Structured-Learning-Based Fleet Control for Autonomous Mobility-on-Demand Systems

K. Jungel[a,*], A. Parmentier[b], M. Schiffer[a,c] and T. Vidal[d,e]

[a] TUM School of Management, Technical University of Munich, Munich, Germany
kai.jungel@tum.de, schiffer@tum.de
[b] CERMICS, École des Ponts, Marne-la-Vallée, France
axel.parmentier@enpc.fr
[c] Munich Data Science Institute, Technical University of Munich, Munich, Germany
[d] CIRRELT & SCALE-AI Chair in Data-Driven Supply Chains, Department of Mathematics and Industrial Engineering, École Polytechnique de Montréal, Canada
thibaut.vidal@cirrelt.ca
[e] Department of Computer Science, Pontifical Catholic University of Rio de Janeiro, Rio de Janeiro, Brazil
* Corresponding author

Extended abstract submitted for presentation at the 11th Triennial Symposium on Transportation Analysis conference (TRISTAN XI)
June 19-25, 2022, Mauritius Island

April 4, 2022

Keywords: Autonomous Mobility-on-Demand, Structured Learning, online algorithm

1 INTRODUCTION

Autonomous Mobility-on-Demand (AMoD) systems constitute a viable alternative to mitigate today's mobility system's externalities such as the rising traffic volume in urban areas and transportation-related pollution. An AMoD system is a centrally controlled fleet of self-driving vehicles serving on-demand ride requests. The central control of the MoD fleet allows the optimal dispatching of vehicles to ride requests in an offline problem setting. The respective online problem setting causes the central control to dispatch vehicles to ride requests without information about future ride requests, in hindsight leading to an optimality gap compared to the offline solution. To reduce this gap, some online approaches aim to anticipate future ride requests, e.g. via sampling from known request distributions (Alonso-Mora *et al.* , 2017). However, the fleet control of sampling-based approaches is biased towards the predefined request distribution and tends to misleading dispatching and rebalancing decisions in scenarios with inaccurate distributional information.

We address this problem by proposing a structured-learning-based framework that allows for prescriptive online dispatching learned from offline dispatching solutions. We derive this framework in four steps: first, we show how to state the underlying dispatching problem in its offline variant as a k-disjoint shortest paths problem, which allows to solve it in polynomial time. Second, we introduce an extended problem representation for the online dispatching problem, which allows to incorporate future ride requests in the graph representation of each online batch. Third, we present a structured learning model (cf. Parmentier, 2021) which learns to predict the arc weights for this extended dispatching graph such that the solution of the extended dispatching problem incorporates prescriptive rebalancing decisions. This finally allows us to devise an online dispatching algorithm for a centrally controlled MoD fleet that automatically adapts to different scenarios. We benchmark our framework on the New York taxi data set and show that our

approach outperforms existing MoD control approaches on scenarios with different numbers of requests per time interval and different vehicles to request ratios.

2 PROBLEM SETTING

We consider an operator who has complete control over a MoD vehicle fleet $V = \{v_1, ..., v_n\}$ and dispatches vehicles to serve ride requests $R = \{r_1, ..., r_m\}$. Each ride request has a pickup location, dropoff location, start time, arrival time, and a revenue defining the value for the operator to fulfill this ride request. Then, the operator's objective is to maximize its reward by dispatching vehicles to ride requests. A ride request can represent a solitary ride or pooled rides as we assume pooling decisions to be taken at an upper planning level. Accordingly a vehicle can serve only one ride at a time, and a vehicle can fulfill two subsequent ride requests when they are feasibly connectable in time and space.

Within this setting, we distinguish between an offline and an online problem variant. An offline system covers the complete planning horizon and contains full information about all ride requests. In the online problem, ride requests enter the system over time.

3 METHODOLOGY

Our algorithmic framework utilizes a novel *offline imitation learning paradigm*, see Figure 1. The main rationale of this paradigm is to minimize the distance between an online solution and the corresponding offline solution that could be obtained under full information. To reach this goal, we use an algorithm Ω to transform the online instance Γ^{on}, which represents the rolling horizon online batch, into an extended online instance Γ^{extOn}, and use an encoding algorithm φ_Θ to parameterize Γ^{extOn}. The encoding algorithm φ_Θ learns the parameterization of Γ^{extOn} from precalculated offline solutions, which allows to incorporate information about possible future system states and the value of corresponding (rebalancing) decisions. We then solve the extended online problem instance Γ^{extOn} using algorithm $\mathcal{A}$ and denote the solution with x^{extOn}.

