Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence (IJCAI-23)
Journal Track

Reinforcement Learning from Optimization Proxy for Ride-Hailing Vehicle Relocation (Extended Abstract)*

Enpeng Yuan , Wenbo Chen , Pascal Van Hentenryck
Georgia Institute of Technology
{eyuan8, wchen616, pvh}@gatech.edu

Abstract

Idle vehicle relocation is crucial for addressing demand-supply imbalance that frequently arises in the ride-hailing system. Current mainstream methodologies - optimization and reinforcement learning - suffer from obvious computational drawbacks. Optimization models need to be solved in real-time and often trade off model fidelity (hence quality of solutions) for computational efficiency. Reinforcement learning is expensive to train and often struggles to achieve coordination among a large fleet. This paper designs a hybrid approach that leverages the strengths of the two while overcoming their drawbacks. Specifically, it trains an optimization proxy, i.e., a machine-learning model that approximates an optimization model, and refines the proxy with reinforcement learning. This Reinforcement Learning from Optimization Proxy (RLOP) approach is efficient to train and deploy, and achieves better results than RL or optimization alone. Numerical experiments on the New York City dataset show that the RLOP approach reduces both the relocation costs and computation time significantly compared to the optimization model, while pure reinforcement learning fails to converge due to computational complexity.

1 Introduction

The rapid growth of ride-hailing markets has transformed urban mobility, offering on-demand mobility services via mobile applications. While major ride-hailing platforms such as Uber and Didi leverage centralized dispatching algorithms to find good matching between drivers and riders, operational challenges persist due to frequent imbalances between demand and supply. Consider morning rush hours as an example: most trips originate from residential areas to business districts where a large number of vehicles accumulate and remain idle. Relocating these vehicles back to the demand areas is crucial to maintaining quality of service and income for the drivers.

Extensive studies have focused on vehicle relocation problems in real time. Existing methodologies fit broadly into two categories: model-based and model-free approaches. Model-based approaches, e.g., Model Predictive Control (MPC) involves the solving of an optimization program using expected demand and supply information over a future horizon [Miao *et al.*, 2015; Miao *et al.*, 2017; Iglesias *et al.*, 2017; Tsao *et al.*, 2018; Riley *et al.*, 2020]. Model-free approaches (predominantly Reinforcement Learning) train a state-based decision policy by interacting with the environment and observing the rewards [Verma *et al.*, 2017; Lin *et al.*, 2018; Wen *et al.*, 2017; Holler *et al.*, 2019; Oda and Joe-Wong, 2018; Guériau and Dusparic, 2018; Lin *et al.*, 2018; Jiao *et al.*, 2021]. While both approaches have demonstrated promising performance in simulation and (in some cases) real-world deployment [Jiao *et al.*, 2021], they have obvious drawbacks: the optimization needs to be solved in real-time and often trades off model fidelity (and hence solution quality) for computational efficiency. Reinforcement learning does not require a model but needs a large number of samples to train. Consequently, most work simplifies the problem (e.g., by restricting relocations to nearby regions and/or limiting coordination among the fleet) to reduce computational complexity.

This paper addresses these challenges by proposing a Reinforcement Learning from Optimization Proxy (RLOP) approach that combines optimization, supervised learning, and reinforcement learning. The RLOP framework is a special case of Reinforcement Learning from Expert Demonstration (RLED) where the expert is an optimization algorithm [Ramírez *et al.*, 2021]. The RLOP has two main steps:

1. It first applies supervised learning to obtain an *optimization proxy* for a relocation optimization, i.e., it trains a machine learning model that approximates the mapping between the inputs of the optimization and its actionable decisions.
2. It then seeds an RL component with the optimization proxy as the initial policy. The RL component further improves this policy by interacting with the environment and capturing the real system dynamics and long-term effects that are beyond the capabilities of the model-based optimization.

To the best of the authors' knowledge, this paper is the first application of an RLED framework to tackle vehicle relo-

*The full paper has been published at Journal of Artificial Intelligence Research [Yuan *et al.,* 2022]

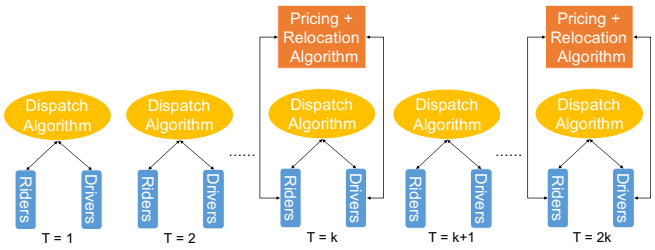

Figure 1: The Real-Time Ride-Hailing Operations.

cation problems, and one of the few RL models with a fully centralized policy. The only other paper using a centralized formulation only demonstrates their approach in a simplified setting due to computational complexity [Mao *et al.*, 2020].

