Ma, Jie Zhang, and Xu Chi. MVMoE: Multi-task vehicle routing solver with mixture-of-experts. In *Proc. ICML*, pp. 61804–61824, 2024.

Zefang Zong, Meng Zheng, Yong Li, and Depeng Jin. MAPDP: Cooperative multi-agent reinforcement learning to solve pickup and delivery problems. In *Proc. AAAI*, volume 36, pp. 9980–9988, 2022.

A Dataset Details

In this study, we generate 3 different types of datasets, called uniform, finite and cities. For each type of dataset we vary the amount of objects to be delivered $M = \{100, 1000, 5000, 50000\}$, and for the finite and cities distribution we also vary the amount of clusters, $K \in \{3, 5\}$.

The positions of the K clusters in finite are sampled uniformly from $[-1, 1]^2$. Meanwhile, in cities we sample the positions from a list of rescaled longitudes and latitudes of the 80 most populated German cities. The rescaling is performed by subtracting the mean longitude and latitude of $(51, 10)$ and dividing by 5. The source and destination cluster assignments are sampled from a flat Dirichlet distribution. The exact position of the M targets and sources is then sampled from a Gaussian distribution around the positions of one of the corresponding K clusters, with a standard deviation of 0 and 0.02 for finite and cities respectively, with samples clipped between -1 and 1. For the latter, this standard deviation of 0.02 corresponds to a city radius (95% of samples inside) of 16 km.

For the uniform dataset we sample M sources and destinations uniformly from $[-1, 1]^2$. We generate 50 instances for each combination of M and k, resulting in 200 VRP instances for uniform, and 400 VRP instances for cities and finite.

B Proofs

B.1 Mean Field Convergence

Proof of Theorem 2.1. First, note that with deterministic $\hat{\pi}$ comes constant (non-random) T_t output over all μ_t, ν_t. Indeed, even constant MFC policies are sufficient due to the deterministic evolution of μ_t, ν_t given the initial state, and are equivalent to any trained RL policy $\hat{\pi}$ for the limiting MFC problem with open-loop control sequence $T_0, T_1, T_2, \ldots$. We therefore use an equivalent optimal, but constant MFC policy $\hat{\pi}'_t(T|\cdot,\cdot) = \delta_{T_t}$. Therefore, T_t is no longer random and entirely deterministic.

The statement is shown inductively by convergence in expectation, $\mathbb{E}\left[\|\mu_t^N - \mu_t\|_\infty\right] \to 0$ and $\mathbb{E}\left[\|\nu_t^N - \nu_t\|_\infty\right] \to 0$. Convergence in probability then follows. At time $t = 0$, the statement is immediate by weak law of large numbers (LLN). Assuming the statement at time t, then at time $t+1$, as a short sketch, we obtain

$$\begin{aligned}\mathbb{E}\left[\|\mu_{t+1}^N - \mu_{t+1}\|_\infty\right] \leq \sum_{x\in\mathcal{X}} \mathbb{E}\left[\left|\frac{1}{N}\sum_{\substack{s\in\mathcal{X}\\ i\leq N}} \delta_{x_t^i}(s) u_{t,s}^i - \frac{1}{N}\sum_{\substack{s\in\mathcal{X}\\ i\leq N}} \delta_{x_t^i}(s) T_t(s,x)\right|\right] \\ + \mathbb{E}\left[\left|\frac{1}{N}\sum_{\substack{s\in\mathcal{X}\\ i\leq N}} \delta_{x_t^i}(s) T_t(s,x) - \sum_{s\in\mathcal{X}} \mu_t(s) T_t(s,x)\right|\right],\end{aligned}$$

and can apply induction assumption for the latter term, and a weak LLN argument for the former. The same holds for ν_t^N, ν_t.

