A Learning-Based Optimization Approach for Autonomous Ridesharing Platforms with Service-Level Contracts and On-Demand Hiring of Idle Vehicles

Beirigo, Breno A.; Schulte, Frederik; Negenborn, Rudy R.

DOI
10.1287/trsc.2021.1069

Publication date
2022

Document Version
Final published version

Published in
Transportation Science

Transportation Science

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

A Learning-Based Optimization Approach for Autonomous Ridesharing Platforms with Service-Level Contracts and On-Demand Hiring of Idle Vehicles

Breno A. Beirigo, Frederik Schulte, Rudy R. Negenborn

To cite this article:
Breno A. Beirigo, Frederik Schulte, Rudy R. Negenborn (2022) A Learning-Based Optimization Approach for Autonomous Ridesharing Platforms with Service-Level Contracts and On-Demand Hiring of Idle Vehicles. Transportation Science 56(3):677-703. https://doi.org/10.1287/trsc.2021.1069

Published in Transportation Science on October 13, 2021 as DOI: 10.1287/trsc.2021.1069.
This article has not been copyedited or formatted. The final version may differ from this version.

http://pubsonline.informs.org/journal/trsc

TRANSPORTATION SCIENCE
Vol. 56, No. 3, May–June 2022, pp. 677–703
ISSN 0041-1655 (print), ISSN 1526-5447 (online)

A Learning-Based Optimization Approach for Autonomous Ridesharing Platforms with Service-Level Contracts and On-Demand Hiring of Idle Vehicles

Breno A. Beirigo,[a] Frederik Schulte,[a] Rudy R. Negenborn[a]

[a] Maritime and Transport Technology (MTT), Mechanical, Maritime and Materials Engineering (3mE), Delft University of Technology, 2628 CD Delft, Netherlands

Contact: b.alvesbeirigo@tudelft.nl, https://orcid.org/0000-0002-7584-2136 (BAB); f.schulte@tudelft.nl, https://orcid.org/0000-0003-3159-4393 (FS); r.r.negenborn@tudelft.nl, https://orcid.org/0000-0001-9784-1225 (RRN)

Received: February 1, 2020
Revised: November 24, 2020
Accepted: January 23, 2021
Published Online in Articles in Advance: October 13, 2021

https://doi.org/10.1287/trsc.2021.1069

Abstract. Current mobility services cannot compete on equal terms with self-owned mobility products concerning service quality. Because of supply and demand imbalances, ridesharing users invariably experience delays, price surges, and rejections. Traditional approaches often fail to respond to demand fluctuations adequately because service levels are, to some extent, bounded by fleet size. With the emergence of autonomous vehicles, however, the characteristics of mobility services change and new opportunities to overcome the prevailing limitations arise. In this paper, we consider an autonomous ridesharing problem in which idle vehicles are hired on-demand in order to meet the service-level requirements of a heterogeneous user base. In the face of uncertain demand and idle vehicle supply, we propose a learning-based optimization approach that uses the dual variables of the underlying assignment problem to iteratively approximate the marginal value of vehicles at each time and location under different availability settings. These approximations are used in the objective function of the optimization problem to dispatch, rebalance, and occasionally hire idle third-party vehicles in a high-resolution transportation network of Manhattan, New York City. The results show that the proposed policy outperforms a reactive optimization approach in a variety of vehicle availability scenarios while hiring fewer vehicles. Moreover, we demonstrate that mobility services can offer strict *service-level contracts* to different user groups featuring both delay and rejection penalties.

History: This paper has been accepted for the *Transportation Science* Special Issue on Transportation in the Sharing Economy.

Funding: This publication is part of the project "Dynamic Fleet Management" [Project Number 14896 P14-18, Project 3] of the research programme i-CAVE, which is (partly) financed by the Dutch Research Council (NWO), domain Applied and Engineering Sciences (TTW).

Keywords: autonomous ridesharing platform • stochastic heterogeneous demand • stochastic vehicle supply • machine learning • approximate dynamic programming • service-level contracts • on-demand hiring

1. Introduction

Mobility-on-demand (*MoD*) platforms and *transportation network companies* (*TNCs*) such as Uber and Lyft have grown substantially and altered mobility behavior worldwide. Although envisioned to make vehicle ownership superfluous and, eventually, alleviate congestion, these ride-hailing platforms have primarily won customers from traditional public transport modes (Castiglione et al. 2018). Ultimately, MoD solutions can only challenge vehicle ownership if service providers can offer high service levels consistently.

Sufficient vehicle supply is a critical factor when it comes to providing consistent service levels efficiently and sustainably. However, most existing models for ridesharing are not capable of responding quickly to significant demand changes. First, often fixed fleet sizes are assumed, which makes it hard to react to demand fluctuations on a tactical level, let alone in real time. Second, when providers rely on third-party vehicles (i.e., independent drivers), they typically balance supply and demand using surge prices: fares at undersupplied areas dynamically increase both to attract more drivers and suppress excessive demand. Such a strategy, however, is highly controversial because it mainly benefits the platform at the expense of drivers and riders (Xu, Yin, and Ye 2020). Regardless of the strategy, some customers end up being penalized with excessive delays, abusive prices, and rejections. With the emergence of *autonomous vehicles* (*AVs*), however, new possibilities to overcome the shortcomings caused by demand–supply imbalances arise. As soon as vehicle availability is detached from driver availability, ridesharing platforms can count on a larger pool of vehicles, which, currently

Published in Transportation Science on October 13, 2021 as DOI: 10.1287/trsc.2021.1069.
This article has not been copyedited or formatted. The final version may differ from this version.

nonautomated, remain parked about 95% of the time (Shoup 2017).

In this study, we consider an *autonomous mobility-on-demand* (*AMoD*) system where a ridesharing platform can occasionally hire *freelance autonomous vehicles* (*FAVs*), that is, idle third-party-owned AVs, to support its own *platform-owned autonomous vehicles* (*PAVs*), fulfilling the demand adequately. Hence, in contrast with related literature, we model a highly diversified mobility system where AV ownership is disseminated among the platform and individuals, who simultaneously own and hire out their vehicles. We refer to this system as the AMoD-H. To guarantee service quality, the platform establishes strict *service-level contracts* (SLCs) with its user base, such that contract violations (e.g., extra delays, rejections) incur penalties. Hence, by harnessing FAV availability, the platform can shorten the minimum size of the own fleet while addressing personalized demand fluctuations in real time.

Modeling the AMoD-H poses several challenges because requests have to be handled dynamically in the face of (i) irregular FAV availability and (ii) uncertain demand. First, whereas fleet availability is mostly taken for granted, we assume that the location, announcement time, and total service duration of freelance vehicles are uncertain. For example, FAV availability may resemble that of future AVs whose owners commute by car to work and decide to rent out their vehicles to an AMoD platform during designated intervals such that they have a chance to profit from otherwise unproductive parking times. Second, analogously to service offers in the aviation and rail industry, we segment users into first and second classes, such that the former is willing to pay a premium to enjoy higher service levels. We consider not only the stochastic trip distribution but also class membership distribution when designing anticipatory rebalancing strategies. Based on such details, platforms can improve decision making by taking into account demand patterns arising within its user base, besides moving forward in the direction of a more personalized user experience. Because we consider a real-world transportation demand setting, determining an optimal policy would incur all "curses of dimensionality," and we are unable to enumerate all possible states and decisions, let alone the uncertainty associated with requests and FAV hiring. We therefore develop an *approximate dynamic programming* (*ADP*; Powell 2011) algorithm using *value function approximations* (*VFAs*). In the proposed approach, the dual variables of the underlying assignment problem, defined through a *mixed integer programming* (*MIP*) formulation, are iteratively used to approximate value functions representing the benefit of having an additional vehicle of either type at a certain location and time. Moreover, particularly for the freelance fleet, such approximations also indicate whether it is worthwhile to engage an FAV in further rebalancing or pickup actions, based on its remaining available time or how far it is from its owner's location. At the same time, VFAs are actively used in the objective function of the MIP formulation to weigh the outcome of present decisions (e.g., vehicle rebalancing, parking, and hiring). Eventually, after a number of iterations and value function updates, these learned approximations more accurately represent future states, such that solution quality improves over time. From a methodological perspective, the approach offers the following:

1. An ADP algorithm for a novel AMoD application that sustains contracted service levels of a heterogeneous user base by controlling vehicle supply on the operational level through on-demand hiring. Requests and third-party AVs arrive stochastically within the service area, such that the platform needs to determine a policy to fulfill the demand using either vehicle type (i.e., PAV or FAV).

2. A hierarchical aggregation structure that summarizes state features using both time and space dimensions. Spatial levels comprise increasingly larger clusters (regional centers) set up according to a minimum coverage set formulation on a high-resolution street network of Manhattan, New York City. In turn, temporal levels conform with the level of responsiveness demanded by modern mobility-on-demand applications, in which decisions (e.g., user–vehicle matching, vehicle dispatching, and rebalancing) need to be derived in short intervals.

3. An online discount function that dampens value function approximations arising from decisions involving multiperiod travel times (i.e., resource transformations that take more than one period). Besides leading to more robust estimations and simplifying the state representation, we show that such a discount function enables more complex rebalancing strategies because vehicles can consider varying distance ranges.

From a managerial viewpoint, we show that our policy addresses the requirements of all stakeholders:

1. Users enjoy personalized service levels and are compensated when these are violated.

2. Cities can impose strict street use regulations, such as the maximum number of cars per intersection, congestion pricing, and parking schemes. This level of control is enabled by our network representation, which is directly anchored to the real-world physical structure.

3. Independent AV owners can profit from their cars' idleness by making them available to join a transportation platform during predefined time windows.

4. AMoD platforms may keep the minimum number of cars necessary to maintain customers' service levels, or instead, rely entirely on the freelance fleet, while maximizing profits.

The outline of this paper is as follows. We present our literature review in Section 2, define the problem

Published in Transportation Science on October 13, 2021 as DOI: 10.1287/trsc.2021.1069.
This article has not been copyedited or formatted. The final version may differ from this version.

in Section 3, and formulate it using the language of dynamic resource management in Section 4. Section 5 presents our approximate dynamic algorithm, and Section 6 lays out the details of our experimental study and analyzes the performance of our method when dealing with several transportation scenarios. Finally, Section 7 concludes the work and presents an outlook for future research.

2. Literature

The goal of this literature review is threefold. First, we identify the underlying *dial-a-ride problem* (*DARP*) our model stems from (Section 2.1). Second, we survey studies on transportation platforms in which the demand (parcels or passengers) is fulfilled both by company- and/or third-party-owned vehicles (Section 2.2). Third, we analyze the mobility-on-demand literature that considers anticipation mechanisms (Section 2.3). We show that our work is the first to address two different sources of uncertainty, namely, third-party vehicle availability and user service levels.

2.1. The Dynamic and Stochastic Dial-a-Ride Problem

In this study, we introduce a generalization of the classic *dynamic and stochastic dial-a-ride problem* (for a comprehensive survey on DARP, see Ho et al. 2018). Regarding the sources of uncertainty linked to our problem, as pointed out by Ho et al. (2018), stochasticity is generally on the side of demand, and rarely on the side of the supply. The lack of studies on supply stochasticity can also be seen across other transportation problems. For instance, in the *vehicle routing problem* (*VRP*) literature, the bulk of stochastic models also focus on uncertain demand features (for a review on stochastic VRPs, see Oyola, Arntzen, and Woodruff 2018), with a few exceptions (e.g., vehicle breakdown). On the other hand, for demand as a source of uncertainty, service-level stochasticity has not yet been explored in the literature. This type of stochasticity arises from a user base with heterogeneous customer profile segments whose transportation patterns, as well as their expectations regarding service quality, differ markedly.

2.2. On-Demand and Crowdsourced Vehicles

In this section, we review studies in which a crowdsourcing platform matches the demand (partially or entirely) to third-party vehicles. Most studies in this category refer to ridesharing or crowd-shipping scenarios, in which a platform seeks to fit riders or parcels into already planned driver routes. Moreover, in contrast with our work, these studies do not consider vehicle automation, such that the availability and preferences of the drivers (rather than the AV owners) have to be taken into account to design feasible routes. Because we focus only on dual-fleet models, the reader may refer to Furuhata et al. (2013) and Le et al. (2019) for comprehensive reviews on ridesharing and crowd-shipping, respectively.

First, regarding the ridesharing scenario, Lee and Savelsbergh (2015) assume dedicated drivers complement the ad hoc fleet, satisfying rider requests that would otherwise remain unmatched. The authors argue that ensuring service levels is essential to retain more participants and investigate the cost-benefit of employing dedicated drivers. Their findings suggest this cost-benefit depends on the number and time flexibility of the participants, as well as on the similarity between their travel patterns. Santos and Xavier (2015) consider a setting where passengers are willing to share both taxis and rides, as long as sharing leads to lower costs than private trips. On the other hand, vehicle owners can reduce costs by servicing multiple passengers on the way to their destination. A *greedy randomized adaptive search procedure* (*GRASP*) heuristic is used to solve the dynamic version of the problem in a realistic scenario, with requests arriving at every minute and private vehicles at every hour throughout a 12-hour horizon. Although the underlying DARP variant they propose is general enough to accommodate our problem, leveraging passenger and vehicle stochasticity is out of the scope of their study. Moreover, following the ridesharing tradition, they assume drivers stop servicing customers as soon as they reach their destination. In contrast, our formulation is closer to the *general pickup and delivery problem* considered by Savelsbergh and Sol (1998), where vehicles are stationed at a home depot, from where they can go back and forth within a designated time window.

Second, regarding the crowd-shipping scenario, Archetti, Savelsbergh, and Speranza (2016) consider that a company can rely on *occasional drivers* (*ODs*) besides their own fleet to deliver goods. After arriving at the company's depot, each OD can make at most one delivery, provided that the extra travel distance required to do so does not violate a flexibility threshold. Arslan et al. (2019) build on Archetti, Savelsbergh, and Speranza's (2016) work by considering ODs that can realize multiple pickup and/or drop-off tasks as long the extra time and number of stops does not inconvenience the drivers. Similarly to traditional ridesharing approaches, however, they assume that the delivery platform relies solely on third-party vehicles and that tasks can be eventually handled by an emergency backup fleet to keep service levels high. Dahle, Andersson, and Christiansen (2017) also extend Archetti, Savelsbergh, and Speranza's (2016) work by assuming that ODs can perform multiple pickup and delivery operations within a time window. Later, Dahle et al. (2019) focus on the design of compensation

schemes that can fulfill ODs personal expectations, which are modeled through threshold constraints. They show that even suboptimal compensation schemes, which do not attract as many ODs, can yield substantial cost savings.

Table 1 offers an alternative view on the pickup and delivery literature where requests can be fulfilled both by dedicated and third-party vehicles. In the first column, papers are subsumed under parcel and passenger categories. In the second column, we identify how authors refer to the provider's fleet and the third-party fleet. We also indicate whether the model considers a multitrip (i.e., vehicles can return to their depot multiple times) or single-trip (i.e., vehicles stop the service as soon as they reach their depot or destination) setting. A checkmark in the "capacity" column indicates each vehicle can handle multiple requests at a time. To highlight how information regarding the third-party vehicles and the customer demand unfolds throughout time, we adopt the standard taxonomy used to classify transportation problems (e.g., VRPs, DARPs). Traditionally, these problems can fall into four categories, namely, static deterministic (SD), dynamic deterministic (DD), static stochastic (SS), and dynamic stochastic (DS). Dynamic or static classes indicate whether new information (e.g., demand, third-party vehicles) can modify existing plans. In turn, deterministic or stochastic classes indicate whether information about the uncertainty (e.g., demand and third-party vehicle distributions) is available at decision time. Finally, the last column shows the method used to solve each problem.