The RLOP framework has three important benefits. First, The optimization proxy approximates the model-based optimization with high fidelity and is order of magnitude faster. Second, The RL component is significantly easier to train since it starts with a high-quality policy. Third, The RL component improves the optimization proxy by capturing long-term effects and real system dynamics present in sample trajectories.

The application of RLOP framework to relocation problems comes with several challenges. First, the relocation decisions are typically high-dimensional (e.g., number of vehicles to relocate between each zone) and sparse. This creates great difficulty for supervised and reinforcement learning. Second, the predicted relocation decisions may be infeasible since most learning algorithms cannot enforce integrality or physical constraints that the decisions need to satisfy.

To tackle these challenges, this paper proposes an *aggregation-restoration-disaggregation procedure*. It predicts the relocation decisions at an aggregated level, restores them back to feasible solutions, and then disaggregates them to the original granularity by applying a *polynomial-time* transportation optimization. As a result, the dimensionality and sparsity of the decisions are reduced considerably, and the approach remains computationally efficient.

The proposed RLOP framework is evaluated on the New York Taxi data set. The experimental results show that RLOP reduces the relocation costs by 10% while being order of magnitude faster than pure optimization and RL. These results suggest that the RLOP framework provides a promising approach for the real-time operations of ride-hailing systems. It is also important to stress that the RLOP framework is general and can be applied with any relocation optimization and RL techniques.

2 Problem Definition

The real-time ride-hailing system, depicted in Figure 1, has three key components: vehicle routing, idle vehicle relocation, and dynamic pricing. The vehicle routing algorithm matches requests to vehicles and chooses the vehicle routes. It operates at the individual request level with high frequency (e.g., every $15 - 60$ seconds). Because of the tight time constraints and large number of requests, the routing algorithm is usually myopic, taking only the current demand and supply into account. Idle vehicle relocation and dynamic pricing, on the other hand, are forward-looking in nature. Idle vehicle relocation repositions the vehicles preemptively to anticipate demand, and dynamic pricing balances expected demand and supply in a future horizon. The two decisions also take place at a lower frequency (e.g., every $5 - 20$ minutes). This paper focuses on idle vehicle relocation and abstracts away the other two components. The goal is to reduce rider waiting time by relocating idle vehicles while accounting for the relocation costs. This paper assumes that the ride-hailing platform uses a fleet of autonomous vehicles or their own pool of drivers who follow instructions exactly - the platform can thus relocate the vehicles at will.

3 The RLOP Framework

The RLOP framework has two stages: supervised learning and reinforcement learning. The supervised-learning stage trains an optimization proxy, i.e., a machine-learning model that approximates the actionable decisions of an optimization model. The reinforcement-learning stage takes the optimization proxy as the initial policy and refines it by a policy gradient method.

3.1 The Optimization Proxy

The supervised-learning stage trains a machine-learning model to predict the actionable decisions of a zone-level relocation model $\mathcal{M} : \mathcal{S} \to \mathcal{W}$ where $\mathcal{S}$ is the model input and $\mathcal{W}$ is the relocation decision, i.e., the number of vehicles to relocate between each zone in the dispatch area. Hence $|\mathcal{W}| = |\mathcal{Z}|^2$, where $\mathcal{Z}$ is the set of zones in the dispatch area. The training data can be generated by running $\mathcal{M}$ on a set of problem instances and extracting its results. *It is important to stress that the framework is general and can work with any relocation model as long as the decisions are at the zone-to-zone level.*

The machine-learning model takes the optimization model's input $\mathcal{S}$ and predicts its relocation decisions $\mathbf{w} = [x^r_{ij}]_{i,j \in \mathcal{Z}}$ - number of vehicles to relocate between each zone pair (i, j). In reality, $\mathbf{w}$ is high-dimensional ($|\mathcal{W}| = |\mathcal{Z}|^2$) and sparse, since most vehicles relocate to a few high-demand zones. The high-dimensionality and sparsity makes supervised learning difficult. It also imposes significant challenges for RL in the second stage since sampling in high-dimensional action space is expensive and makes the training unstable. Therefore, this paper designs an aggregation-disaggregation procedure - it predicts $\mathbf{w}$ at the aggregated (zone) level and then disaggregates the predictions via an efficient optimization procedure.