More precisely, for the analysis, we lift the actions to a new action space $\mathcal{X}^{\mathcal{X}}$ (Carmona et al., 2018) and note the equivalence *in distribution* to the original MARL system by using $u_{t,x}^i$ whenever vehicle i is in state x. Then, we may apply the discussed constant MFC policies by sampling $u_{t,x}^i \sim T_t(x,\cdot)$ for all $x \in \mathcal{X}$ and using the corresponding component, instead of $u_t^i \sim T_t(x_t^i, \cdot)$. The advantage in analysis is that $u_{t,x}^i$ is i.i.d. for all i, t, x. Then, $\nu_{t+1}^N(s,d)$ is given by subtracting from $\nu_t^N(s,d)$ the random variables $\frac{1}{N}\mathbf{1}_{x_t^i=s, u_{t,s}^i=d}$ for each of $i \in \mathcal{N}$ vehicles, up to at most $NR\nu_{t+1}^N(s,d)$ many.

The overall result is then obtained in two steps, which are shown in App. B.2 and B.3. The proof proceeds in the above system where the actions are i.i.d., and the results follow for the original system by equivalence *in distribution* of both systems for the two following lemmas.

Lemma B.1 (Propagation of Chaos I). *Consider finite $\mathcal{X}$, $t_{\max} < \infty$ and deterministic $\hat{\pi}$. For each $t \leq t_{\max}$, as $N \to \infty$,*

$$\mathbb{E}\left[\|\mu_t^N - \mu_t\|_\infty\right] = \mathcal{O}(1/\sqrt{N}) \to 0\,.$$

Lemma B.2 (Propagation of Chaos II). *Consider finite $\mathcal{X}$, $t_{\max} < \infty$ and deterministic $\hat{\pi}$. For each $t \leq t_{\max}$, as $N \to \infty$,*

$$\mathbb{E}\left[\|\nu_t^N - \nu_t\|_\infty\right] = \mathcal{O}(1/\sqrt{N}) \to 0$$

From the above lemmas, the proof of Theorem 2.1 follows: The desired statement was shown inductively by convergence in expectation, $\mathbb{E}\left[\|\mu_t^N - \mu_t\|_\infty\right] \to 0$ and $\mathbb{E}\left[\|\nu_t^N - \nu_t\|_\infty\right] \to 0$ by Lemmas B.1 and B.2. Convergence in probability then follows by Van der Vaart (2000, Theorem 2.7) and deterministic μ_t, ν_t. ☐

B.2 Propagation of Chaos I

Proof of Lemma B.1. For the analysis, we lift the actions to a new action space $\mathcal{X}^{\mathcal{X}}$ (Carmona et al., 2018) and note the equivalence *in distribution* to the original MARL system, as discussed in B.1. We prove the statement via induction over t. For $t = 0$, the statement follows from a law of large numbers argument (see induction step below, first term). Thus, it remains to show the induction step. Assume that the equation holds at time t. Then, at time $t + 1$ we have