2.3. Stochastic Mobility-on-Demand Problems

Assuming a fleet of centrally controlled (autonomous) vehicles, Alonso-Mora et al. (2017), Vazifeh et al. (2018), and Fagnant and Kockelman (2018) have demonstrated that historical taxi demand could be almost entirely fulfilled with significantly fewer vehicles, especially when passengers are willing to share their rides. Studies have also shown that service levels can be substantially improved through anticipatory rebalancing strategies. For example, demand data have been already successively exploited using frequentist approaches (e.g., Alonso-Mora, Wallar, and Rus 2017), reinforcement learning (e.g., Wen, Zhao, and Jaillet 2017; Gueriau and Dusparic 2018; Lin et al. 2018), model predictive control (e.g., Zhang, Rossi, and Pavone 2016; Iglesias et al. 2018; Tsao, Iglesias, and Pavone 2018), and approximate dynamic programming (e.g., Al-Kanj, Nascimento, and Powell 2020). Most of these approaches, however, dimension fleet size experimentally, by simulating configurations that can service the target demand under predefined minimum service-level requirements. As pointed out by Vazifeh et al. (2018), fleet-size inflation can be required as a consequence of trip-demand bursts, occurring, for instance, after concerts or sports matches. Hyland and Mahmassani (2017, p. 30) refer to this ability to change the fleet size to flex with demand as "fleet size elasticity" and highlight that the benefits of increasing vehicle supply in the short term are likely significant. The authors also point out that, although uncommon within the context of *shared autonomous vehicle* (*SAV*) fleet management research, current TNCs rely entirely on this feature, constantly manipulating prices to attract more drivers. Likewise, vehicle ownership may be highly disseminated in the future autonomous mobility market, with most AVs owned by individuals and small fleet operators rather than a single service provider (Campbell 2018). Although some models are flexible enough to handle dynamic fleet inflation (e.g., Ma, Zheng, and Wolfson 2015; Gueriau and Dusparic 2018), research on short-term SAV fleet

Table 1. Transportation Problems in Which Demand Is (Partially) Fulfilled by Third-Party Vehicles

Reference	Terminology	Third-party fleet supply	Single trip (S)/ multitrip (M)	Capacity	Demand	Method
Parcel transportation						
Archetti, Savelsbergh, and Speranza (2016)	Company vehicle, occasional driver	SD	S		SD	TS
Arslan et al. (2019)	Back-up vehicle, ad hoc driver	DD	S		DD	Heuristic
Dahle, Andersson, and Christiansen (2017)	Company vehicle, occasional driver	DD, DS	S	√	DD	LP, MIP
Dahle et al. (2019)	Company vehicle, occasional driver	DD	S	√	SD	LP, MIP
Savelsbergh and Sol (1998)	Company vehicle, independent driver	SD	M	√	DD	B&P
Passenger transportation						
Lee and Savelsbergh (2015)	Dedicated driver, ad hoc driver	DD	S	√	DD	NS
Santos and Xavier (2015)	Taxi, car owner	SD, DD	S	√	SD, DD	GRASP
This study	Platform AV, freelance AV	DS	M		DS	ADP

Note. B&P, branch and price; LP, linear programming; NS, neighborhood search; TS, Tabu search.

Published in Transportation Science on October 13, 2021 as DOI: 10.1287/trsc.2021.1069.
This article has not been copyedited or formatted. The final version may differ from this version.

size elasticity is still lacking (Narayanan, Chaniotakis, and Antoniou 2020).

3. Problem Description

The AMoD-H emerges on AV-based transportation platforms aiming to fulfill a set of pickup and delivery requests P arising on an urban network $G = (N, E)$, where N is a set of nodes (locations), and E is a set of directed edges (streets). Requests arrive in batches P_t, where all requests $r \in P_t$ have arrived at continuous times in the interval $[t-1, t)$, for discrete time $t \in \mathcal{T} = \{0, 1, 2, 3, \ldots, T\}$. We consider that request arrival follows a known stochastic process $\mathcal{F}^{\mathrm{P}}$ concerned with two sources of uncertainty:

- *Request distribution*: The number of requests, arrival times, and origin-destination nodes depend on user demand patterns.
- *Request class*: Each request is associated with a service quality class $c \in C$ that identifies minimum service-level requirements, particularly maximum pickup delays. Requests render the highest contributions when these requirements are respected, or, otherwise, incur class-dependent waiting and rejection penalties.

To ensure these service quality requirements are met fully, the platform can hire additional vehicles online to address unexpected supply–demand mismatches. The fleet comprises a set of platform-owned autonomous vehicles K^{PAV} and a set of freelance autonomous vehicles K^{FAV}, such that the total fleet set is $K = K^{\mathrm{PAV}} \cup K^{\mathrm{FAV}}$. Whereas PAVs can initiate service at any location $n \in N$, FAVs are distributed throughout a set of locations $O \subseteq N$, where they are typically parked. We refer to locations $o \in O$ as stations because FAVs are required to return to them upon finishing the service contract. Although O is known in advance, the platform deals with three sources of uncertainty (which follow a stochastic process $\mathcal{F}^{\mathrm{O}}$) when dealing with FAVs, namely, the following:

- *Vehicle-station distribution*: Some parking locations can be more prone to accommodate FAVs. For example, vehicles can routinely park in the surroundings of their owners' locations (e.g., workplaces, garages) or in more affordable parking places on the outskirts of the city.
- *Announcement time*: Vehicles are available to pick up users at stations at different times for a given day. For example, FAVs can become available downtown as soon as they drop their owners at work. Alternatively, some owners can make their FAVs available (possibly, from their garage) during the night or over the weekend. Regardless of the case, provided that one's itinerary is somewhat irregular due to external factors (e.g., weather, congestion) or particular preferences (e.g., appointments, company's culture), the announcement time can change. For example, a station that typically accommodates a hundred vehicles can have 20%, 50%, and 30% of them arriving in the intervals [7:00 a.m., 8:00 a.m.), [8:00 a.m., 9:00 a.m.), and [9:00, 10:00 a.m.), respectively.
- *Contract duration*: From the announcement time on, FAV owners make their assets available only during a predefined time interval. Consequently, vehicles must stop servicing users at the right moment, such that they have enough time to travel back to their respective stations before their owner's deadline. Analogously to the announcement times, the contract durations may depend on several factors related to an owner's schedule, leading to varying return deadlines. For example, contracts can be short (e.g., shopping, doctor appointments), average (e.g., office hours, evening), and long (e.g., the whole weekend, vacation).

Finally, over the planning horizon $\mathcal{T}$, the platform aims to maximize the total contribution accrued by adequately servicing the requests while minimizing the operational costs associated with routing and hiring vehicles.

3.1. Example

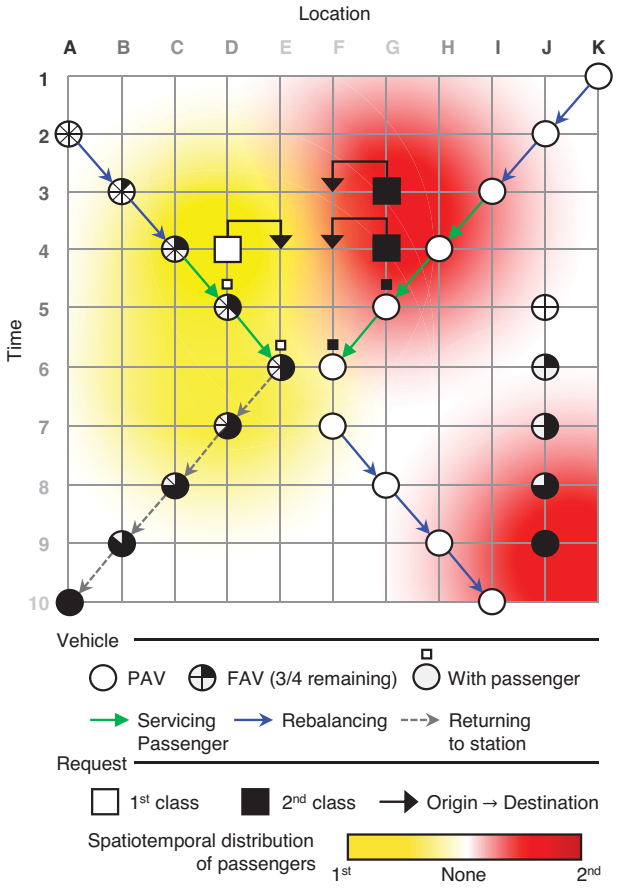
In Figure 1, we illustrate the interplay between the elements of our model. For the sake of simplicity, we represent both vehicle and request discrete locations on a one-dimensional space for each time step such that $N = \{\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D}, \mathrm{E}, \mathrm{F}, \mathrm{G}, \mathrm{H}, \mathrm{I}, \mathrm{J}, \mathrm{K}\}$ and consider a time horizon $\mathcal{T} = \{1, 2, \ldots, 10\}$. We assume it takes a single period to travel between each location pair.

We represent the stochastic process $\mathcal{F}^{\mathrm{P}}$ in time and space by manipulating the transparency of yellow and red colors, corresponding to first- and second-class customers, respectively (see the color bar at the bottom). Regardless of the class, the more opaque the color, the higher the probability of finding a request. We assume first-class requests (i) generate higher profits and (ii) demand higher service levels, such that failing to adhere to their performance requirements incurs higher penalties. We illustrate such higher service levels by assuming first-class users require to be picked up within one period, whereas second-class users are willing to wait up to two periods. In turn, regarding the availability of the freelance fleet, we represent the stochastic process $\mathcal{F}^{\mathrm{O}}$ using the shades of gray on the axis tick labels. We assume that the darker the shade, the higher the chance of an FAV appearing. Additionally, we assume FAV contract durations last on average four time steps.

In the following, we describe the behavior of three vehicles in detail throughout 10 periods. First, at time $t = 1$, the PAV at location K is faced with two decisions; namely, it can stay in its current location or move to a more promising location in anticipation of future demand. Promising areas consist of

Published in Transportation Science on October 13, 2021 as DOI: 10.1287/trsc.2021.1069.
This article has not been copyedited or formatted. The final version may differ from this version.

Figure 1. (Color online) Example of the AMoD-H



high-demand locations that typically generate the highest profits to the service provider. Pursuing such future profits, the PAV performs two rebalance movements, moving empty from K to J, and from J to I. Once it arrives at location I at time $t = 3$, the PAV is assigned to a second-class request (black square) demanding a trip from G to F (solid upper arrow). From this moment on, we consider the PAV is servicing the user, which covers both pickup and delivery times. Once the PAV delivers the second-class user at F, it stays in F for one period, and then rebalances to location I, in anticipation of future passenger demand.

Second, at decision time $t = 2$, an FAV with an eight-period contract duration becomes available at location A and is immediately rebalanced to the high-demand area. Upon arriving at location C, it is matched to a first-class request demanding a trip from D to E. The FAV travels from C to D to pick up the user and finishes the service at location E and time $t = 6$. By this time, the FAV is available for four additional periods but spends this remaining time traveling back to its station at A to comply with the contract deadline.

Finally, at location J and time $t = 5$, a second FAV with a four-period contract duration appears. However, because it cannot reach high-demanding areas in the subsequent periods, this vehicle ends up not being hired by the platform, staying still until the end of its contract at time $t = 9$.

4. Problem Formulation

We model the problem using the language of dynamic resource management (see Simão et al. 2009), where AVs (resources) are servicing subsequent trip request batches (tasks) occurring at discrete times $t \in \{1, 2, \ldots, T\}$. Subsequently, we present the elements of our model: the system state (Section 4.1), information arrival (Section 4.2), decisions (Section 4.3), costs (Section 4.4), and objective function (Section 4.5).

4.1. System State

The state of a single resource is defined by an four-attribute vector a given by

$$a = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{pmatrix} = \begin{pmatrix} \text{Vehicle type} \\ \text{Remaining servicing time} \\ \text{Station} \\ \text{Current location} \end{pmatrix} = \begin{pmatrix} a_{\text{type}} \\ a_{\text{remain}} \\ a_{\text{station}} \\ a_{\text{current}} \end{pmatrix}.$$

We refer to a single vehicle $k \in K$ with attribute a as k^a. First, the vehicle type attribute a_{type} helps distinguishing between third-party-owned, freelance, and platform-owned vehicles. This distinction is crucial because FAVs operate under a stricter availability and different cost plan (owners are entitled to a higher share of the profits).

The duration of such availability, in turn, is captured by the remaining servicing time attribute a_{remain}, which corresponds to the remaining time interval an FAV still can spend servicing orders. PAVs, on the other hand, are assumed to be always available. Thus, as time goes on, an FAV is increasingly unable to pick up new orders, especially those whose destinations are far away from the vehicle's station.

The station attribute a_{station} corresponds to the start and terminal location of each FAV, supposedly, the parking place where the owner expects the vehicle to return once the remaining servicing time has expired. It is worth noting that attributes a_{remain} and a_{station} are not taken into consideration for PAVs; we assume these vehicles are available indefinitely, besides not being obliged to depart from or return to a station.

Finally, similarly to a_{station}, the current location field a_{current} expresses where a vehicle is on the service area. The locations identified by attributes a_{station} and a_{current} integrate the node set N of the street network graph $G = (N, E)$.

By including the temporal dimension, we have a_t, or the attribute vector of an AV at time t. Let $\mathcal{A}$ be the

Published in Transportation Science on October 13, 2021 as DOI: 10.1287/trsc.2021.1069.
This article has not been copyedited or formatted. The final version may differ from this version.

set of all possible vehicle attribute vectors. The state of all vehicles with the same state vector is modeled using

R_{ta} = Number of vehicles with attribute vector a at time t,
$R_t = (R_{ta})_{a\in\mathcal{A}}$ = The resource state vector at time t.

Each request, in turn, is modeled using a three-attribute vector b, given by

$$b = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} \text{Origin} \\ \text{Destination} \\ \text{Class} \end{pmatrix} = \begin{pmatrix} b_{\text{origin}} \\ b_{\text{dest}} \\ b_{\text{class}} \end{pmatrix}.$$

We refer to a single trip $r \in P$ with attribute vector b as r^b. Similarly to vehicle locations, origin (b_{origin}) and destination (b_{dest}) attributes correspond to nodes of the street network (i.e., $b_{\text{origin}}, b_{\text{dest}} \in N$), whereas the b_{class} attribute identifies the requested service quality $c \in C$.

Let $\mathcal{B}$ be the set of all possible request attribute vectors. The state of all rides with the same state vector occurring at time t is modeled using

D_{tb} = The number of trips with attribute vector b at time t,
$D_t = (D_{tb})_{b\in\mathcal{B}}$ = The request state vector at time t.

With the resource and request state vectors, we defined our system state vector as

$$S_t = (R_t, D_t).$$

4.2. Exogenous Information

Although the underlying system is known to evolve continuously over time, we measure states S_t before making any decisions at discrete periods $t \in \{0, 1, 2, 3, \ldots, T\}$. Between subsequent periods $t-1$ and t, we account for the exogenous information processes concerning both vehicle and demand attribute updates using variables

$\hat{R}_{ta}$ = The change in the number of FAVs with attribute a resulting from information arriving between t–1 and t,

$\hat{D}_{tb}$ = The number of new requests with attribute b placed between t–1 and t,

$W_t = (\hat{R}_t, \hat{D}_t)$ = Exogenous information arriving between t–1 and t.

For the complete stochastic process, we let $\omega \in \Omega$ represent the sample path $W_1, W_2, \ldots, W_T$, where Ω is the set of all sample paths.

In this study, we consider that $\hat{R}_{ta}$ is concerned only with FAVs entering the system. However, it could also account for several alternative sources of uncertainty, such as travel delays, vehicle breakdowns, and early termination of FAV contracts. In any case, whenever an AV attribute changes randomly from a to a', we will have $\hat{R}_{ta} = -1$ and $\hat{R}_{ta'} = +1$.

4.3. Decisions

Regarding the types of decisions used to act on the fleet, we assume every vehicle can *service* a user (which is reachable within its maximum pickup delay), *stay parked* in its current location waiting to pick up users, *rebalance* to a more promising location, or, in the case of FAVs, *return* to its station before the contract deadline. Decisions are described using

d^{stay} = Decision to stay parked in the current location,
d^{return} = Decision to return to the station (FAV only),
$\mathcal{D}^{\text{R}}$ = Set of all decisions to rebalance (i.e., move empty) to a set of neighboring locations,
$\mathcal{D}^{\text{S}}$ = Set of all decisions to service a user, where an element $d \in \mathcal{D}^{\text{S}}$ represents the decision to cover a trip request of type $b_d \in \mathcal{B}$,
$\mathcal{D}^{\text{S}}_c$ = Subset of decisions in $\mathcal{D}^{\text{S}}$ associated with each service quality $c \in C$,
$\mathcal{D}$ = Set of all decisions $d \in \mathcal{D}^{\text{S}} \cup \mathcal{D}^{\text{R}} \cup \{d^{\text{stay}}\} \cup \{d^{\text{return}}\}$,
x_{tad} = Number of times decision d is applied to a vehicle with attribute vector a at time t,
$x_t = (x_{tad})_{a\in\mathcal{A}, d\in\mathcal{D}}$ = Decision vector at time t.