More precisely, the zone-level relocation decision $\mathbf{a} \in \mathcal{A}$ - number of vehicles to relocate into and out of each zone $i \in \mathcal{Z}$ - is predicted by a machine-learning model $\hat{\mathcal{O}}_\theta : \mathcal{S} \to \mathcal{A}$, rounded and restored to a feasible solution, and disaggregated to zone-to-zone level by a transportation optimization problem $\mathcal{TO} : \mathcal{A} \to \mathcal{W}$. Since $\mathcal{TO}$ can be solved in polynomial time, the procedure remains computationally efficient. Readers are referred to the original paper for a detailed description of this procedure [Yuan *et al.*, 2022]. To ensure that the machine-learning model can be refined by the policy gradient

method in the RL stage, $\hat{\mathcal{O}}_\theta$ needs to be **differentiable** with respect to its parameters θ. For example, $\hat{\mathcal{O}}_\theta$ can be an artificial neural network or a linear regression parametrized by θ. The RLOP framework however is general and can accommodate any other machine-learning model.

3.2 Reinforcement Learning

The supervised-learning stage trains an optimization proxy $\hat{\mathcal{O}}_\theta : \mathcal{S} \to \mathcal{A}$ from an relocation model. The RL process starts from $\hat{\mathcal{O}}_\theta$ and improves it by a policy gradient method. Specifically, the RL step models the relocation problem as a Markov Decision Process (MDP). MDP is characterized by a state space $\mathcal{S}$, an action space $\mathcal{A}$, a reward function $R(\mathbf{s}, \mathbf{a})$, a transition function $P(\mathbf{s}'|\mathbf{s}, \mathbf{a})$, and a discount factor $\gamma \in [0, 1]$. The goal is to find a stochastic decision policy $\pi_\theta : \mathcal{S} \to \mathcal{P}(\mathcal{A})$ parametrized by θ, i.e., a mapping from the state space to a probability distribution over the action space, that maximizes the total expected discounted reward

$$J(\theta) = \mathbb{E}_{P,\pi_\theta}\left[\sum_{t=0}^{T_e} \gamma^t R(\mathbf{s}_t, \mathbf{a}_t)\right] \tag{1}$$

For the present application, the state and action space are the same as the input and output space of the optimization proxy $\hat{\mathcal{O}}_\theta : \mathcal{S} \to \mathcal{A}$ so that $\hat{\mathcal{O}}_\theta$ can be transformed into an initial policy for RL. The details of this transformation will be presented shortly. The reward function $R(\mathbf{s}_t, \mathbf{a}_t) = -u_t - \beta v_t$ is a weighted average of customer satisfaction and system cost, where u_t is the total waiting time of riders who emerges in epoch t, v_t is the expected time that vehicles will relocate due to action $\mathbf{a}_t$, and β is a hyperparameter.

The policy is trained iteratively based on the policy gradient theorem [Sutton and Barto, 2018]

$$\nabla_\theta J(\theta) = \mathbb{E}_{P,\pi_\theta}\left[\sum_{t=0}^{T_e} G_t \nabla_\theta \log P_{\pi_\theta}(\mathbf{a}_t|\mathbf{s}_t)\right] \tag{2}$$

where $G_t = \sum_{\tau=t}^{T_e} \gamma^{\tau-t} R_\tau$ is the total (discounted) reward since epoch t in the trajectory $\tau = (\mathbf{s}_0, \mathbf{a}_0, R_0, ..., \mathbf{s}_{T_e}, \mathbf{a}_{T_e}, R_{T_e})$ and $P_{\pi_\theta}(\mathbf{a}_t|\mathbf{s}_t)$ is the probability of taking action $\mathbf{a}_t$ in state $\mathbf{s}_t$ under the decision policy π_θ. In reality, the expectation term in (2) is intractable to compute and is approximated by Monte-Carlo sampling.