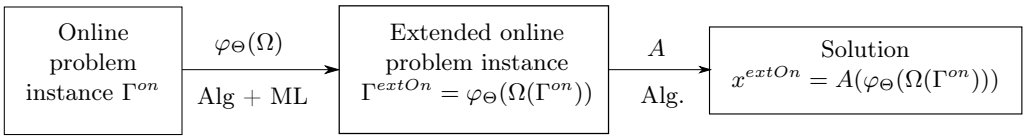
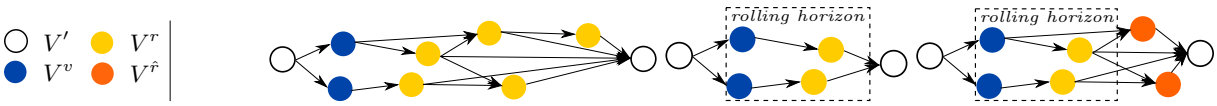


Figure 1 – *The offline imitation learning paradigm*

The crucial part in the offline imitation learning paradigm is to learn parameter vector Θ such that the solution $x^{extOn} = A(\varphi_\Theta(\Omega(\Gamma^{on})))$ imitates an optimal offline solution x^{off}. Successfully applying this paradigm requires (a) an offline control algorithm that finds an optimal solution to the offline problem, (b) an online control algorithm that is compatible with the offline problem, (c) the formulation of the online control problem as a structured-prediction problem, and (d) a tractable structured-learning problem.

Offline control: We formulate our offline dispatching problem as a k-disjoint shortest paths problem (cf. Schiffer *et al.* , 2021). We consider a weighted directed graph $G = (V, A, W)$ with a set of vertices V, a set of arcs A and a set of weights W with a weight $w \in W$ for each arc $a \in A$. The vertex set is $V = \{V' \cup V^v \cup V^r\}$, where V^r denotes all ride requests, V^v denotes the initial positions of vehicles, and V' contains a dummy source and a dummy sink vertex. We then construct this graph such that only ride request representing vertices that can be subsequently served are connected by an arc. Further, we connect each vehicle vertex to all reachable ride requests, the dummy source to all vehicle vertices and all ride request vertices to the dummy sink, see Figure 2a. Finally, we associate the weight of a ride request vertex's incoming arc with the ride request's negative reward. This allows us to obtain a solution to the offline problem



(a) *offline* (b) *online* (c) *extended online*

Figure 2 – *Comparison of offline, online, and extended online dispatching graph*

in polynomial time by solving a k-disjoint shortest path problem on this graph structure; here, each obtained path denotes a feasible vehicle schedule that is part of the optimal dispatching strategy. We refer to this algorithm as $\mathcal{A}$.

Online control: To obtain an online control problem that is compatible with its offline counterpart's algorithm, we use a re-optimization approach (cf. Bertsimas *et al.* , 2019) in a rolling horizon fashion. Here, we construct a dispatching graph that contains only the ride request vertices representing ride requests that enter the system in between two time steps, see Figure 2b. To make this dispatching graph compatible with its offline counterpart, we extend it with artificial vertices $V^{\hat{r}}$, see Figure 2c. To do so, the extension algorithm Ω splits the spatial simulation area into square rebalancing cells of equal size and predicts a future ride demand from a historical ride request distribution for every rebalancing cell. Each estimated future ride demand is an artificial request vertex in the extended online batch graph representation and constitutes a rebalancing location. Then, an offline solution with a vehicle fulfilling a future ride request coincides with a vehicle driving to an artificial request vertex in the extended online solution.

Structured-prediction problem: The encoder algorithm φ_Θ calculates the weights on the graph representation of the extended online instance. The encoder calculates the weights w_j on arc a_j via a linear combination of parameters Θ and features $\phi(a_j, \Gamma^{extOn})$, e.g., location or request information, describing this arc:

$$w_j = \langle \Theta | \phi(a_j, \Gamma^{extOn}) \rangle \quad \forall j \in \{1, ..., m\}. \tag{1}$$

This allows us to reformulate algorithm $\mathcal{A}$ as a minimization problem:

$$\arg\min_{y \in \mathcal{Y}} \langle \Theta | \phi(y, \Gamma^{extOn}) \rangle \quad \text{with} \quad \phi(y, \Gamma^{extOn}) = \sum_{a_j \in y} \phi(a_j, \Gamma^{extOn}). \tag{2}$$

Structured-learning problem: We formulate a structured learning problem,

$$\min_{\Theta} \sum_{i=1}^{n} L(\Theta; y_i, x_i), \tag{3}$$

that learns parameters Θ such that the transformation from Γ^{on} to Γ^{extOn} leads to a solution x^{extOn} that minimizes the distance to x^{off}. We use a Fenchel-Young loss $L(\Theta; y, x)$ with perturbations $\mathcal{Z}$ and excluded dual summand similar to Parmentier & t'Kindt (2021),

$$L(\Theta; y, x) = \mathbb{E}_{\mathcal{Z}}(\min_{y \in \mathcal{Y}} \langle \Theta + \mathcal{Z} | \phi(y; x) \rangle) - \langle \Theta | \phi(y; x) \rangle, \tag{4}$$

leading to a convex optimization problem that we solve with the BFGS algorithm.