Transition Function. To model how decisions affect vehicle states, we consider a deterministic transition function a^M. Hence, before any new information arrives, applying a decision d to a vehicle with attribute vector a at time t leads to a postdecision attribute vector

$$a' = a^M(a, d).$$

In turn, the new time of availability is given by

$$t' = \begin{cases} t+1 & \text{if } d = d^{\text{stay}}, \\ t + \tau(t, a, d), & \text{if } d \in \mathcal{D}^{\text{S}} \cup \mathcal{D}^{\text{R}} \cup \{d^{\text{return}}\}, \end{cases}$$

where $\tau(t, a, d)$ is the travel time spent to carry out a decision d to service a user, rebalance to another location, or return to the station. We consider that between t and t', vehicles are busy, such that the system cannot exert any control over them. Therefore, if, for instance, a decision d to cover a trip b is applied to a vehicle with attribute vector a at time t, this vehicle will end up in state a' (with $a'_{\text{current}} = b_{\text{dest}}$), and can only be used again at t'.

Abiding by the Street Network Capacity. To avoid unrealistic vehicle distributions, we assume locations $j \in N$ can accommodate only up to $k_j^{\max}$ vehicles. In a real-world setting, different locations have different capacities, which may depend not only on the physical infrastructure (e.g., number of parking places), but also on city regulations. An artificial threshold may be imposed, for instance, to alleviate local

Published in Transportation Science on October 13, 2021 as DOI: 10.1287/trsc.2021.1069.
This article has not been copyedited or formatted. The final version may differ from this version.

congestion or improve accessibility to surrounding facilities. To implement this restriction, we keep track of the number of vehicles k_j^{inbound} ($0 \le k_j^{\text{inbound}} \le k_j^{\max}$) inbound to j and assure that at most $k_j^{\max} - k_j^{\text{inbound}}$ extra vehicles can enter j. Furthermore, we define the set of all decisions leading vehicles k^a to postdecision locations j as

$$\mathcal{D}_{a,j} = \{d | a' = a^M(a,d),\ a'_{\text{current}} = j,\ a'_{\text{current}} \neq a'_{\text{station}},\ \forall d \in \mathcal{D} \setminus \mathcal{D}^S\}.$$

Using $\mathcal{D}_{a,j}$ and k_j^{inbound}, we can calculate the postdecision number of vehicles inbound to j (through either rebalance or stay decisions) and ensure the maximum capacity of j is not violated (see Constraints (3)). One must notice that $\mathcal{D}_{a,j}$ does not cover FAVs inbound to their own stations (i.e., $a'_{\text{current}} = a'_{\text{station}} = j$). FAVs are assumed to have free access to their home stations at any time.

Fulfilling Contract Time Windows. An FAV with attribute vector a can only be acted on using a decision d to stay, rebalance, or service users when there is enough remaining servicing time to return to its station, that is,

$$\tau(t,a,d) + \tau(t',a',d^{\text{return}}) \le a_{\text{remain}} \quad \forall a \in \mathcal{A},\ d \in \mathcal{D} - d^{\text{return}}.$$

Otherwise, the decision is deemed to be invalid, and the corresponding x_{tad} variable is preemptively discarded. At each period t, we define the set of vehicle attribute vectors associated to FAVs that must return to their station using

$$\mathcal{A}^{\text{return}} = \{a |\ \forall a \in \mathcal{A}^{\text{FAV}},\ \tau(t,a,d^{\text{return}}) = a_{\text{remain}}\}.$$

Although FAVs will eventually realize the return decision, we consider that they can always return to their stations directly, even before their contract due times. Doing so adds flexibility to FAV operations because they can rebalance back and forth from their stations, when suitable. This way, provided that FAV owners have already covered parking costs at their stations, the platform may find it worthwhile to rebalance FAVs back sometimes, to evade city parking costs.

4.3.4. Constraints. The decision variables x_{tad} must satisfy the following constraints:

$$\sum_{d \in \mathcal{D}} x_{tad} = R_{ta} \qquad \forall a \in \mathcal{A}, \tag{1}$$

$$\sum_{a \in \mathcal{A}} x_{tad} \le D_{tb_d} \qquad \forall d \in \mathcal{D}^S, \tag{2}$$

$$\sum_{a \in \mathcal{A}} \sum_{d \in \mathcal{D}_{a,j}} x_{tad} \le k_j^{\max} - k_j^{\text{inbound}} \qquad \forall j \in N, \tag{3}$$

$$x_{tad^{\text{return}}} = R_{ta} \qquad \forall a \in \mathcal{A}^{\text{return}}, \tag{4}$$

$$x_{tad} \ge 0 \qquad \forall a \in \mathcal{A},\ \forall d \in \mathcal{D}. \tag{5}$$

Constraints (1) guarantee that all available vehicles (i.e., parked at the current period t) are assigned to a decision, whereas Constraints (2) ensure that any trip request (identified by b_d) can be assigned to at most one vehicle. In turn, Constraints (3) enforce that the number of vehicles inbound to a location j do not surpasses j's remaining capacity. Finally, Constraints (4) ensure that every FAV whose returning trip delay $\Delta t(a_{\text{current}}, a_{\text{station}})$ is equal to its remaining contract duration a_{remain} is obliged to return to its station.

4.4. Cost Function

Applying a decision d to a vehicle k^a at time t takes the vehicle to state a' at time t' and generates a contribution c_{tad} given by

$$c_{tad} = \begin{cases} \beta^k \cdot \Big(p^c_{\text{base}} + p_{\text{time}} \cdot \Delta t_{\text{trip}} - c^k_{\text{time}} \cdot (\Delta t_{\text{pickup}} + \Delta t_{\text{trip}}) - c^c_{\text{delay}} \cdot w_{\text{delay}}\Big) & (\textit{service}), \\ -c^k_{\text{time}} \cdot \Delta t_{\text{rebalance}} & (\textit{rebalance}), \\ -c^k_{\text{time}} \cdot \Delta t_{\text{return}} & (\textit{return}) \\ c^{t,j}_{\text{stay}}, & (\textit{stay}). \end{cases}$$

Contributions c_{tad} comprise

β^k = Platform profit margin when using vehicle k,

p^c_{base} = Base fare of request $b = b_d$ of decision d from quality class $c = b_{\text{class}}$,

p_{time} = Time-dependent fare,

Δt_{trip} = Trip duration $\Delta t(b_{\text{origin}}, b_{\text{dest}})$ of request $b = b_d$,

c^k_{time} = Time − dependent operational cost of vehicle k,

Δt_{pickup} = Pickup duration $\Delta t(a_{\text{current}}, b_{\text{origin}})$ from current location to trip b origin,

c^c_{delay} = Penalty due to the excess delay w_{delay},

w_{delay} = $\max\{0, \Delta t_{\text{pickup}} - w^c_{\text{pickup}}\}$ excess delay over the pickup delay w^c_{pickup} contracted by a user from class c,

$\Delta t_{\text{rebalance}}$ = Rebalance travel duration $\Delta t(a_{\text{current}}, r)$ to target location $r \in N$,

Δt_{return} = Return travel duration $\Delta t(a_{\text{current}}, a_{\text{station}})$ of vehicle with $a_{\text{type}} = FAV$,

Published in Transportation Science on October 13, 2021 as DOI: 10.1287/trsc.2021.1069.
This article has not been copyedited or formatted. The final version may differ from this version.

$$c^{t,j}_{\text{stay}} = \text{Cost of staying at location } j \text{ at time } t,$$
$$\text{such that } j = a_{\text{current}}, \text{and } j \in N.$$

The profit margin β^k determines the percentage owed to the platform by assigning trips to a vehicle k^a of type $a_{\text{type}} \in \{\text{PAV, FAV}\}$. It allows us to adequately adjust, from the perspective of the platform, the incentive FAVs have to serve at their available times. Similarly to today's MoD applications, we assume most profits belong to the independent contractors, namely, FAV owners. In turn, constant c^k_{time} represents typical operational costs (e.g., tolls, fuel, wear and tear) for a vehicle k. We consider these costs to be equal for all vehicles, regardless of the type. The basis of a pay-per-use parking system is captured by the cost $c^{t,j}_{\text{stay}}$ of staying at the current location at time t, allowing city managers to create incentives for vehicles to avoid parking in congested areas. A user who requests a ride in class c expects to be picked up within w^c_{pickup} time units, but can tolerate up to $w^c_{\text{tolerance}}$ time units over w^c_{pickup} to be serviced, as long as he is compensated for the excess delay $w_{\text{delay}} = \max\{0, \Delta t_{\text{pickup}} - w^c_{\text{pickup}}\}$ and $w_{\text{delay}} \le w^c_{\text{tolerance}}$. From the platform perspective, this compensation represents a delay penalty c^c_{delay} incurred for $w_{\text{delay}} > 0$, defined as

$$c^c_{\text{delay}} = p^c_{\text{base}} / w^c_{\text{tolerance}}.$$

This way, if $w_{\text{delay}} = w^c_{\text{tolerance}}$, the base fare is totally offset by the penalty, and the platform will profit only from the time-dependent fare. Furthermore, we consider that when the platform fails to pick up a user from class c within the class maximum waiting time $w^c_{\text{pickup}} + w^c_{\text{tolerance}}$, the platform has to bear a rejection penalty

$$c^c_{\text{rejection}} = \rho \cdot p^c_{\text{base}},$$

where ρ is a penalty factor. Hence, when rejecting a request, the platform does not only fail to profit but may be required to compensate the inconvenienced users for a breach of contract, according to their service-level class. By setting up ρ, we can choose the extent to which rejections incur further losses, allowing us to experiment with different penalization schemes. Ultimately, for each period t, the total rejection penalty resulting from failing to service users from different classes is given by the function

$$P_t(S_t, x_t) = \sum_{c \in C} c^c_{\text{rejection}} \left(\sum_{\substack{b \in \mathcal{B} \\ b_{\text{class}} = c}} D_{tb} - \sum_{a \in \mathcal{A}} \sum_{d \in \mathcal{D}^S_c} x_{tad} \right). \quad (6)$$

We consider the service-level violation penalties c^c_{delay} and $c^c_{\text{rejection}}$ to be essential elements of SLCs, once they further back up the platform's commitment to service-level fulfillment. By combining these two penalties, we guarantee that the platform is always better off servicing a user, regardless of the service-level violation. Even when the base fare is totally offset by the delay penalty, the platform still can profit from the time-dependent fare, whereas rejecting a user always leads to losses. Finally, the contribution function representing the profit a platform can accrue at each period t is given by

$$C_t(S_t, x_t) = \sum_{a \in \mathcal{A}} \sum_{d \in \mathcal{D}} c_{tad} x_{tad} \; - \; P_t(S_t, x_t). \quad (7)$$

4.5. Objective

Let $X^\pi_t(S_t)$ be a decision function representing a policy $\pi \in \Pi$ that maps a state $S_t \in \mathcal{S}$ to a decision $x_t \in \mathcal{X}_t$, where $\mathcal{S}$ is the state space, $\mathcal{X}_t$ is the set of feasible decisions in state S_t, and Π is the set of potential decision functions. Starting from an initial state S_0, we aim to determine the optimal policy π^* that maximizes the expected cumulative contribution, discounted by a factor γ, over the planning horizon T:

$$F^*_0(S_0) = \max_{\pi \in \Pi} \mathbb{E}\left\{ \sum_{t=0}^{T} \gamma C_t(S_t, X^\pi_t(S_t)) | S_0 \right\}. \quad (8)$$

5. Algorithmic Strategies

In principle, we can solve Equation (8) by means of classical dynamic programming, recursively computing (backward through time) Bellman's optimality equations

$$V_t(S_t) = \max_{x_t \in \mathcal{X}_t} \left(C_t(S_t, x_t) + \gamma \mathbb{E}\{ V_{t+1}(S_{t+1}) | S_t, x_t \} \right), \quad (9)$$

where $S_{t+1} = S^M(S_t, x_t, W_{t+1})$. The transition function $S^{M,W}$ describes how the predecision state S_t evolves to the subsequent predecision state S_{t+1}, upon applying decisions x_t and receiving random information W_{t+1}. For each period, using the expected contributions V_{t+1} allows us to account for the downstream effect of decision making.

Solving (9), however, is computationally intractable for our problem setting. Doing so would incur all the three "curses of dimensionality" (Powell 2011). First, we are unable to enumerate all states S_t in state space $\mathcal{S}$, to evaluate value functions $V_t(S_t)$. Second, we are unable to find the optimal decision from the decision space $\mathcal{X}_t$ for all states in $\mathcal{S}$. Third, we are unable to determine the outcome space, whose dimensionality depends on the random information W_{t+1}, which for our problem comprises the uncertainty associated with the appearance of requests and FAVs.

5.1. An Approximate Dynamic Programming Algorithm

To estimate the value functions around each state in Equation (9), we develop an approximate value iteration algorithm (see, e.g., Powell, Simão, and Bouzaiene-Ayari 2012). This ADP algorithm relies on

Published in Transportation Science on October 13, 2021 as DOI: 10.1287/trsc.2021.1069.
This article has not been copyedited or formatted. The final version may differ from this version.

the concept of postdecision state, which is a deterministic state immediately after implementing a decision and before any new information has arrived. Thus, applying the decision vector x_t to state S_t leads to a deterministic postdecision state

$$S_t^x = S^{M,x}(S_t, x_t),$$

where $S^{M,x}$ is a transition function describing how the system evolves from S_t to S_t^x using decisions x_t. Then, from the postdecision state S_t^x, we can compute the subsequent predecision state

$$S_{t+1} = S^{M,W}(S_t^x, W_t(\omega))$$

using $S^{M,W}$, a transition function that models the arrival of new information $W_t(\omega)$. Through these functions, using a given policy π over the planning horizon T would produce a sequence $(S_0, x_0, S_0^x, W_1(\omega), S_1, x_1, S_1^x, W_2(\omega), S_2, \ldots, S_{T-1}, S_{T-1}^x, W_T(\omega), S_T)$. By breaking Equation (9) into two steps we have

$$V_t(S_t) = \max_{x_t \in \mathcal{X}_t} \left(C(S_t, x_t) + \gamma V_t(S_t^x)\right),$$
$$V_t(S_t^x) = \mathbb{E}\left\{V_{t+1}(S_{t+1}) | S_t^x\right\}.$$

Because we cannot compute $V_t(S_t^x)$ exactly, we aim to find $\overline{V}_t(S_t^x)$, that is, a value function approximation around the deterministic postdecision state S_t^x. Once we already penalize rejections, we assume the unmet requests from the postdecision demand vector D_t^x are not carried over to future periods (i.e., we set $D_t^x = \emptyset$). In practice, this assumption implies that users will either walk away or reenter the system in the next period through a new request upon being rejected. Hence, the postdecision state is equivalent to the postdecision resource vector, that is, $\overline{V}_t(S_t^x) = \overline{V}_t(R_t^x)$. Following the ADP algorithm, we estimate these approximations iteratively, such that at each iteration $n = 1, 2, \ldots, I$, a different sample path ω^n is considered, and we can take decisions using the value functions learned up to iteration $n-1$. Accordingly, to indicate the iterative nature of the algorithm, a superscript n is added to all variables.

Assuming $\overline{V}_t^n(R_t^{x,n})$ is linear in R_{ta}^n, we have

$$\overline{V}_t^n(R_t^{x,n}) = \sum_{a' \in \mathcal{A}} \overline{v}_{t'a'}^n \sum_{a \in \mathcal{A}} \sum_{d \in \mathcal{D}} \delta_{a'}(a,d) x_{tad}, \tag{10}$$

where

$\overline{v}_{t'a'}^n$ = marginal value of a vehicle with postdecision attribute vector a' at arrival time t' at iteration n,

$\delta_{a'}(a,d)$ = transition function equals one if $a^M(a,d) = a'$ and zero otherwise.

In our problem, the marginal values $\overline{v}_{ta}^n$ have slightly different interpretations, depending on vehicle type. For PAVs, these values approximate the overall contribution (i.e., until the end of the simulation horizon T) of assigning an incremental vehicle to a certain location at a certain time. For FAVs, however, a marginal value also reflects a vehicle's remaining contract duration and the station location. For example, FAVs with higher remaining service durations, operating in locations close to their stations, are expected to draw higher contributions. Conversely, FAVs far from their stations and with contracts about to expire cannot render contributions as high because the last moments of their contracts are reserved for return trips to their stations.

Although we assume $\overline{V}_t^n(S_t^{x,n})$ is linear in R_t^n, we acknowledge this assumption is prone to result in an oversupply of vehicles in regions associated with high marginal values. Intuitively, the more vehicles rebalance to a certain region, the lower becomes their average contribution because only a few of them actually will service users. Instead of dampening these values as the number of vehicles increases (by using piecewise-linear approximations as in Topaloglu and Powell 2006), we limit the number of vehicles arriving at each network location. Our fine-grained spatiotemporal representation (featuring short periods and exact street coordinates) allow us to restrict the number of vehicles entering each location in Constraints (3). Besides avoiding vehicles flooding certain areas, these constraints add a degree of realism to the model because they are based on the physical capacity of the actual infrastructure as well as city's traffic rules.