It remains to specify how the optimization proxy $\hat{\mathcal{O}}_\theta$ can be turned into an initial policy for RL. Recall that $\hat{\mathcal{O}}_\theta : \mathcal{S} \to \mathcal{A}$ is a deterministic mapping from the state space to the action space. RL starts from a Gaussian policy $\pi^0_\theta(\cdot) = \mathcal{N}(\hat{\mathcal{O}}_\theta(\cdot), \Sigma)$ centered around $\hat{\mathcal{O}}_\theta$ with covariance Σ. The covariance matrix Σ is a diagonal matrix whose diagonal entry Σ_{ii} is the (sampling) variance of an relocation action a_i (a_i is an entry of $\mathbf{a} \in \mathcal{A}$). Note that a_i is one of the prediction labels of $\hat{\mathcal{O}}_\theta$, so its empirical distribution can be estimated in the supervised-learning stage. Therefore, Σ_{ii} can be taken as a certain percentage of a_i's characteristic statistics such as its empirical mean or median in the supervised-learning dataset. Prior knowledge on Σ is extremely valuable since a well-chosen Σ can lead to a more efficient exploration during training.

Algorithm 1: RLOP

```
Train a differential optimization proxy Ô_θ to approximate a given relocation model;
Choose learning rate α, discount factor γ, trade-off parameter β and covariance Σ ;
for Episode = 1, 2, ... do
    for i = 1, ..., N do
        for t = 0, 1, ...T_e do
            Observe current state s_t^i and sample an action a_t^i from current policy
            π_θ(s_t^i) = N(Ô_θ(s_t^i), Σ);
            Round and disaggregate a_t^i to feasible zone-to-zone level action w_t^i by
            transportation optimization TO;
            Implement w_t^i in the simulator and observe reward R_t^i;
        end
    end
    Compute the policy gradient ∇_θ J(θ) by Eq (2) ;
    θ ← θ + α∇_θ J(θ)
end
```

The policy gradient algorithm is summarized in Algorithm 1. Note that, after sampling action $\mathbf{a}$ from π_θ, $\mathbf{a}$ should be rounded and restored to zone-to-zone level by the transportation optimization $\mathcal{TO} : \mathcal{A} \to \mathcal{W}$. Again, note that the RLOP framework is general and can incorporate any specific reinforcement-learning techniques (e.g., actor-critic, PPO, off-policy sampling, etc.) appropriate for the problem at hand.

4 Simulation Study

The RLOP framework is evaluated on Yellow Taxi Data in Manhattan, New York City [NYC, 2019]. It is trained from $2017/01$ to $2017/05$ and evaluated in $2017/06$, 8am - 9am of weekdays, when the demand is at its peak and the need for relocation the greatest. The experiments use the end-to-end simulation framework in [Riley *et al.*, 2020]. The simulator has two main components: a ride-sharing routing algorithm and a relocation MPC model. The routing algorithm batches riders into a time window and optimizes every 30 seconds. The relocation MPC model is executed every 5 minutes. It partitions the Manhattan area into 60 zones and time into 5-minute epochs. All models must be executed in the 30 seconds batch window.

4.1 The Optimization Proxy

The optimization proxy approximates the MPC model in [Riley *et al.*, 2020]. It is trained from $2017/01$ to $2017/05$. These daily instances are run by the simulator and the MPC model's inputs and outputs are extracted as training data. In total, $15,000$ data points are used in training and 2500 data points are held out for testing.

	Lasso	MLP	LSTM	Transformer
MSE	15.90	6.68	6.64	6.45

Table 1: Testing Loss of Machine Learning Models.

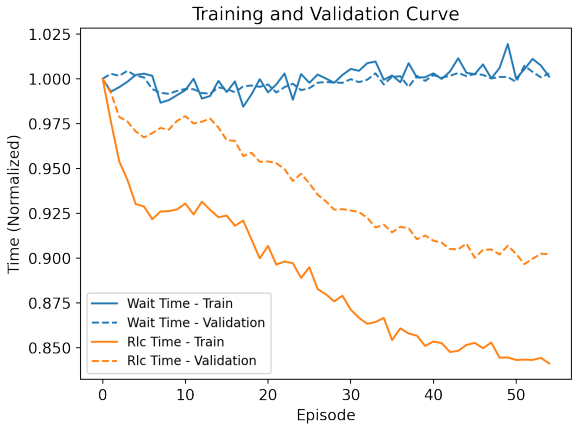

Figure 2: Training and Validation Curve of Reinforcement Learning (Normalized).