4 PRELIMINARY RESULTS

We apply our algorithmic framework on a real-world case study in Manhattan using the New York City taxi data set. We compare our structured learning based algorithm (SLBA) against two benchmarks, a *naive online approach*, and a *sampling online approach*. The naive approach, does

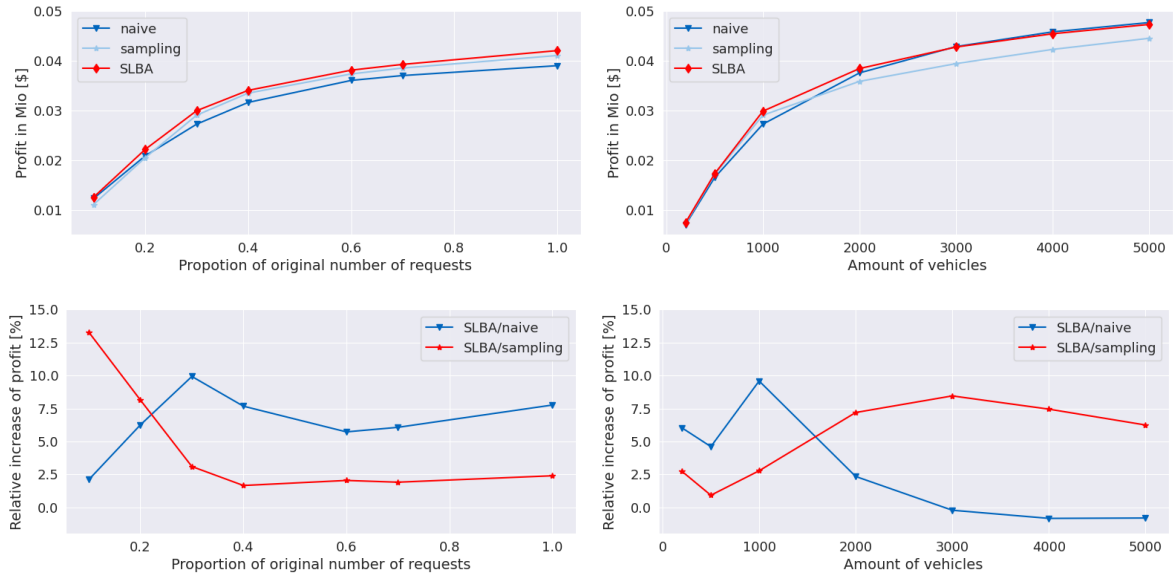


Figure 3 – *The x-axis describes different scenarios (left: different densities of request distributions, right: different amount of vehicles over a fixed amount of requests). The y-axis shows the performance indicator (upper row: absolute profit of different benchmarks, lower row: the relative improvement of our learning approach to the benchmark approach.*

not consider information about future ride requests, whereas the sampling approach incorporates artificial ride requests, sampled from a given distribution, into the dispatching .

Figure 3 compares these benchmarks with our approach. Besides the stable and superior performance of our approach, the results show the pitfalls of both benchmarks: the naive approach outperforms the sampling approach for scenarios with sparse request distributions, where sampling likely triggers suboptimal rebalancing decisions, and for scenarios with a high vehicles to requests ratio, where the vehicle fleet can always fulfill all ride requests without additional rebalancing. The sampling approach outperforms the naive approach on scenarios with a high request density, where sampling from the predefined distribution is a good indicator for future requests, and in the case with a low ratio of vehicles to requests when wrong rebalancing decisions are unlikely. Our SLBA performs better or equally good than both benchmarks across all analyzed settings, which shows the superiority of our imitation learning paradigm, which allows for effective prescriptive rebalancing in various system states.

Acknowledgements: This work was supported by the German Research Foundation (DFG) under grant no. 449261765.

References

Alonso-Mora, J., Wallar, A., & Rus, D. 2017. Predictive routing for autonomous mobility-on-demand systems with ride-sharing. *Pages 3583–3590 of: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Bertsimas, D., Jaillet, P., & Martin, S. 2019. Online vehicle routing: The edge of optimization in large-scale applications. *Operations Research*, **67**(1), 143–162.

Parmentier, A. 2021. Learning to Approximate Industrial Problems by Operations Research Classic Problems. *Operations Research*.

Parmentier, A., & t'Kindt, V. 2021. Learning to solve the single machine scheduling problem with release times and sum of completion times. *arXiv preprint arXiv:2101.01082*.

Schiffer, M., Hiermann, G., Rüdel, F., & Walther, G. 2021. A polynomial-time algorithm for user-based relocation in free-floating car sharing systems. *Transportation Research Part B: Methodological*, **143**, 65–85.