We do not restrict the number of vehicles dropping passengers at the same location because they are already bounded by the number of demands. Because of the characteristics of our problem setting, especially the adoption of short periods and the spatiotemporal distribution of the demand, it is unlikely that a high number of users are arriving at the same location at the same time.

Finally, the problem of finding the optimal decision function is

$$X_t^\pi(S_t^n) = \underset{x_t \in \mathcal{X}_t^n}{\operatorname{argmax}} \left(\sum_{a \in \mathcal{A}} \sum_{d \in \mathcal{D}} c_{tad} x_{tad} + \gamma \sum_{a' \in \mathcal{A}} \overline{v}_{t'a'}^{n-1} \sum_{a \in \mathcal{A}} \sum_{d \in \mathcal{D}} \delta_{a'}(a,d) x_{tad} \right) \tag{11}$$

$$= \underset{x_t \in \mathcal{X}_t^n}{\operatorname{argmax}} \sum_{a \in \mathcal{A}} \sum_{d \in \mathcal{D}} \left(c_{tad} + \gamma \sum_{a' \in \mathcal{A}} \overline{v}_{t'a'}^{n-1} \delta_{a'}(a,d) \right) x_{tad} \tag{12}$$

$$= \underset{x_t \in \mathcal{X}_t^n}{\operatorname{argmax}} \sum_{a \in \mathcal{A}} \sum_{d \in \mathcal{D}} \left(c_{tad} + \gamma \overline{v}_{t'a^M(a,d)}^{n-1} \right) x_{tad}. \tag{13}$$

5.2. A Discount Mechanism for Multiperiod Travel Times

Mobility-on-demand users typically require quick response times from transportation platforms. For this reason, most studies on urban MoD applications either process requests as soon as they are placed or in batches, usually considering short time intervals. Following such practice in our ADP approach, however,

Published in Transportation Science on October 13, 2021 as DOI: 10.1287/trsc.2021.1069.
This article has not been copyedited or formatted. The final version may differ from this version.

prevents us from assuming that all decisions acting on the resources will be completed in the subsequent period. In fact, most pickup and rebalancing decisions can last longer than a single period. Although we work with a high-resolution street network, our locations still correspond to a restricted subset of all possible coordinates. The lower the resolution of the underlying map, the fewer the locations available, and the more multiperiod travel times can be expected between location pairs. Therefore, incorporating such a feature helps to create a more robust solution, independent of the length of the periods or the underlying map structures.

Such multiperiod resource-transformation times have a significant influence on the value function approximations. To avoid adding another attribute to our resource attribute vector to account for the arrival time at the destination location (see Topaloglu and Powell 2006), we implement an online discount mechanism to all value function approximations associated with postdecision states arising from rebalancing decisions. We dampen the value function of postdecision states $a' = a^M(a,d)$ by discounting the opportunity cost of staying still (i.e., $d = d^{\text{stay}}$) during the rebalancing periods $t'' \in \{t+1,\ t+2,\ \ldots,\ t'-1\}$ using

$$\overline{v}^n_{tt'a'} = \overline{v}^n_{t'a'} - \sum_{t''=t+1}^{t'-1} \overline{v}^n_{t''a^M(a,d^{\text{stay}})} \ \forall\ d \in \mathcal{D}^R. \tag{14}$$

In Equation (14), if the resource-transformation time takes a single period (i.e., $t' = t+1$), we have $\overline{v}^n_{tt'a'} = \overline{v}^n_{t'a'}$. Because we do not allow vehicles to interrupt a rebalance trip, this adaptation is crucial to avoid vehicles being too far-sighted, pursuing future rewards at long-distance locations while ignoring the requests that might occur (in the next periods) in the surroundings of their starting location after their departure. On the other hand, this adaptation also avoids vehicles being stranded in low-demand areas, allowing them to move directly to farther high-demand areas instead of endlessly rebalancing to nearby low-demand neighbors. Therefore, at every decision time, an idle vehicle can also rebalance to farther, high-value function locations, as long as this decision is (i) at least as good as staying still for the whole rebalancing time and (ii) competitive in relation to rebalancing to its closest neighbors.

5.3. Value Function Updates

At iteration n, we consider a sample path ω^n that determines $\hat{R}^n_t = \hat{R}_t(\omega^n)$ and $\hat{D}^n_t = \hat{D}_t(\omega^n)$, such that $W_t(\omega^n) = (\hat{R}^n_t, \hat{D}^n_t)$. Let $\overline{V}^{n-1}_t(S^{x,n}_t)$ be an approximation of the value of being in the postdecision state $S^{x,n}_t = S^{M,x}(S^n_t, x_t)$ considering the first $n-1$ iterations. Given the state $S^n_t = S^{M,W}(S^{x,n}_{t-1}, W_t(\omega^n))$, we can make decisions at time t by solving the optimization problem

$$F(S^n_t) = \underset{x_t \in \mathcal{X}^n_t}{\operatorname{argmax}} \left(C_t(S^n_t, x_t) + \gamma \overline{V}^{n-1}_t(S^{x,n}_t) \right), \tag{15}$$

where we seek to determine the decision vector x_t in the feasible region $\mathcal{X}^n_t$ that maximizes the sum of the current contribution and the expected contribution (discounted by a γ factor).

In Algorithm 1, we present how our optimization problem is inserted into a classic ADP algorithm. First, all value function approximations are set to zero by default. Then, we start from an initial state $S^1_0 = (R^1_0, D^1_0)$, where R^1_0 comprises the state vectors of PAVs randomly distributed throughout the map, and D^1_0 is empty (i.e., no requests have arrived). We update value functions $\overline{v}^n_{ta}$ using the samples $\hat{v}^n_{ta}$ drawn from attribute vector a at time t and iteration n. New samples are smoothed using step sizes α_n, which are updated every iteration according to the McClain's rule (see George and Powell 2006), such that

$$\alpha_n = \frac{\alpha_{n-1}}{1 + \alpha_{n-1} - \overline{\alpha}},$$

where $\overline{\alpha}$ is a constant that is approached as n advances. Initially, we set $\alpha_1 = 1$ such that value functions can start with the first sample value measured for each state.

Algorithm 1 (An Approximate Dynamic Programming Algorithm to Solve the AMoD-H Assignment Problem)

1: Choose an initial approximation $\overline{V}^0_t,\ \forall\ t \in \mathcal{T} = \{0, 1, \ldots, T\}$.
2: Set the initial state to S^1_0.
3: **for** $n = 1, \ldots, I$ **do**
4: Choose a sample path ω^n.
5: **for** $t = 0, 1, \ldots, T$ **do**
6: Let x^n_t be the solution of the optimization problem (15).
7: Let $\hat{v}^n_{ta}$ be the dual variable corresponding to the resource conservation constraint (1) for each $R^n_{ta} > 0$.
8: Update the value function using

$$\overline{v}^n_{ta} = (1 - \alpha_n)\overline{v}^{n-1}_{ta} + \alpha_n \hat{v}^n_{ta}.$$

9: Compute the subsequent predecision state:

$$S^{x,n}_t = S^{M,x}(S^n_t, x^n_t),$$
$$S^{t+1n}_{\ \ S} = S^{M,W}(S^{x,n}_t, W_{t+1}(\omega^n)).$$

10: Update the total number of vehicles K_j inbound to each location $j \in N$.
11: **end for**
12: **end for**
13: Return value functions: $\{\overline{v}^n_{ta} |\ \forall\ t \in \mathcal{T},\ \forall\ a \in \mathcal{A}\}$.

5.4. Approximating the Value Function

Because we are unable to enumerate all the attributes in the state space $\mathcal{A}$, we use hierarchical aggregation (Section 5.4.1) to create a sequence of state spaces.

Aggregating on the space dimension helps to estimate the value function of states featuring locations that were not yet visited, by using the estimates of regions in hierarchically superior levels. We define regions by clustering nodes in our street network that can be accessed from central locations within increasingly higher maximal delays. Besides aggregating across space, near periods can aggregate up to larger time intervals because the value function of a vehicle at a location (or region) is likely to carry some resemblance to the value functions of anterior/posterior periods. Such resemblance, therefore, allow us to approximate value functions across periods that belong to longer time intervals. Later, in Section 6.2.2, we present the final spatiotemporal hierarchical aggregation structure, achieved experimentally by assessing the performance of different aggregation structures on a single baseline scenario.

5.4.1. Hierarchical Aggregation. In order to estimate the value function of attributes not yet observed, we use hierarchical aggregation coupled with the *weighting by inverse mean squared errors* (WIMSE) formula proposed by George, Powell, and Kulkarni (2008). Our hierarchical aggregation structure lays out a sequence of state spaces $\{(\mathcal{T}\times\mathcal{A})^{(g)}, g=1,2,\ldots,|\mathcal{G}|\}$ with successively fewer elements, where $(\mathcal{T}\times\mathcal{A})^{(g)}$ represents the gth level of aggregation of the time-attribute space $\mathcal{T}\times\mathcal{A}$. Hence, each attribute $ta\in\mathcal{T}\times\mathcal{A}$ can be aggregated up to an attribute $ta^{(g)}=G^g(ta)$, where $G^g:\mathcal{T}\times\mathcal{A}\rightarrow(\mathcal{T}\times\mathcal{A})^{(g)}$. Doing so allows us to estimate the value $\overline{v}^n_{ta}$ associated with an attribute ta by combining the values $\overline{v}^{(g,n)}_{ta}$ from superior levels using

$$\overline{v}^n_{ta}=\sum_{g\in\mathcal{G}} w^{(g,n)}_{ta}\cdot\overline{v}^{(g,n)}_{ta}.$$

Weights $w^{(g)}_{ta}$ on the estimates of different aggregation levels are inversely proportional to the estimates of their mean squared deviations, according to the WIMSE formula:

$$w^{(g,n)}_{ta}\propto\frac{1}{\left(\overline{\sigma}^2_{ta}\right)^{(g,n)}+\left(\overline{\mu}^{(g,n)}_{ta}\right)^2},$$

where $\left(\overline{\sigma}^2_{ta}\right)^{(g,n)}$ is the variance of the estimate $\overline{v}^{(g,n)}_{ta}$, and $\left(\overline{\mu}^{(g,n)}_{ta}\right)^2$ is the aggregation bias, that is, the difference between the estimate $\overline{v}^{(g,n)}_{ta}$ at aggregate level g and the estimate $\overline{v}^{(0,n)}_{ta}$ at the disaggregate level. Next, we normalize all weights by doing

$$w^{(g,n)}_{ta}=\frac{1}{\left(\overline{\sigma}^2_{ta}\right)^{(g,n)}+\left(\overline{\mu}^{(g,n)}_{ta}\right)^2}\left[\sum_{g'\in\mathcal{G}}\frac{1}{\left(\overline{\sigma}^2_{ta}\right)^{(g',n)}+\left(\overline{\mu}^{(g',n)}_{ta}\right)^2}\right]^{-1}. \quad (16)$$

5.4.2. The Street Network Map. AMoD studies within the scope of reinforcement learning and ADP generally assume cars can rebalance to their immediate neighboring zones. Moreover, such rebalancing operations are expected to last at most a single period, such that, at decision time, all vehicles are either servicing customers or idle (potentially, after finishing rebalancing). Such zones, however, are defined using artificial grids (e.g., Wen, Zhao, and Jaillet 2017; Al-Kanj, Nascimento, and Powell 2020) or neighborhood borders (e.g., Gueriau and Dusparic 2018, Lin et al. 2018), which do not necessarily translate into realistic drivable streets. In contrast, we work with a high-resolution transportation network of Manhattan comprising 6,430 nodes and 11,581 edges. Therefore, pickup and rebalance decisions consist of movements between real-world street coordinates, discretized in a set of network nodes, which we guarantee to be no longer than 30 seconds away from one another (at an average speed of 20 km/h). Such a high-granularity setup allows us to consider a more realistic demand matching scenario because real-world trip requests have a larger set of candidate nodes to which their GPS coordinates can be approximated.

5.4.3. Hierarchical Regional Centers. In order to define hierarchical regions in our street network map, we implement a variant of the facility location problem proposed by Toregas et al. (1971), which is concerned with the time that separates a location from its closest facility. The goal of this problem is to determine the minimum set of facilities in the street network graph $G=(N,E)$ that together can cover (reach) all others within s time units. Let

x_j = 1 if a facility is located at $j\in N$, 0 otherwise,
t_{ij} = Travel time between nodes $i,j\in N$,
s = The maximal service delay of a vehicle departing from a facility,
$N_{p,s}$ = Subset of locations able to reach location $p\in N$ within s time units (i.e., $N_{p,s}=\{j\,|\,t_{jp}\leq s,\ \forall\, j\in N\}$).

The minimum set covering problem is defined as follows:

$$\text{Minimize}: \sum_{j\in N} x_j \quad (17)$$

$$\text{Subject to}: \sum_{j\in N_{p,s}} x_j\geq 1 \qquad \forall p\in N, \quad (18)$$

$$x_i\in\{0,1\} \qquad \forall i\in N. \quad (19)$$

Published in Transportation Science on October 13, 2021 as DOI: 10.1287/trsc.2021.1069.
This article has not been copyedited or formatted. The final version may differ from this version.

An optimal solution to this set covering problem would give us the location of the minimum set of facilities $J_s \subseteq N$ that would be required to service all locations p and still ensure a maximal service time of s units for the entire system. We assume these facilities are regional centers $j \in J_s$ and consider that each node $p \in N$ integrates the region of its closest center $j = \arg\min_{i \in J_s} t_{ip}$.

5.5. Benchmark Policy

We benchmark our method against a myopic policy π_{myopic} comprising two phases. In the first phase, we determine the optimal vehicle-request assignment at period t by maximizing the contribution function given by Equation (7). Next, in the second phase, idle vehicles are optimally rebalanced to undersupplied locations using the linear program proposed by Alonso-Mora et al. (2017). This program aims to minimize the total travel distance of reaching the pickup locations of unassigned requests while guaranteeing that either all vehicles or all requests are assigned. The original formulation is adapted such that it abides by the contractual deadlines of freelance vehicles. We preemptively discard FAVs that, although idle, cannot reach any rebalancing targets within their remaining service times.

To increase the matching rate, we assume rebalancing decisions can be revoked at every decision time. Hence, in the first phase of our policy, both rebalancing and idle vehicles can be assigned to new requests. Thanks to our high-resolution network representation, we can calculate the current locations of all rebalancing vehicles at each period t. Therefore, a rebalancing vehicle can be matched to any request occurring in the surroundings of its ongoing route (i.e., the shortest path to its destination).

6. Experimental Study

We implemented our approach using Python 3.6 and Gurobi 8.1. Test cases were executed on a 2.60 GHz Intel Core i7 with 32 GB of random access memory.

6.1. Training and Testing Data Sets

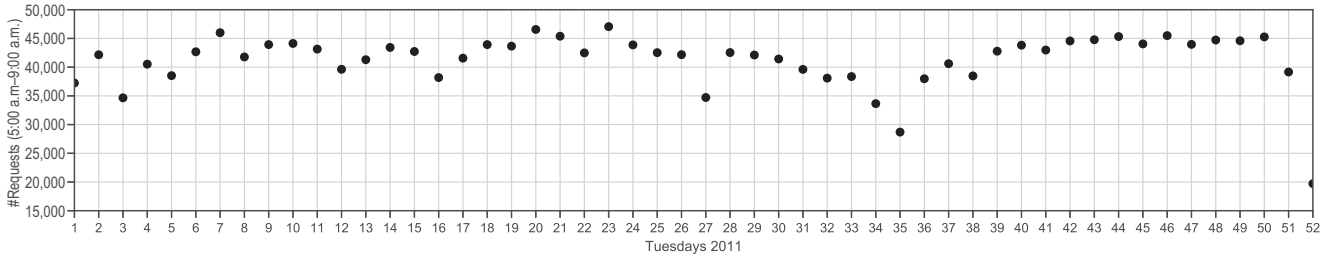
We create our data set by randomly sampling 10% of the 2011 Manhattan, New York City, taxi demand. Value functions are created using requests sampled from the 15th Tuesday, and their effectiveness is assessed on testing instances created using samples from the remaining 51 Tuesdays of 2011. This setup allows that we investigate how the policy learned from a single weekday performs throughout the whole year. To assess the quality of our VFA policy, we compare the average profit and service level across the 51 samples against the averages provided by the benchmark policy. Figure 2 shows the total request count for each day. For the sake of fairness, the random processes associated with trip sampling, service class assignment, and fleet distribution are a function of the iteration number (i.e., the seed). This way, regardless of the configuration, we guarantee that the same information will be considered across all training iterations and testing instances.