Several machine learning models are trained to learn the relocation decisions. The model inputs are expected demand D_{it}, expected supply V_{it}, and expected vehicle shortage $(D_{it}-V_{it})$ in each zone i and epoch t in the MPC horizon. The target is zone-level relocation decisions $\mathbf{a}$. The testing loss is given in Table 1. MLP (multi-layer perceptron) is selected as the final model by virtue of good performance and fewer parameters.

4.2 Reinforcement Learning

The optimization proxy is refined by reinforcement learning in 2017/05. Since the number of riders in most daily instances ranges from 22,000 to 29,000, four instances with [23960, 25768, 27117, 28312] riders are selected and the policy is trained on these representative instances. Algorithm 1 with the baseline is run with $\alpha = 0.005$, $\beta = 0.75$ and $\gamma = 0.75$. The sampling variance Σ_{ii} is taken as $0.05a_i^{0.75}$ where $a_i^{0.75}$ is the 75th percentile of action a_i in the supervised-learning data set (recall that a_i is a prediction label for the optimization proxy). To make sure that RL does not overfit on the selected representative instances, the policy is validated on other instances in 2017/05 after each training episode and the algorithm stops when the average reward on the validation set fails to improve for 5 consecutive episodes. The training and validation curves (broken down into waiting and relocation time) in Figure 2 show that the relocation costs drop dramatically, while the waiting times stay about the same. The algorithm converges in 55 episodes: the training is significantly more efficient computationally than pure reinforcement learning algorithms, which typically converge in tens of thousands of episodes.

	Avg Wait Time	Avg Rlc. Time	Avg Run Time
MPC	2.21	4.03	1.603
Opt. Proxy	2.18	4.06	0.016
RLOP	2.21	3.62	0.019

Table 2: Summary Statistics of Tested Models.

4.3 Evaluation Results

The trained policy is evaluated on weekdays in 2017/06. The proposed RLOP approach is compared with the optimization proxy as well as the MPC optimization. Pure reinforcement learning without an initial policy seeded with the optimization proxy (Algorithm 1 without step 1) fails to converge due to the high-dimensional state and action spaces: it is too expensive computationally to be applied in this setting. Table 2 reports average rider waiting time and vehicle relocation time (in minutes) as well as model run time (in secondss). RLOP achieves similar rider waiting time as the other two models but with less relocation cost. In particular, its relocation time is 10.1% lower than the MPC and 10.8% lower than the optimization proxy. Moreover, the optimization proxy and the RLOP are much faster than the MPC and are guaranteed to run in polynomial time. The longest MPC instance takes 9.73s while the optimization proxy and the RLOP framework remain within fractions of a second on all instances. The main computational cost of the RLOP framework lies in the offline stage where data for supervised learning and RL are generated through simulation. Nevertheless, RLOP is still more efficient than RL which requires a prohibitively large number of samples to train. The optimization proxy did slightly better than the MPC on certain metrics since the MPC optimization is based on an approximation of the ride-sharing system - its decisions are optimal for the approximation but not necessarily for the real system. *Overall, these promising results show that the RLOP is an efficient and effective approach for idle vehicle relocation in real-time settings.*

5 Conclusion

Preemptively relocating idle vehicles is crucial for addressing demand-supply imbalance that frequently arises in the ride-hailing system. Current mainstream methodologies - optimization and reinforcement learning - suffer from computational complexity in either offline training or online deployment. This paper proposes a reinforcement learning from Optimization Proxy (RLOP) approach to alleviate their computational burden and search for better policies. It trains a machine-learning policy to approximate an optimization model and then refines the policy by reinforcement learning. To reduce dimensionality and sparsity of the prediction and action space, this paper presents an aggregation-disaggregation procedure which predicts relocation actions at the aggregated level and disaggregates the predictions via a polynomial-time optimization. On the New York City dataset, the RLOP approach achieves significantly lower relocation costs and computation time compared to the optimization model, while pure reinforcement learning is too expensive computationally for practical purposes.

Acknowledgments

This research is partly supported by NSF Awards 2112533 and 1854684. Many thanks to Professor Yao Xie for insightful comments on the paper.

References

[Guériau and Dusparic, 2018] Maxime Guériau and Ivana Dusparic. Samod: Shared autonomous mobility-on-demand using decentralized reinforcement learning. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 1558–1563, 2018.