$$\begin{aligned}
&\mathbb{E}\left[\|\mu_{t+1}^N - \mu_{t+1}\|_\infty\right] = \mathbb{E}\left[\max_{x\in\mathcal{X}} |\mu_{t+1}^N(x) - \mu_{t+1}(x)|\right] \\
&\leq \mathbb{E}\left[\sum_{x\in\mathcal{X}} |\mu_{t+1}^N(x) - \mu_{t+1}(x)|\right] = \sum_{x\in\mathcal{X}} \mathbb{E}\left[|\mu_{t+1}^N(x) - \mu_{t+1}(x)|\right] \\
&= \sum_{x\in\mathcal{X}} \mathbb{E}\left[\left|\left(\frac{1}{N}\sum_{s\in\mathcal{X}}\sum_{i\leq N} \delta_{x_t^i}(s) u_{t,s}^i(x)\right) - \sum_{s\in\mathcal{X}} \mu_t(s) T_t(s,x)\right|\right] \\
&= \sum_{x\in\mathcal{X}} \mathbb{E}\left[\left|\left(\frac{1}{N}\sum_{s\in\mathcal{X}}\sum_{i\leq N} \delta_{x_t^i}(s) u_{t,s}^i(x)\right) - \left(\frac{1}{N}\sum_{s\in\mathcal{X}}\sum_{i\leq N} \delta_{x_t^i}(s) T_t(s,x)\right)\right.\right. \\
&\qquad\qquad \left.\left. + \left(\frac{1}{N}\sum_{s\in\mathcal{X}}\sum_{i\leq N} \delta_{x_t^i}(s) T_t(s,x)\right) - \sum_{s\in\mathcal{X}} \mu_t(s) T_t(s,x)\right|\right] \\
&\leq \sum_{x\in\mathcal{X}} \mathbb{E}\left[\left|\left(\frac{1}{N}\sum_{s\in\mathcal{X}}\sum_{i\leq N} \delta_{x_t^i}(s) u_{t,s}^i(x)\right) - \frac{1}{N}\sum_{s\in\mathcal{X}}\sum_{i\leq N} \delta_{x_t^i}(s) T_t(s,x)\right|\right] \\
&\qquad\qquad + \mathbb{E}\left[\left|\left(\frac{1}{N}\sum_{s\in\mathcal{X}}\sum_{i\leq N} \delta_{x_t^i}(s) T_t(s,x)\right) - \sum_{s\in\mathcal{X}} \mu_t(s) T_t(s,x)\right|\right] \\
&= \sum_{x\in\mathcal{X}} \mathbb{E}\left[\left|\left(\frac{1}{N}\sum_{s\in\mathcal{X}}\sum_{i\leq N} \delta_{x_t^i}(s) u_{t,s}^i(x)\right) - \frac{1}{N}\sum_{i\leq N} \mathbb{E}\left[u_t^i(x) \middle| x_t^1, \ldots, x_t^N\right]\right|\right] \\
&\qquad\qquad + \mathbb{E}\left[\left|\sum_{s\in\mathcal{X}} T_t(s,x) \left(\mu_t(s) - \frac{1}{N}\sum_{i\leq N} \delta_{x_t^i}(s)\right)\right|\right] \\
&\leq \sum_{x\in\mathcal{X}} \mathbb{E}\left[\frac{1}{N^2}\left(\sum_{i\leq N} u_t^i(x) - \mathbb{E}\left[u_t^i(x) \middle| x_t^1, \ldots, x_t^N\right]\right)^2\right]^{1/2} + \sum_{s\in\mathcal{X}} T_t(s,x) \underbrace{\mathbb{E}\left[\left|\mu_t(s) - \frac{1}{N}\sum_{i\leq N} \delta_{x_t^i}(s)\right|\right]}_{\leq \mathcal{O}(1/\sqrt{N}) \text{ by IA}}
\end{aligned}$$

$$= \sum_{x\in\mathcal{X}} \mathbb{E}\left[\frac{1}{N^2}\sum_{i\leq N}\left(u_t^i(x) - \mathbb{E}\left[u_t^i(x)\middle|x_t^1,\dots,x_t^N\right]\right)^2\right]^{1/2} + \mathcal{O}(1/\sqrt{N})$$
$$= \frac{2|\mathcal{X}|}{\sqrt{N}} + \mathcal{O}(1/\sqrt{N}),$$

which concludes the proof. □

B.3 Propagation of Chaos II

Proof of Lemma B.2. For the analysis, we lift the actions to a new action space $\mathcal{X}^{\mathcal{X}}$ (Carmona et al., 2018) and note the equivalence *in distribution* to the original MARL system, as discussed in B.1. As before, we prove the Lemma via induction over t and note that the induction start $t = 0$ holds by a law of large numbers argument (alternatively, see induction step below). Now, the induction step is provided by the following argument, where we assume that the statement holds for t:

$$\mathbb{E}\left[\|\nu_{t+1}^N - \nu_{t+1}\|_\infty\right] = \mathbb{E}\left[\max_{s,d\in\mathcal{X}}|\nu_{t+1}^N(s,d) - \nu_{t+1}(s,d)|\right]$$
$$\leq \mathbb{E}\left[\sum_{s,d\in\mathcal{X}}|\nu_{t+1}^N(s,d) - \nu_{t+1}(s,d)|\right] = \sum_{s,d\in\mathcal{X}}\mathbb{E}\left[|\nu_{t+1}^N(s,d) - \nu_{t+1}(s,d)|\right]$$
$$= \sum_{s,d\in\mathcal{X}}\mathbb{E}\left[\left|\max\left(0, \underbrace{\nu_t^N(s,d) - \frac{1}{RN}\sum_{i\leq N}\delta_{x_t^i}(s)u_{t,s}^i(d)}_{=:\mathrm{I}_N}\right) - \max(0, \underbrace{\nu_t(s,d) - \mu_t(s)T_t(s,d)/R}_{=:\mathrm{II}})\right|\right] \tag{5}$$