Figure 3 offers a close-up on the transportation demand of the 15th Tuesday, highlighting the morning peak from which we draw samples. The dashed lines at 4:30 a.m. and 10:00 a.m. delineate the full extent of our experiment. During the interval [4:30 a.m., 5:00 a.m.), the fleet has a 30-minute offset (30 periods) to rebalance in order to serve the future demand. Request batches arrive every other minute in the interval [5:00 a.m., 9:00 a.m.) (240 periods), and vehicles have a termination offset [9:00 a.m., 10:00 a.m.) (60 periods) to make sure all requests picked up around the end of the trip sampling threshold can be delivered. The rebalance offset and the lack of requests at the end of the trip sampling interval allow us to better assess the performance of our anticipatory rebalancing method. Regarding the computation time, training and testing algorithms take on average five and two minutes, respectively, to process a single iteration comprising 330 steps.

6.2. Model Tuning

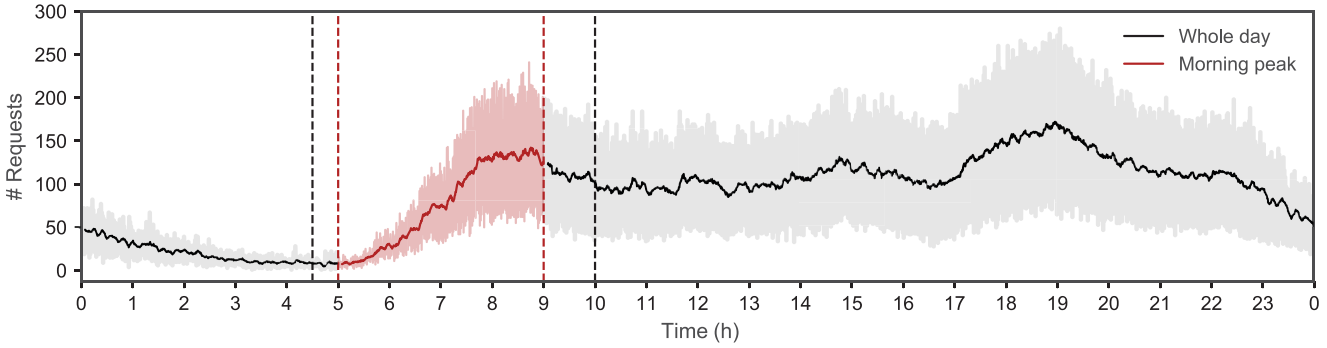
In this section, we motivate our algorithmic choices by showing their effectiveness experimentally. First,

Figure 2. Request Count Between 5:00 a.m. and 9:00 a.m. Throughout All 52 Tuesdays of 2011 of the Manhattan Taxi Demand Data Set



Note. VFAs are determined using only samples from the 15th Tuesday.

Figure 3. (Color online) Demand Pattern of Manhattan Taxi Trips on a Typical Tuesday, 2011



Notes. At every ADP iteration, our simulation draws samples from the morning peak (the interval from 5:00 a.m. to 9:00 a.m.). The dashed lines at 4:30 a.m. and 10:00 a.m. mark the full length of the experiment. The interval [4:30 a.m., 5:00 a.m.) is a rebalancing offset, whereas the interval [9:00 a.m., 10:00 a.m.) is a termination offset. The former is laid out to provide extra time for vehicles to rebalance before any requests arrive, and the latter allows enough time to deliver all requests picked up during the sampling interval.

we describe the baseline scenario we use throughout the tuning process. Next, we present the spatiotemporal hierarchical aggregation structure we use to approximate value functions. Then, we highlight the effectiveness of our discount function when dealing with multiperiod travel times, show how we set up our rebalancing strategy, and describe the importance of setting a limit on the number of vehicles allowed in each node of the street network. Finally, we offer a sensitivity analysis on the maximum pickup times and base fares values.

Regarding the tuning of the ADP parameters, we have found that letting the step size $\overline{\alpha} = 0.1$ and the discount factor $\gamma = 1$ has led to superior performance experimentally for $I = 500$ iterations. Hence, we adopted these values across all considered scenarios.

6.2.1. Baseline Scenario. Before we study the impact of FAV hiring and service classes, we tune our model using a baseline scenario that emulates a traditional MoD application with a fixed fleet size, homogeneous users (i.e., no service quality classes), and no service-level penalties. This scenario features a fleet of 300 PAVs, which are randomly distributed throughout the street network at the beginning of each iteration. Every minute, we sample the correspondent request batch such that 10% of the requests are selected, totaling about 4,300 requests over all periods. We set up the PAV fleet size experimentally, aiming to service the sampled demand partially. We assume such an undersupplied scenario to guarantee that there are always unmet requests left to be addressed, eventually, by the freelance fleet. Additionally, following Constraints (3), we assume there can be only five vehicles inbound to each location. Table 2 summarizes the baseline scenario parameters.

6.2.2. Hierarchical Aggregation Levels. We determine our aggregation levels experimentally by analyzing the quality of the solutions provided by different schemes that combine both space and time. Figure 4 illustrates the underlying structure of our spatial aggregation configuration, showing to which regional center each location in N aggregates up. In Table 3, we show the decline in the attribute space size for each aggregation level. At the most disaggregated level (i.e., $g = 0$), we consider that both FAV and PAV value functions are indexed by time and location. FAVs, in particular, are also indexed by their remaining contract durations and station locations. Because considering fine-grained values for the FAV-only attributes could lead to an excessively large attribute space, we replace them with coarser substitutes. First, for the remaining contract durations, we assume values are discretized in hours. Assuming FAVs arrive in the system during the 240 1-minute trip sampling intervals (see Table 2), the longest contract can last four hours. Therefore, contract durations in intervals 1–60 periods, 61–120 periods, 121–180 periods, and 180–240 periods aggregate up to one, two, three, and four remaining hours, respectively. Second, we assume the station locations aggregate up to one of the 21 10-minute regional centers. At aggregation level 1, we aggregate time up to 3-minute intervals and locations to the closest 5-minute regional center. Additionally, we stop considering FAV-related attributes, therefore using only the spatiotemporal information to index them. Finally, at aggregation level 2, we continue to aggregate time in 3-minute intervals and aggregate locations up to the coarser 10-minute regional centers. At this level, we are left out with 2,310 possible attributes for each fleet type, substantially improving our ability to estimate values $\hat{v}_{ta}$ of states not yet visited.

We separate states by car type to emphasize that PAVs and FAVs are not interchangeable: FAVs are

Published in Transportation Science on October 13, 2021 as DOI: 10.1287/trsc.2021.1069.
This article has not been copyedited or formatted. The final version may differ from this version.

Table 2. Parameters for the Baseline Scenario, Featuring a Fixed PAV Fleet, Homogeneous Users, and No Service-Level Penalties

Problem characteristic	Attribute value(s)
Fleet size ($\|K\|$)	300 PAVs
Max. #vehicles/location ($k_j^{\max}$)	5 (for all locations $j \in N$)
Base fare (p_{base})	\$2.4
Distance fare (p_{time})	\$1.0/km
Driving costs (c_{time})	\$0.1/km
Pickup delay (w_{pickup})	10 min
Number of locations ($\|N\|$)	6,430
Period length	1 min
Simulation length (T)	330 periods (morning peak): 30: rebalancing offset (30 min) 240: trip sampling (5:00 a.m. to 9:00 a.m.) 60: finalize delivery offset (1 h)
Demand stochastic process ($\mathcal{F}^{\text{P}}$)	10% of the real-world Manhattan taxi demand on the 15th Tuesday of 2011 (randomly sampled)

expected to work harmonically with PAVs as a back-up fleet. The marginal value of an FAV at a certain time and location differs from that of its PAV counterpart, not only because it depends on the contract duration and station location attributes, but mainly because FAV operations entail a lower profit margin to the platform, which consequently leads to lower value functions.

6.2.3. Effectiveness of the VFA Discount Function. We assess whether our discount function is able to produce high-quality value function approximations by disabling it and allowing vehicles to rebalance to increasingly farther distances. We assume vehicles can rebalance to eight regional centers determined using 1-, 5-, and 10-minute maximal service delays. Accordingly, we label these experiments as 8RC1, 8RC5, and 8RC10, and add an extra label [P] to indicate the cases where we apply the discount function. Therefore, in all test cases featuring the label [P], rebalancing leads to penalties proportional to the trip duration. We benchmark these results against a simple

Figure 4. (Color online) Regional Center Distribution on the Manhattan Street Network Graph

N (6,430)
Manhattan street
network node set

RC5 (50)
5-minute
regional centers

RC10 (21)
10-minute
regional centers

Notes. Labels RC5 and RC10 identify the regional centers determined using the facility location problem formulation considering maximal service delays of 5 and 10 minutes, respectively. Each location in N is connected to its respective regional center (white circle) by a line.

Table 3. Hierarchical Aggregation Levels

g	# periods	# location	# contracts[a]	# stations[a]	$\lvert\mathcal{T}\rvert \times \lvert\mathcal{A}^{FAV}\rvert$	$\lvert\mathcal{T}\rvert \times \lvert\mathcal{A}^{PAV}\rvert$
0	330 (t = 1 min)	6,430 (N)	4 (4 h/60 min)	21 (RC10)	178,239,600	2,121,900
1	110 (t = 3 min)	50 (RC5)	—	—	5,500	5,500
2	110 (t = 3 min)	21 (RC10)	—	—	2,310	2,310

Note. A dash indicates that the attribute is not considered.
[a]Considered only for FAVs.

rebalancing procedure where vehicles can move only to their adjacent neighbors. Because traveling to these neighbors in the street network graph is guaranteed to take less than 30 seconds, no multiperiod travel times are incurred. Figure 5 shows that for all three rebalancing strategies considered, applying the discount function leads to superior results, with the 8RC1[P] rebalancing configuration having the highest profits and percentage of serviced users by the 500th iteration.

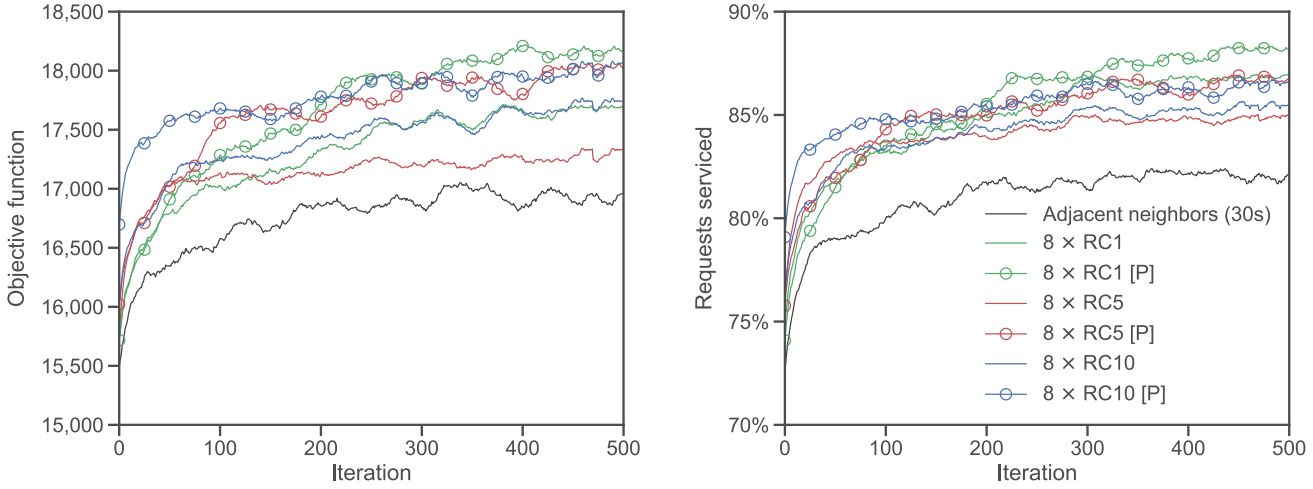
6.2.4. Rebalancing Configurations. We study rebalancing configurations in which vehicles can move to a subset of increasingly distant regional centers besides their adjacent neighboring locations. Notably, in Figure 5, the 8RC10[P] configuration allows high-quality results at the beginning of the simulation (first 200 iterations) but is later surpassed by the 8RC1[P] configuration. The performance of the long-distance configuration (8RC10[P]) is inferior to that of its counterparts because rebalancing to farther 10-minute regional centers prevents vehicles from consistently measuring a greater range of states, although it allows them to escape from low-demand areas faster initially. Therefore, rebalancing to 1-minute regional centers offers a more balanced trade-off between exploration and exploitation, because vehicles can visit more locations (number of RC1 locations = 758 ≈ 12% of node set N) and bypass the intricate complexity of the real-world street networks.

In order to assess whether we could benefit from combining the short-distance 8RC1 configuration with medium- and long-distance rebalancing movements, we created the following configurations:

- *Short* + *medium* (8RC1_4RC5): Rebalance to eight 1-minute regional centers and four 5-minute regional centers.
- *Short* + *long* (8RC1_4RC10): Rebalance to eight 1-minute regional centers and four 10-minute regional centers.
- *Short* + *medium* + *long* (8RC1_4RC5_2RC10): Rebalance to eight 1-minute regional centers, four 5-minute regional centers, and two 10-minute regional centers.

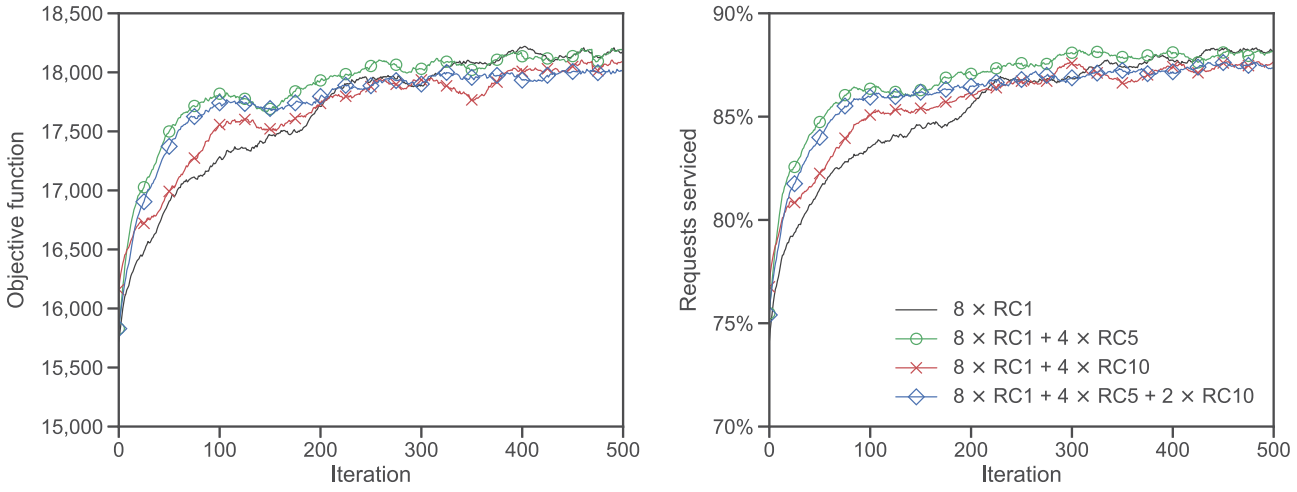
Figure 6 shows that configuration 8RC1_RC5 (i.e., adding four five-minute regional centers to the rebalancing pool of eight one-minute regional centers) results in higher performance than the 8RC1 configuration initially while having comparable convergence behavior after the 400th iteration. However, because the rebalancing configurations perform similarly at the end of the training iterations, we select 8RC1 as the default rebalancing configuration. We do so, mainly because this configuration requires less processing time than its counterparts, once fewer targets are considered.

Figure 5. (Color online) Performance Comparison of Rebalancing Strategies When Using the Discount Function (Represented by Label [P])



Published in Transportation Science on October 13, 2021 as DOI: 10.1287/trsc.2021.1069.
This article has not been copyedited or formatted. The final version may differ from this version.