[Holler *et al.*, 2019] John Holler, Risto Vuorio, Zhiwei (Tony) Qin, Xiaocheng Tang, Yan Jiao, Tiancheng Jin, Satinder Singh, Chenxi Wang, and Jieping Ye. Deep reinforcement learning for multi-driver vehicle dispatching and repositioning problem. In Jianyong Wang, Kyuseok Shim, and Xindong Wu, editors, *2019 IEEE International Conference on Data Mining, ICDM 2019, Beijing, China, November 8-11, 2019*, pages 1090–1095. IEEE, 2019.

[Iglesias *et al.*, 2017] Ramón Iglesias, Federico Rossi, Kevin Wang, David Hallac, Jure Leskovec, and Marco Pavone. Data-driven model predictive control of autonomous mobility-on-demand systems. *CoRR*, abs/1709.07032, 2017.

[Jiao *et al.*, 2021] Yan Jiao, Xiaocheng Tang, Zhiwei Qin, Shuaiji Li, Fangfang Zhang, Hongtu Zhu, and Jie ping Ye. Real-world ride-hailing vehicle repositioning using deep reinforcement learning. *ArXiv*, abs/2103.04555, 2021.

[Lin *et al.*, 2018] Kaixiang Lin, Renyu Zhao, Zhe Xu, and Jiayu Zhou. Efficient large-scale fleet management via multi-agent deep reinforcement learning. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '18, page 1774–1783, New York, NY, USA, 2018. Association for Computing Machinery.

[Mao *et al.*, 2020] Chao Mao, Yulin Liu, and Zuo-Jun (Max) Shen. Dispatch of autonomous vehicles for taxi services: A deep reinforcement learning approach. *Transportation Research Part C: Emerging Technologies*, 115:102626, 2020.

[Miao *et al.*, 2015] Fei Miao, Shan Lin, Sirajum Munir, John Stankovic, Hua Huang, Desheng Zhang, Tian He, and George Pappas. Taxi dispatch with real-time sensing data in metropolitan areas — a receding horizon control approach. *IEEE Transactions on Automation Science and Engineering*, 13, 04 2015.

[Miao *et al.*, 2017] Fei Miao, Shuo Han, Abdeltawab M. Hendawi, Mohamed E Khalefa, John A. Stankovic, and George J. Pappas. Data-driven distributionally robust vehicle balancing using dynamic region partitions. In *2017 ACM/IEEE 8th International Conference on Cyber-Physical Systems (ICCPS)*, 2017.

[NYC, 2019] NYC. NYC Taxi & Limousine Commission - trip record data. https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page, 2019. Accessed: 2020-10-01.

[Oda and Joe-Wong, 2018] Takuma Oda and Carlee Joe-Wong. Movi: A model-free approach to dynamic fleet management. In *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, pages 2708–2716, 2018.

[Ramírez *et al.*, 2021] Jorge Ramírez, Wen Yu, and Adolfo Perrusquía. Model-free reinforcement learning from expert demonstrations: a survey. *Artificial Intelligence Review*, 1(1), 2021.

[Riley *et al.*, 2020] Connor Riley, Pascal Van Hentenryck, and Enpeng Yuan. Real-time dispatching of large-scale ride-sharing systems: Integrating optimization, machine learning, and model predictive control. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 4417–4423. International Joint Conferences on Artificial Intelligence Organization, 7 2020.

[Sutton and Barto, 2018] Richard Sutton and Andrew Barto. *Reinforcement Learning: An Introduction*. The MIT Press., Cambridge, Massachusetts., 2018.

[Tsao *et al.*, 2018] Matthew Tsao, Ramon Iglesias, and Marco Pavone. Stochastic model predictive control for autonomous mobility on demand. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 3941–3948, 2018.

[Verma *et al.*, 2017] Tanvi Verma, Pradeep Varakantham, Sarit Kraus, and Hoong Chuin Lau. Augmenting decisions of taxi drivers through reinforcement learning for improving revenues. In *ICAPS*, 2017.

[Wen *et al.*, 2017] Jian Wen, Jinhua Zhao, and Patrick Jaillet. Rebalancing shared mobility-on-demand systems: A reinforcement learning approach. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 220–225, 2017.

[Yuan *et al.*, 2022] Enpeng Yuan, Wenbo Chen, and Pascal Van Hentenryck. Reinforcement learning from optimization proxy for ride-hailing vehicle relocation. *J. Artif. Intell. Res.*, 75:985–1002, 2022.