For the next argument, fix some arbitrary tuple $(s,d)\in\mathcal{X}^2$. For the moment, focus on

$$\mathbb{E}\left[\left|\nu_t^N(s,d) - \frac{1}{RN}\sum_{i\leq N}\delta_{x_t^i}(s)u_{t,s}^i(d) - (\nu_t(s,d) - \mu_t(s)T_t(s,d)/R)\right|\right]$$
$$\leq \mathbb{E}\left[\left|\nu_t^N(s,d) - \nu_t(s,d)\right|\right] + \frac{1}{R}\mathbb{E}\left[\left|\mu_t(s)T_t(s,d) - \frac{1}{N}\sum_{i\leq N}\delta_{x_t^i}(s)u_{t,s}^i(d)\right|\right] = \mathcal{O}(1/\sqrt{N}),$$

where the first summand converges to zero by the induction assumption and the second summand converges to zero by an argument as in the proof of Theorem B.1. The above result implies convergence of equation (5) to zero because for each arbitrary but fixed tuple $(s,d)\in\mathcal{X}^2$ and any $N\in\mathbb{N}$ we have

$$|\max(0,\mathrm{I}_N) - \max(0,\mathrm{II})| \leq |\mathrm{I}_N - \mathrm{II}|\,.$$

To see that the inequality is true, make a case distinction. If both I_N and II are non-negative, the statement trivially holds. If both I_N and II are negative, the left side of the inequality is 0 and thereby the inequality is obviously fulfilled since there is an absolute value on the right side. Finally, in the third case, assume that I_N is non-negative and II is negative (or vice versa). In the third case, the left side equals I_N while the right side equals $\mathrm{I}_N - \mathrm{II} > \mathrm{I}_N$ since II is negative. This completes the proof of Theorem B.2. □

B.4 Approximate Optimality with High Probability

Proof of Corollary 2.2. The proof idea is to compare the objectives (sums of expected rewards) of the respective policies and systems, and thereby establish the following sequence of inequalities for objectives, holding at least with probability $1-\delta$ as $N\to\infty$:

$$\mathrm{MARL} \overset{(1)}{\geq} \mathrm{MFC}^* - \varepsilon \overset{(2)}{\geq} \mathrm{MFC}'^* - 2\varepsilon \overset{(3)}{\geq} \mathrm{MARL}^* - 2\varepsilon\,.$$

Let $\varepsilon > 0$ arbitrary and consider an optimal MFC solution where the corresponding objective value is denoted by MFC* in the above inequality sequence. By Theorem 2.1, the probability of any positive fraction of objects $\%_{\mathrm{rem}} = \sum_{s,d} \nu^N_{t_{\max}}(s,d) \geq 0$ remaining after termination time $t_{\max}$ of the MFVRP MDP tends to zero, or equivalently

$$\mathbb{P}\left(\%_{\mathrm{rem}} < \frac{\varepsilon}{8\operatorname{diam}(\mathcal{X})}\right) = \mathbb{P}\left(\left|\sum_{s,d} \nu^N_{t_{\max}}(s,d) - \sum_{s,d} \nu_{t_{\max}}(s,d)\right| < \frac{\varepsilon}{8\operatorname{diam}(\mathcal{X})}\right) \to 1$$

as $N \to \infty$, as an optimal MFC solution transports all objects in the limit, $\sum_{s,d} \nu_{t_{\max}}(s,d) = 0$. Moreover, by Theorem 2.1, the MARL trajectories' rewards are close to the MFC rewards up to time $t_{\max}$ with high probability,

$$\mathbb{P}\left(\left|\sum_{t \leq t_{\max}} r^N_t - \sum_{t \leq t_{\max}} r_t\right| < \frac{\varepsilon}{4}\right) \to 1.$$