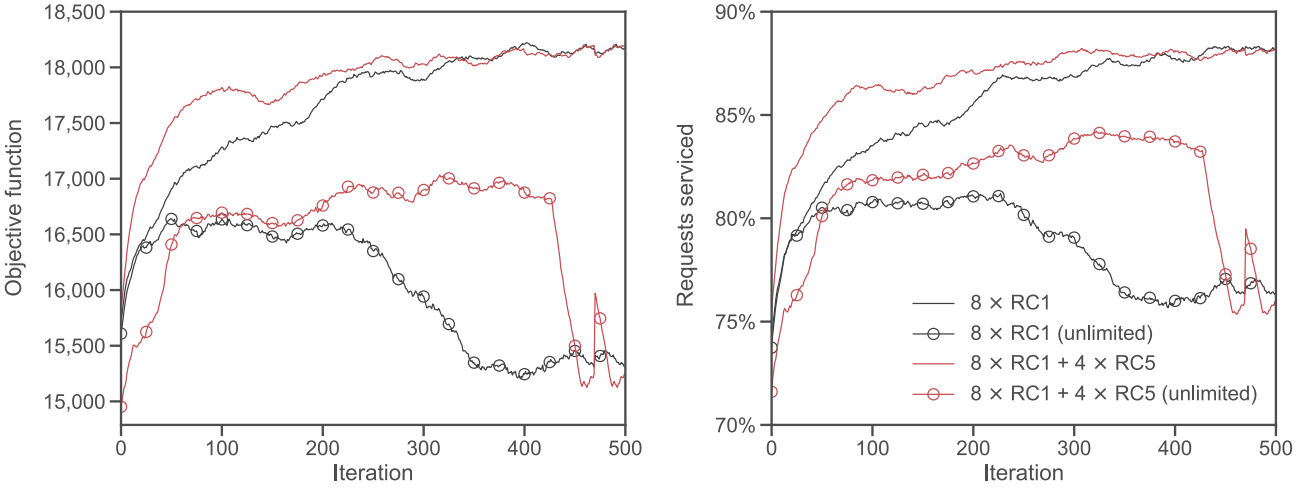
Figure 6. (Color online) Performance Comparison of Rebalancing Strategies Combining Short-, Medium-, and Long-Distance Targets



6.2.5. Setting the Maximum Number of Vehicles per Location. Our baseline scenario considers that no more than five vehicles can be inbound to any location, according to Constraints (3). To investigate how much these constraints contribute to generating high-quality value function approximations, we run the test cases 8RC1 and 8RC1$_4$RC5, allowing that an unlimited number of vehicles move to each location. Figure 7 shows that besides reaching a subpar performance initially, disabling the maximum number of vehicles/location constraints is a great source of instability as the experiment progresses. Vehicles end up rebalancing in troves to locations associated with high-value function approximations, producing a rather unrealistic scenario in our problem setting, where locations correspond to GPS coordinates in a Manhattan street segment. Figure 8 shows that allowing that up to five vehicles are inbound to each location achieves the best performance for our baseline scenario and rebalance configuration 8RC1$_4$RC5.

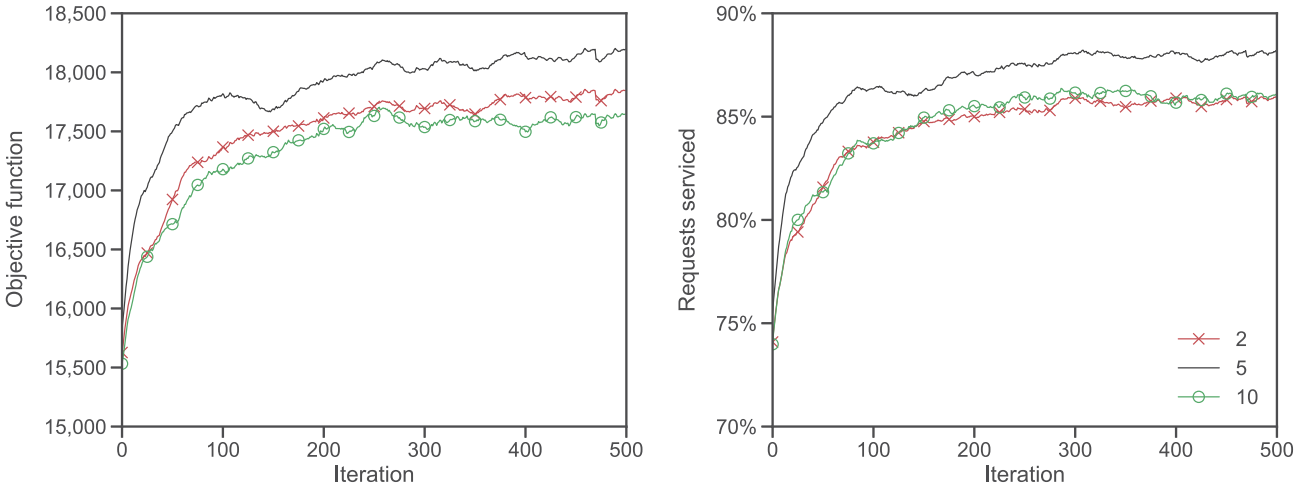
6.2.6. Base Fare Values and Service Levels. We also analyze the impact of base fare values and pickup delays on the overall performance. Whereas base fare values directly influence the scale of value functions, the maximum pickup delays limit the matching radius of vehicles. Table 4 presents the average results of our testing instances considering our baseline scenario under nine different combinations of maximum pickup delays and base fare values. Apart from the average number of requests serviced, pickup delay, and the objective function, it also shows the average trip

Figure 7. (Color online) Effect of Allowing an Unlimited Number of Vehicles at Each Node for Rebalancing Strategies 8 × RC1 and 8 × RC1 + 4 × RC5



Published in Transportation Science on October 13, 2021 as DOI: 10.1287/trsc.2021.1069.
This article has not been copyedited or formatted. The final version may differ from this version.

Figure 8. (Color online) Performance of Rebalancing Configuration 8 × RC1 + 4 × RC5 When Allowing That at Most 2, 5, and 10 Vehicles Are Inbound to Each Location



distance of both serviced and rejected users as well as the shares of the fleet total time spent parked, rebalancing, picking up, and carrying users. Because we consider 330 periods and a 300-vehicle fleet, this total fleet time corresponds to 990,000 periods (300 × 330).

Increasing base fares makes the contribution accumulated via distance rates more and more irrelevant, as indicated by the increment in the average trip distance of rejected requests. Therefore, adopting high base fares create a bias toward short-duration trips, as indicated by the decrease of both the share of the time picking up users and their average trip distances. Accordingly, vehicle rebalancing also increases once vehicles tend to return more frequently to high-demand areas. As for the influence of higher maximum pickup delays, increasing delays from 5 to 10 minutes can result in about a 10% increase in the number of requests serviced. Such an increase, however, is moderate when we contrast 10- and 15-minute delays (about two percentage points). This result suggests that, for the fleet size we have set, it is unlikely that increasing pickup delays even further will lead to more pickups. Because we consider the decision to pick up or reject a user is taken within a single period, eventually, there are not enough vehicles to fulfill the demand, regardless of how long users are willing to wait.

6.3. Platform Fleet Management

In this section, we illustrate the behavior of our π_{VFA} policy. Besides the baseline parameters described in Table 2, we consider the best tuning settings achieved in Section 6.2, namely, the three hierarchical aggregation levels presented in Table 3, the five-vehicle limit per location, and the rebalancing strategy 8RC1. Figure 9 and Figure 10 compare the performance of the proposed VFA policy against the myopic policy on a single

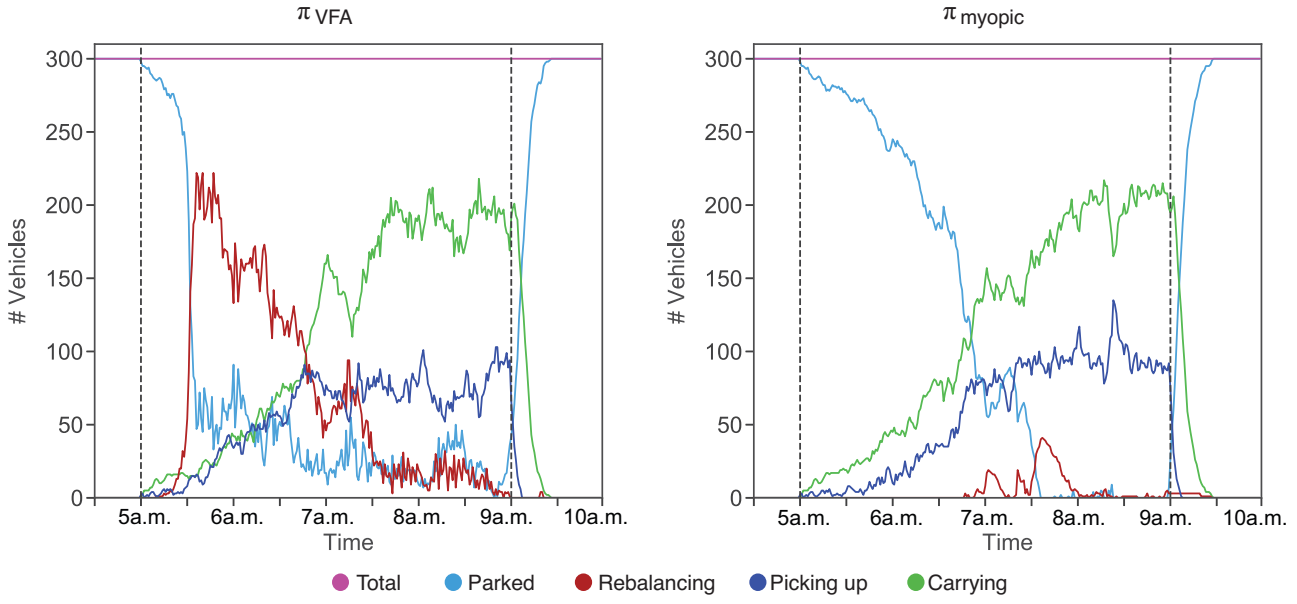
Table 4. Impact of Maximum Pickup Delays and Base Fare Values on the Solution Quality Considering the Baseline Scenario

Max. delay (min)	Base fare ($)	Requests serviced (%)	Pickup delay (min)	Objective func.($)	Trip distance (km)		Fleet total time/status			
					Serviced	Rejected	Rebalancing (%)	Picking up (%)	Carrying (%)	Parked (%)
5	2.4	76.88	2.49	15,305	2.95	3.32	10.36	7.92	28.11	53.60
	4.8	75.49	2.50	22,200	2.88	3.51	11.73	7.82	26.94	53.51
	9.6	78.19	2.49	38,190	2.81	3.83	12.91	8.07	27.29	51.73
10	2.4	85.74	3.72	16,834	2.89	3.94	8.64	13.17	30.73	47.47
	4.8	87.77	3.72	25,540	2.80	4.77	10.31	13.52	30.48	45.69
	9.6	89.34	3.86	43,531	2.77	5.34	8.83	14.29	30.69	46.19
15	2.4	87.77	4.35	17,335	2.93	3.80	7.98	15.75	31.94	44.33
	4.8	89.35	4.40	26,095	2.83	4.81	8.43	16.24	31.40	43.94
	9.6	89.71	4.47	43,594	2.77	5.46	9.88	16.57	30.79	42.76

Note. Each value corresponds to an average of the results achieved for the 51 testing instances.

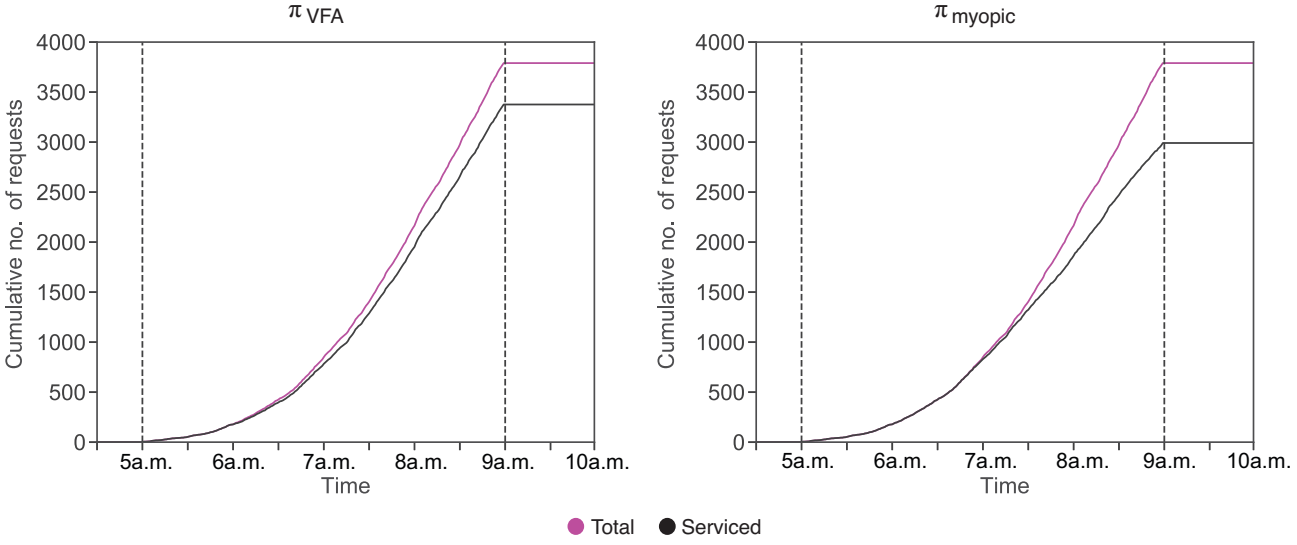
Published in Transportation Science on October 13, 2021 as DOI: 10.1287/trsc.2021.1069.
This article has not been copyedited or formatted. The final version may differ from this version.

Figure 9. (Color online) Comparison of the Number of PAVs by State (Parked, Rebalancing, Picking Up, and Carrying Passengers) for Each Policy on a Single Testing Instance



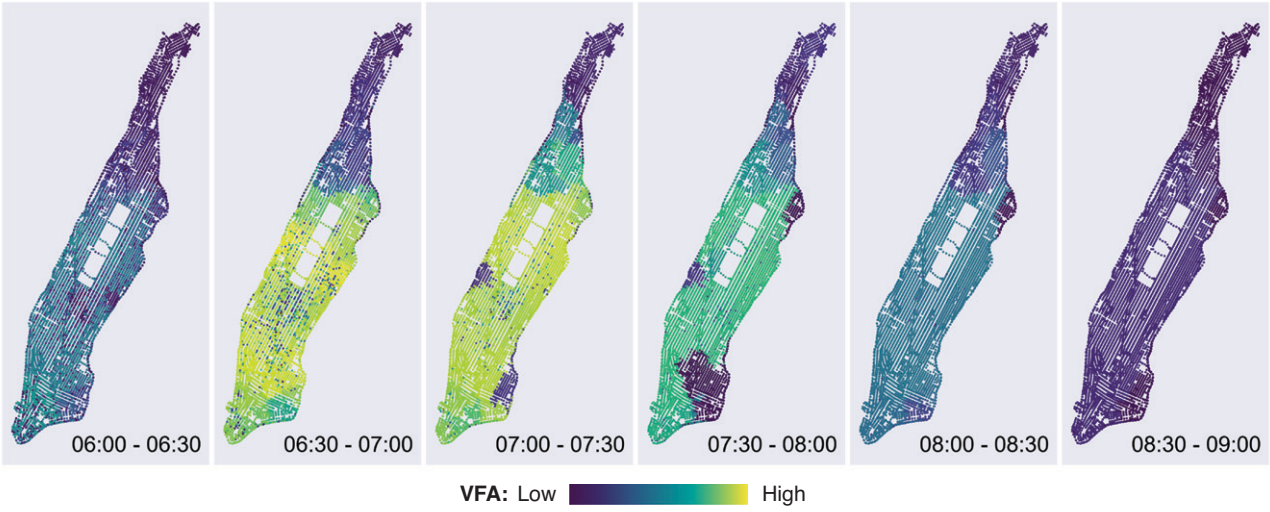
testing instance. Because the myopic policy reacts to request rejection, from Figure 9, we can see that the fleet can fulfill the demand entirely until about 6:45 a.m., when the first rebalancing movement appears. In contrast, under our VFA policy, most vehicles are rebalancing before 6:30 a.m. As can be seen in Figure 10, the π_{myopic} rejects fewer users than π_{VFA} until about 7:30 a.m., but from this time on, the π_{VFA} outperforms the myopic approach, ultimately resulting in about 14% more users serviced. The difference between the policies is further highlighted in the busiest period (from 8:00 a.m. to 9:00 a.m.). The myopic policy reacts immediately to the demand peak, picking up as many users as possible, disconsidering the future outcome of these decisions. In contrast, under the VFA policy, caring about postdecision outcomes causes many vehicles to

Figure 10. (Color online) Comparison of the Cumulative Number of Requests Serviced Throughout All Time Steps on a Single Testing Instance



Note. The VFA policy leads to an 89.18% service rate, whereas the myopic policy can reach a 78.17% service rate.

Figure 11. (Color online) Average VFAs for Each Location in the Street Network Graph Across Sequential 30-Minute Intervals



Note. Value functions are the highest in the middle section of Manhattan from 6:00 a.m. to 7:30 a.m., shortly before the demand reaches its peak.

stay parked or rebalance, which may result in some rejections initially, but leads to higher service rates in the long run.

Figure 11 further illustrates where vehicles are likely to move to, based on the dimension of the value functions exploited by π_{VFA}. For each location in N, we average the estimates across 30-minute intervals from 6:00 a.m. to 9:00 a.m. As can be seen from Figure 11, having more vehicles in the middle section of Manhattan between 6:30 a.m. and 7:30 a.m. is likely to lead to higher contributions. This period is consistent with the predominance of rebalancing operations shown in Figure 9. After 7:30 a.m., VFAs get lower and lower, although demand is the highest. As demonstrated by Al-Kanj, Nascimento, and Powell (2020) for a similar AMoD setting, value functions monotonically decrease with time, because they reflect the expected revenue vehicles can accrue until the end of the time horizon. Hence, as time goes on, vehicles have less time to pick up users and make profits.

6.4. Enforcing Service-Level Contracts

In this section, we build upon the baseline scenario such that service-level violation penalties are taken into consideration. We show how the penalization mechanisms, namely, the tolerance delays and rejection penalties, can lead to a higher service rate while compensating users who have had their service levels violated. First, regarding the service-level preferences, we assume first-class users (SQ1) can wait at most 5 minutes to be picked up, whereas second-class users (SQ2) can wait at most 10 minutes. Proportionally, we assume that the base fare of SQ1 users is twice the SQ2 base fare, such that $p_{\text{base}}^{SQ1} = 4.8$ and $p_{\text{base}}^{SQ2} = 2.4$. One should notice that the parameters defined for SQ2 users coincide with those used in the tuning.

As for the penalty parameters, we consider five-minute tolerance delays for both classes and rejection penalties $\rho \in \{0,1,2\}$. For $\rho = 0$, we have a scheme where only delay penalties are incurred, whereas for $\rho \in \{1,2\}$, rejection penalties are equivalent to one and

Table 5. Summary of the Parameters Used to Enforce Service-Level Contracts on Different User Bases

Problem characteristic	Attribute value(s)
Class (c)	SQ1, SQ2
Max. pickup delay class c (w_{pickup}^{c})	SQ1 = 5 min, SQ2 = 10 min
Waiting tolerance class c ($w_{\text{tolerance}}^{c}$)	SQ1 = 5 min, SQ2 = 5 min
Penalty factor (ρ)	{0, 1, 2}
Base fare (p_{base}^{c})	SQ1 = \$2.4, SQ2 = \$4.8
User base	Scenarios: A1: Only SQ1 users A2: Only SQ2 users A3: The 20% highest tippers are SQ1

Published in Transportation Science on October 13, 2021 as DOI: 10.1287/trsc.2021.1069.
This article has not been copyedited or formatted. The final version may differ from this version.

twofold the user base fares. Finally, we also analyze the impacts of these penalties when servicing users from SQ1 and SQ2, both separately (scenarios A1 and A2) and combined (scenario A3). In A3, the service class distribution follows a stochastic process where first-class user locations and request times coincide with the 20% most generous tippers (among tipping users) of the Manhattan demand occurring between 5:00 a.m. and 9:00 a.m. To create this distribution, we first aggregate all requests from the taxi demand considered (all 2011 Tuesdays) according to their location and placement time (within five-minute bins). Next, we assign first-class labels to all requests whose tip/fare ratio ranks over the 80th percentile, which is around 0.26. Then, we determine for each location and time bin pair the ratio of first-class requests, which we consider as the probability of them appearing.

Table 5 summarizes the parameters that we use to build upon the PAV baseline scenario to assess the impact of enforcing service-level contracts.

6.4.1. Sensitivity Analysis of Penalization Schemes.

In Table 6, we show for the homogeneous user base scenarios A1 and A2 to what extent manipulating the penalization scheme alters the average performance of the fleet from both user and platform perspectives. For comparison, in the top row of each user base, we place the results for instances similar to those presented in Table 4, in which neither delay nor rejection penalties are applied.

Although in practice the same maximum delays are considered (i.e., 10 and 15 minutes), applying tolerance delays alone (i.e., $\rho = 0$) leads to faster pickups for both SQ1 and SQ2 classes. Because any time spent within the tolerance delay offsets the base fare values, the π_{VFA} policy ends up incorporating a greater sense of urgency. From the perspective of the provider, such a penalty mechanism enables improved user service levels at the expense of slightly lower total contributions. This trade-off is more prominent for the user base A1, for which pickup delays decreased by 41 seconds while increasing by 0.25 percentage points the number of serviced requests, at the expense of $1,287 fewer profits. Moreover, a close analysis of the fleet total time indicates that the tolerance delays remarkably impact the fleet management strategy to service A2, because vehicles spend more time rebalancing and less time parking. These relations suggest that tolerance delays help to achieve more accurate VFAs, which adequately and quickly drive vehicles to the most promising areas.

However, solely adopting tolerance delays only improves the ride experience of serviced users, compensating them according to the inconvenience inflicted. A true commitment to SLCs has to also adequately compensate those who have been through the greatest possible inconvenience, namely, service rejection. By making up for rejections, platforms can improve customer loyalty, once users can trust that the transportation provider genuinely strives to keep consistent service quality, to the point of having "skin in the game" (i.e., risking company profits). Our results show that, besides providing such a guarantee, the application of rejection penalties can also increase the number of requests serviced, with vehicles spending more time rebalancing and less time parked. High penalty factors, however, create a rejection bias against long-distance requests (see the increase in the mean trip distance associated with rejections). Conversely, the trip distance of serviced requests decreases, indicating that the fleet management strategy consists of fulfilling short trips and quickly rebalancing back to high-demand areas.

Ultimately, our findings suggest that both measures are effective to improve service quality, such that we incorporate them in our standard setup. Hence, we adopt the five-minute tolerance delays and rejection penalties equivalent to the base fare (i.e., $\rho = 1$) because these offer a more balanced trade-off regarding users' trips distances. To illustrate how this scheme works in the current transportation setting, in the following, we exemplify how the service provision unfolds for a regular SQ1 user. First, in the case that the request cannot be fulfilled, the platform warns the user (within one minute) and compensates him immediately a rejection penalty equivalent to the base fare. Otherwise, when the user can be serviced, a vehicle takes, on average, 3.48 minutes to pick up him. When the waiting time surpasses the five-minute threshold, a fraction of the base fare is discounted from the user's total trip cost, proportional to the waiting time in the tolerance interval.

6.5. Vehicle Productivity and Fleet Size

Although we have demonstrated that our penalization scheme can improve PAV-fleet productivity and user service levels, Table 6 shows that the platform still cannot service about 10% of the users. Our results indicate that this inability to cover the demand entirely is due to insufficient vehicle supply. As can be seen in Figure 9, under our VFA policy, most vehicles are busy (i.e., rebalancing, picking up, or carrying users) during the demand peak. When rejections start to accumulate from about 6:30 a.m. and on (see Figure 10), we can see that the number of parked vehicles drops dramatically, especially in the myopic policy. In such a scarcity scenario, vehicles tend to reject users whose trips are not economically efficient. Typically, a vehicle is better off parking in high-demand areas than traveling to pick up users in low-demand areas, associated with unpromising future returns. This fleet

Table 6. Sensitivity Analysis on Penalization Schemes

Base fare ($)	Max. delay (min)		Rejection penalties (ρ)	Requests serviced (%)	Pickup delay (min)	Objective function	Trip distance (km)		Fleet total time/status			
	Pickup	Tolerance					Serviced	Rejected	Rebalancing (%)	Picking up (%)	Carrying (%)	Parked (%)
No penalties												
4.8	10	—	—	87.77	3.72	25,540	2.80	4.77	10.31	13.52	30.48	45.69
Delay and rejection penalties (user base A1)												
4.8	5	5	0	88.02	3.31	24,253	2.85	4.39	6.31	12.05	31.17	50.47
			1	88.94	3.48	21,584	2.79	5.05	6.43	12.83	30.81	49.93
			2	88.91	3.56	19,009	2.77	5.23	8.10	13.11	30.53	48.26
No penalties												
2.4	15	—	—	87.77	4.35	17,335	2.93	3.80	7.98	15.75	31.94	44.33
Delay and rejection penalties (user base A2)												
2.4	10	5	0	87.76	4.32	16,956	2.94	3.77	9.07	15.65	31.96	43.32
			1	89.13	4.40	15,776	2.83	4.74	8.61	16.19	31.34	43.85
			2	89.25	4.50	14,489	2.80	5.10	10.28	16.57	30.95	42.20

Notes. Lines under "No penalties" feature results for comparable configurations where no penalties are applied (see Table 4). Performance markers consist of the average results achieved by applying our π_{VFA} policy on the 51 testing instances.

management strategy can also be seen during the busiest period in Figure 9, which features two "idleness peaks" (at around 7:15 a.m. and 8:30 a.m.) where about 50 AVs are parked, waiting for future requests.

6.6. Freelance Fleet Management

In this section, we show how a third-party-owned fleet of FAVs can complement the PAV fleet to improve user service levels. First, we describe how we model the uncertainty associated with the freelance fleet availability (Section 6.6.1), and then we assess the outcome of hiring FAVs (Section 6.6.2).

6.6.1. Modeling FAV Availability. We assume both announcement times and contract durations are drawn from a truncated normal distribution $\psi(\overline{\mu}, \overline{\sigma}, a, b; x)$, where $\overline{\mu}$ and $\overline{\sigma}$ are the mean and variance of the normal distribution, whereas a and b specify the truncation interval. Because our study draws on Manhattan's demand, we also harness the daily commuting patterns of the island to establish realistic announcement times. We consider FAVs arrive between 5:00 a.m. and 9:00 a.m., reaching an arrival peak at 8:00 a.m. This arrival pattern is adapted from the time workers leave home to go to work in Manhattan (see Table 7), where most departures (54.60%) occur between 7:00 a.m. to 9:00 a.m.

Regarding the contract durations, we investigated two scenarios. First, in scenario D1, vehicles are available until the end of the trip sampling interval at 9:00 a.m. Second, in scenario D2, contracts can last from one hour to four hours (viz., the trip sampling interval), and most FAVs are made available for two hours, resulting in the distribution $\psi(2\text{h}, 1\text{h}, 1\text{h}, 4\text{h}; x)$. We generate these contract durations in tandem with announcement times, adjusting durations that surpass the maximum simulation time when added to their announcement times. For this reason, contracts in the range [1 h, 1.5 h] become more common because FAVs arriving after 8:30 a.m. have maximum contract durations of 1.5 hours.

Table 7. Time Leaving Home to Go to Work in Manhattan (U.S. Census Bureau 2015)

Time leaving home	Workers (%)
12:00 a.m. to 4:59 a.m.	1.10
5:00 a.m. to 5:29 a.m.	1.40
5:30 a.m. to 5:59 a.m.	1.10
6:00 a.m. to 6:29 a.m.	4.10
6:30 a.m. to 6:59 a.m.	4.60
7:00 a.m. to 7:29 a.m.	9.70
7:30 a.m. to 7:59 a.m.	10.20
8:00 a.m. to 8:29 a.m.	20.10
8:30 a.m. to 8:59 a.m.	14.60
9:00 a.m. to 11:59 a.m.	33.00

Published in Transportation Science on October 13, 2021 as DOI: 10.1287/trsc.2021.1069.
This article has not been copyedited or formatted. The final version may differ from this version.

Table 8. Summary of the Parameters for On-Demand Hiring

Problem characteristic	Attribute value(s)
Profit margin (β)	100% (PAVs) and 30%(FAVs)
Fleet size ($\|K\|$)	300 PAVs + 200 FAVs
Number of stations (O)	Distribution scenarios: Clustered (C): 64 (0.01*N) Scattered (S): 6,430 (1.00*N)
FAV hiring stochastic process ($\mathcal{F}^O$)	Station: chosen at random from O # vehicles/station: random Announcement time: ψ(8 : 00 a.m., 1h, 5 : 00 a.m., 9 : 00 a.m.; x) Contract duration scenarios: D1: from announcement time until 9:00 a.m. D2: ψ(2h, 1h, 1h, 4h; x)

Regarding the spatial distribution of these vehicles over the map, we investigate two deployment scenarios with increasingly higher numbers of stations $O \subseteq N$:

Clustered (*C*): Stations are drawn from 1% distinct randomly chosen locations ($|O| \leq 64$). In this scenario, AVs cruise to park in a small set of parking lots (e.g., because of incentives, city regulations).

Scattered (*S*): Stations are drawn from all available locations ($|O| \leq 6,430$). This scenario simulates the behavior of AVs which park nearby their owners' locations.

We assume that across all iterations, the station location set O remains stable for all deployment scenarios. Thus, under scenario C, for instance, FAVs always start from the same set of 64 nodes.

Table 8 summarizes the parameters governing an operational scenario in which the fleet comprises PAVs and FAVs. This scenario extends our baseline scenario by allowing extra 200 FAVs into the platform, distributed according to the availability settings mentioned earlier.

6.6.2. Improving Service Quality with On-Demand Hiring. In this section, we offer different perspectives on the results achieved when FAVs, which are available according to the parameters described in Table 8, join the PAV fleet to uphold user SLCs. Tables 9 and 10 present an average performance comparison between the VFA and myopic policies on the testing data set for user base A3. Table 9 shows the influence of each FAV availability scenario (i.e., contract duration and station distribution combination) on the mean objective function, percentage of requests serviced, and pickup delay. Table 10 presents the fleet utilization breakdown, that is, the percentage of the total fleet time spent in each vehicle status.