Let the former event be A, and the latter B. Overall, therefore $\mathbb{P}(A \wedge B) \geq \max(0, \mathbb{P}(A) + \mathbb{P}(B) - 1) \to 1$. Moreover, $A \wedge B$ jointly implies that $\mathbb{P}\left(\left|\sum_t r^N_t - \sum_t r_t\right| < \frac{\varepsilon}{2}\right) \to 1$ by definition of the reward, i.e. for sufficiently large N the probability that a sampled solution's return is approximately given by the optimal MFC solution's return tends to one and thus establishes the first inequality (1).

To prove the third inequality (3), note that the return of any optimal MARL solution can also be exactly obtained in a MFC system with modified initial distributions, by using ν^N_0, μ^N_0 as starting values instead of ν_0, μ_0. Letting $T_t(s,d) = \frac{\sum_i \mathbf{1}_s(x^i_t)\mathbf{1}_d(u^i_t)}{\sum_i \mathbf{1}_s(x^i_t)}$ obtains exactly the optimal MARL objective in the modified MFC system denoted as MFC$'^*$ in the above inequality sequence. Therefore, the optimal MARL objective is bounded by the optimal modified MFC objective which corresponds to inequality (3).

Finally, it remains to establish inequality (2). We point out that the optimal MFC objective is continuous in its starting conditions, since Eqs. (2)–(4) are continuous. This implies that the optimal modified MFC objective tends to the optimal MFC objective as ν^N_0, μ^N_0 tend to ν_0, μ_0, and by Thm. 2.1, the probability that ν^N_0, μ^N_0 are close to ν_0, μ_0 tends to one which yields inequality (2).

Therefore, the probability that the sampled MARL solution is ε-optimal in the MARL system also tends to one. $\square$

B.5 Approximate Optimality under Fine Discretization

Proof of Theorem 3.1. The proof structure consists of establishing the following five inequalities of objectives as in Corollary 2.2,

$$\begin{aligned} \text{MARL} &\geq \delta\text{-MARL} - \varepsilon' \\ &\geq \text{MFC}^* - \varepsilon' - \varepsilon \geq \text{MFC}'^* - \varepsilon' - 2\varepsilon \\ &\geq \delta\text{-MARL}^* - \varepsilon' - 2\varepsilon \geq \text{MARL}^* - 2\varepsilon' - 2\varepsilon\,. \end{aligned}$$

First, we obtain the center inequality sequence

$$\delta\text{-MARL} - \varepsilon' \geq \text{MFC}^* - \varepsilon' - \varepsilon \geq \text{MFC}'^* - \varepsilon' - 2\varepsilon \geq \delta\text{-MARL}^* - \varepsilon' - 2\varepsilon$$

by the same argument as in the proof of Corollary 2.2, where δ-MARL refers to the discretized VRP problem.

Therefore, it remains to establish the first and last inequality of the initial inequality sequence. These two inequalities intuitively build on the observation that the discretized δ-MARL system is close to the initial, continuous MARL system. Formally, we know that each of the M objects 'cause' at most two vehicle moves: one to be picked up and one to be delivered to the destination. In the worst case, all N vehicles have to additionally fly back to their respective depots, which upper bounds the total number of travels by $2M + N$.

By the triangle inequality, we then know that each travel in the discretized system deviates at most $2\max_k \operatorname{diam}(C_k)$ from the respective travel in the continuous system. Consequently, the overall sum of deviations, and therefore the deviation of objective in the systems, is upper bounded by $\varepsilon' = 2 \cdot (2M + N)\max_k \operatorname{diam}(C_k)$. This argument holds uniformly for all solutions and hence establishes

$$\text{MARL} \geq \delta\text{-MARL} - \varepsilon' \;\text{ and }\; \delta\text{-MARL}^* \geq \text{MARL}^* - \varepsilon'$$

and thereby concludes the proof. □