For the sake of comparison, the bottom row for each policy in both tables presents the results achieved when hiring is not considered. As can be seen from Table 9, in the no-hiring scenario, the VFA policy can service about 18% more requests than the myopic policy, besides providing lower pickup delays, especially for the SQ2 class. Once hiring is enabled, over 90% of requests are picked up regardless of the policy across all scenarios. However, substantial differences can be seen between the policies when different contract durations are considered. On average, we have found that D1 contracts allow a surplus of about 6,000 more minutes of total fleet time than D2. This extra time reflects positively on the platform profits and in the number of requests serviced. Whereas the average difference across station distribution between D1 to D2 contract durations is about 4%

Table 9. Comparison of the Average Objective Function, Number of Requests Serviced, and Pickup Delays Between VFA and Myopic Policies on All FAV Availability Scenarios

					Pickup delay (min)	
Policy	Contract duration	Station distribution	Objective function	Requests serviced (%)	SQ1	SQ2
Myopic	D1	C	18,267	96.73	3.0	4.6
		S	18,306	96.63	3.0	4.7
	D2	C	17,628	92.66	3.1	5.0
		S	17,587	92.16	3.1	5.0
	No hiring		15,273	75.32	3.2	5.0
VFA	D1	C	18,869	98.80	3.1	4.7
		S	18,986	98.90	3.0	4.7
	D2	C	18,811	98.45	3.1	4.7
		S	18,809	98.43	3.0	4.6
	No hiring		17,442	89.25	3.1	4.4

Table 10. Comparison of the Average Fleet Total Time per Status Across All FAV Availability Scenarios Considering the VFA and Myopic Policies

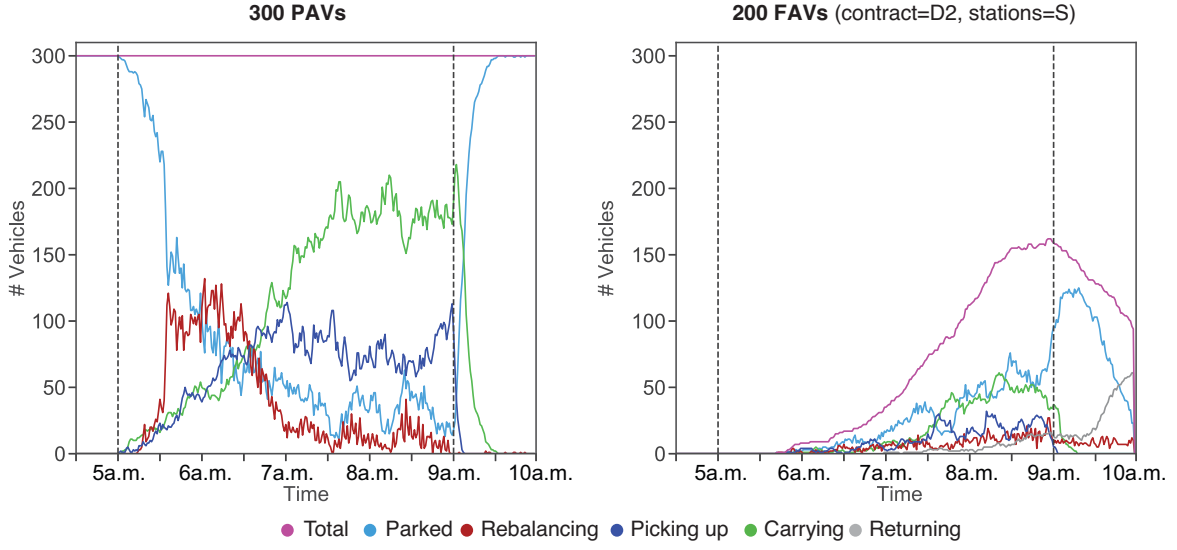
			Rebalancing (%)		Picking up (%)		Carrying (%)		Parked (%)		Returning (%)	
Policy	Contract duration	Station distr.	PAV	FAV	PAV	FAV	PAV	FAV	PAV	FAV	PAV	FAV
Myopic	D1	C	0.49	0.65	13.56	14.79	31.39	22.36	54.57	54.24	—	7.96
		S	0.52	0.88	13.60	14.76	31.46	21.88	54.42	54.23	—	8.26
	D2	C	0.54	1.10	13.64	21.89	31.43	26.06	54.39	38.91	—	12.05
		S	0.57	1.65	13.71	21.69	31.49	25.38	54.24	38.29	—	12.98
	No hiring		0.75	—	14.17	—	31.82	—	53.25	—	—	—
VFA	D1	C	8.14	1.30	15.08	10.23	31.53	20.01	45.25	60.38	—	8.08
		S	7.34	1.23	14.90	10.29	31.61	19.98	46.15	59.73	—	8.76
	D2	C	8.08	1.31	14.98	13.10	31.50	24.08	45.43	51.15	—	10.35
		S	8.09	1.12	14.82	12.48	31.40	24.23	45.68	50.88	—	11.29
	No hiring		8.57	—	15.25	—	31.28	—	44.90	—	—	—

points in the myopic policy, this difference is less than 0.5 percentage points in the VFA policy. The same pattern can be seen in the difference between SQ2 user pickup delays, which differ dramatically across the contract durations scenarios under the myopic policy. Hence, by better managing both vehicle types, the VFA policy can sustain high service levels for all user bases, even under more strict FAV availability. From a different perspective, FAV owners wanting to improve the odds of renting out their vehicles have to set up service availability adequately, such that the platform has enough time to rebalance and return these vehicles.

Moreover, as confirmed by the total fleet time breakdown in Table 10, FAVs tend to stay idle more often under the proposed VFA policy. Although not highlighted by the objective functions because of our low-cost setup, this characteristic is crucial for providers, especially in the light of vehicle automation, when induced demand due to ease of use may play a significant role. City managers are increasingly concerned about traffic, and proposals for imposing congestion charges abound. Therefore, a platform owner is generally better off using fewer vehicles, especially FAVs, which need to spend extra time returning to their origin stations. Ultimately, the proposed VFA policy can find a compromise between service levels and vehicle activity, prioritizing the own fleet over outside hire to address requests.

Figure 12 further illustrates the impact of including 200 FAVs to service user base A3 on a single testing instance. FAVs arrive according to the stochastic

Figure 12. (Color online) Number of Vehicles per Status (Parked, Rebalancing, Servicing Passengers, and Returning to Station) by One-Minute Step Separated by Fleet Type for a Single Testing Instance



Note. The total number of PAVs is constant throughout the whole time horizon, whereas the number of FAVs varies according to a stochastic process.

Published in Transportation Science on October 13, 2021 as DOI: 10.1287/trsc.2021.1069.
This article has not been copyedited or formatted. The final version may differ from this version.

process $\mathcal{F}^{O}$ assuming contract duration scenario D2 and station distribution scenario *S*. It can be seen that the vehicle/status distribution still resembles the results achieved by a PAV-only fleet (see Figure 9), showing that the inclusion of FAVs does not disrupt the PAV-fleet operation significantly. Because we assume 70% of the profits accrued by FAVs belong to their owners, using FAVs returns fewer profits to the platform while inflicting similar operational costs. That is the main reason why the service-level improvement by hiring vehicles (about 10% for the VFA policy) does not translate proportionally into the profits. However, maintaining high service levels results in increased customer satisfaction, which may improve the platform's reputation and generate a higher turnover in the long run.

7. Conclusions

Mobility-on-demand services can only challenge self-owned mobility products if they can offer a competitive service quality. Service quality is based on two core elements, namely, maintaining personalized service levels and making up for inconveniences (i.e., service-level violations) accordingly.

In this paper, we propose a solution to control service quality on an operational level using a learned-based optimization approach. We introduce a model for a dynamic and stochastic dial-a-ride problem arising on an AMoD platform that hires idle AVs to maintain consistent user service levels. Developing an approximate dynamic programming algorithm, we iteratively improve a policy to dispatch and rebalance both platform- and third-party vehicles on a real-world street network of Manhattan. The proposed policy deals with two seldomly considered sources of uncertainty, namely, (i) the spatiotemporal distribution of user service-level preferences and (ii) the availability of third-party vehicles. Whereas (i) allows providers to better address heterogeneous user expectations by rebalancing more vehicles to areas featuring high demanding users, (ii) enables the learning of routing policies that take into account when, where, and how many third-party vehicles are expected to appear throughout the planning horizon.

The proposed approach improves service quality for the ridesharing platform customers in multiple ways. First, penalizing both excessive delays and rejections following SLCs is shown to be an effective measure to increase the number of requests serviced. Second, the policy learned by sampling the demand from a particular weekday was shown to be generic enough to adequately address the demand patterns of all similar weekdays throughout a whole year. Without any hiring, such a policy consistently outperforms a reactive optimization policy, servicing on average about 18% more requests. Moreover, although both policies manage to service most requests when hiring is considered, the proposed policy has been shown to do it more efficiently, using fewer FAVs, and providing better service levels.

We conduct experiments on the historical Manhattan taxi demand considering a variety of fleet and demand configuration scenarios. Using a baseline scenario featuring only PAVs and homogeneous users, we define a hierarchical aggregation structure to approximate value functions of unvisited states. Besides time and space, the proposed layers also consider FAV-specific characteristics, such as contract duration and home station location. In particular, the spatial hierarchical aggregation structure improves existing configurations in which locations aggregate up to ad hoc regions. We propose a minimum set covering formulation to optimally determine regions whose nodes can be accessed from a regional center within a maximal time limit. This formulation offers a more robust and versatile approach to hierarchical spatial aggregation because it automatically captures the peculiarities of any transportation network.

Optimal regional centers are also used to set up several rebalancing strategies, in which vehicles can move to a subset of neighboring centers, determined through different maximal time limits. The obtained results show that rebalancing to short-range regional centers allows vehicles to incrementally escape from perpetually low-demand areas, besides offering a good compromise regarding computational time. Because we adopt short intervals, these rebalancing movements occasionally result in multiperiod travel times (i.e., at decision time, vehicles are still acting on decisions from previous periods). We show that by actively lowering VFAs of postdecision states of farther rebalancing targets improves the performance of our solution for test cases with increasingly higher rebalancing distances.

Moreover, we develop a high-resolution state representation in which the spatial attributes correspond to discretized GPS coordinates (rather than grids, zones, or areas) and periods are no longer than one minute to comply with the demanding expectations of current MoD users. Such characteristics prevent our policy from incurring into infeasible (concerning the infrastructure capacity) or illegal (concerning the city regulations) decisions altogether in real time. Ultimately, making use of the underlying street network allows us not only to comply with real-world constraints but also improves the solution quality. Our experiments demonstrate that constraining the maximum number of vehicles inbound to each intersection is crucial to achieving stable VFAs, because these constraints exempt us from modeling the behavior of nonlinear approximations.

Published in Transportation Science on October 13, 2021 as DOI: 10.1287/trsc.2021.1069.
This article has not been copyedited or formatted. The final version may differ from this version.

This research can be extended in many promising directions. First, one could focus on designing an inverse formulation to determine the minimum number of company-owned vehicles necessary to complement third-party fleets available according to varying stochastic distributions. Second, the requirements of independent owners could take into consideration alternative parameters. For example, they could establish minimum profit margins or compensations to join ridesharing platforms. As a result, platforms would have to consider these parameters to achieve balanced solutions, weighing customer dissatisfaction and outsourcing costs. Additionally, by considering time travel uncertainty, service quality contracts would have to be further adapted to compensate users beyond the violations previously described. Ultimately, this uncertainty could lead to service time window violations on the supply side, such that platforms could also set up contracts prescribing compensations for inconvenienced FAV owners. Last, one could also consider the impact of cities' traffic management policies (e.g., congestion pricing, empty-vehicle fees, parking costs) on platform operations.

References

Al-Kanj L, Nascimento J, Powell WB (2020) Approximate dynamic programming for planning a ride-hailing system using autonomous fleets of electric vehicles. *Eur. J. Oper. Res.* 284(3):1088–1106.

Alonso-Mora J, Wallar A, Rus D (2017) Predictive routing for autonomous mobility-on-demand systems with ride-sharing. *Proc. IEEE/RSJ Internat. Conf. Intelligent Robots Systems* (Institute of Electrical and Electronics Engineers, Vancouver, Canada), 3583–3590.

Alonso-Mora J, Samaranayake S, Wallar A, Frazzoli E, Rus D (2017) On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment. *Proc. Natl. Acad. Sci. USA* 114(3):462–467.

Archetti C, Savelsbergh M, Speranza MG (2016) The vehicle routing problem with occasional drivers. *Eur. J. Oper. Res.* 254(2):472–480.

Arslan AM, Agatz N, Kroon L, Zuidwijk R (2019) Crowdsourced delivery—A dynamic pickup and delivery problem with ad hoc drivers. *Transportation Sci.* 53(1):222–235.

Campbell H (2018) Who will own and have propriety over our automated future? Considering governance of ownership to maximize access, efficiency, and equity in cities. *Transportation Res. Record* 2672(7):14–23.

Castiglione J, Cooper D, Sana B, Tischler D, Chang T, Erhardt G, Roy S, Chen M, Mucci A (2018) TNCs & congestion. Technical report, San Francisco County Transportation Authority, San Francisco.

Dahle L, Andersson H, Christiansen M (2017) The vehicle routing problem with dynamic occasional drivers. *Proc. Eighth Internat. Conf. Comput. Logist.* (Springer, Southampton, UK), 49–63.

Dahle L, Andersson H, Christiansen M, Speranza MG (2019) The pickup and delivery problem with time windows and occasional drivers. *Comput. Oper. Res.* 109(September):122–133.

Fagnant DJ, Kockelman KM (2018) Dynamic ride-sharing and fleet sizing for a system of shared autonomous vehicles in Austin, Texas. *Transportation* 45(1):143–158.

Furuhata M, Dessouky M, Ordóñez F, Brunet ME, Wang X, Koenig S (2013) Ridesharing: The state-of-the-art and future directions. *Transportation Res. Part B: Methodological* 57(November):28–46.

George AP, Powell WB (2006) Adaptive stepsizes for recursive estimation with applications in approximate dynamic programming. *Machine Learn.* 65(1):167–198.

George A, Powell WB, Kulkarni SR (2008) Value function approximation using multiple aggregation for multiattribute resource management. *J. Machine Learn. Res.* 9(68):2079–2111.

Gueriau M, Dusparic I (2018) SAMoD: Shared autonomous mobility-on-demand using decentralized reinforcement learning. *Proc. 21st Internat. Conf. Intelligent Transportation Systems* (ITSC, Maui, HI), 1558–1563.

Ho SC, Szeto WY, Kuo YH, Leung JM, Petering M, Tou TW (2018) A survey of dial-a-ride problems: Literature review and recent developments. *Transportation Res. Part B: Methodological* 111:1–27.

Hyland MF, Mahmassani HS (2017) Taxonomy of shared autonomous vehicle fleet management problems to inform future transportation mobility. *Transportation Res. Record* 2653(1):26–34.

Iglesias R, Rossi F, Wang K, Hallac D, Leskovec J, Pavone M (2018) Data-driven model predictive control of autonomous mobility-on-demand systems. *Proc. IEEE Internat. Conf. Robotics Automation* (Institute of Electrical and Electronics Engineers, Piscataway, NJ), 6019–6025.

Le TV, Stathopoulos A, Van Woensel T, Ukkusuri SV (2019) Supply, demand, operations, and management of crowd-shipping services: A review and empirical evidence. *Transportation Res. Part C: Emerging Tech.* 103(3):83–103.

Lee A, Savelsbergh M (2015) Dynamic ridesharing: Is there a role for dedicated drivers? *Transportation Res. Part B: Methodological* 81:483–497.

Lin K, Zhao R, Xu Z, Zhou J (2018) Efficient large-scale fleet management via multi-agent deep reinforcement learning. *Proc. 24th ACM SIGKDD Internat. Conf. Knowledge Discovery Data Mining* (Association for Computing Machinery, London, UK), 1774–1783.

Ma S, Zheng Y, Wolfson O (2015) Real-time city-scale taxi ridesharing. *IEEE Trans. Knowledge Data Engrg.* 27(7):1782–1795.

Narayanan S, Chaniotakis E, Antoniou C (2020) Shared autonomous vehicle services: A comprehensive review. *Transportation Res. Part C: Emerging Tech.* 111:255–293.

Oyola J, Arntzen H, Woodruff DL (2018) The stochastic vehicle routing problem, a literature review, part i: Models. *EURO J. Transportation Logist.* 7(3):193–221.

Powell WB (2011) *Approximate Dynamic Programming: Solving the Curses of Dimensionality*, 2nd ed. (John Wiley and Sons, Hoboken, NJ).

Powell WB, Simão HP, Bouzaiene-Ayari B (2012) Approximate dynamic programming in transportation and logistics: A unified framework. *EURO J. Transportation Logist.* 1(3):237–284.

Santos DO, Xavier EC (2015) Taxi and ride sharing: A dynamic dial-a-ride problem with money as an incentive. *Expert Systems Appl.* 42(19):6728–6737.

Savelsbergh M, Sol M (1998) Drive: Dynamic routing of independent vehicles. *Oper. Res.* 46(4):474–490.

Shoup D (2017) *The High Cost of Free Parking*, updated ed. (Routledge, London).

Simão HP, Day J, George AP, Gifford T, Nienow J, Powell WB (2009) An approximate dynamic programming algorithm for large-scale fleet management: A case application. *Transportation Sci.* 43(2):178–197.

Topaloglu H, Powell WB (2006) Dynamic-programming approximations for stochastic time-staged integer multicommodity-flow problems. *INFORMS J. Comput.* 18(1):31–42.

Toregas C, Swain R, ReVelle C, Bergman L (1971) The location of emergency service facilities. *Oper. Res.* 19(6):1363–1373.

Tsao M, Iglesias R, Pavone M (2018) Stochastic model predictive control for autonomous mobility on demand. *Proc. 21st Internat. Conf. Intelligent Transportation Systems* (ITSC, Maui, HI), 3941–3948.

U.S. Census Bureau (2015) Time of departure to go to work (Manhattan borough, NY). Table B08302, 2015 American Community Survey five-year estimates, U.S. Census Bureau, Suitland, MD.

Vazifeh MM, Santi P, Resta G, Strogatz SH, Ratti C (2018) Addressing the minimum fleet problem in on-demand urban mobility. *Nature* 557(7706):534–538.

Wen J, Zhao J, Jaillet P (2017) Rebalancing shared mobility-on-demand systems: A reinforcement learning approach. *Proc. IEEE 20th Internat. Conf. Intelligent Transportation Systems* (ITSC, Yokohama, Japan), 220–225.

Xu Z, Yin Y, Ye J (2020) On the supply curve of ride-hailing systems. *Transportation Res. Part B: Methodological* 132:29–43.

Zhang R, Rossi F, Pavone M (2016) Model predictive control of autonomous mobility-on-demand systems. *Proc.* 2016 *IEEE Internat. Conf. Robotics Automation* (ICRA, Stockholm, Sweden), 1382–1389.