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

TRANSPORTATION SCIENCE
Articles in Advance, pp. 1–23
ISSN 0041-1655 (print), ISSN 1526-5447 (online)

Data-Driven Policies for the Online Ride-Hailing Problem with Fairness

Shachaf Ben-Gal,[a] Michal Tzur[a,*]

[a] Department of Industrial Engineering, Iby and Aladar Fleischman Faculty of Engineering, Tel Aviv University, Tel Aviv 6997801, Israel
*Corresponding author

Contact: shachafbengal@gmail.com, https://orcid.org/0009-0006-2079-9181 (SB-G); tzurm@tauex.tau.ac.il, https://orcid.org/0000-0001-6931-8058 (MT)

Received: July 5, 2023
Revised: February 13, 2024; December 2, 2024
Accepted: March 5, 2025
Published Online in Articles in Advance: March 26, 2025

https://doi.org/10.1287/trsc.2023.0068

Copyright: © 2025 INFORMS

Abstract. Ride-hailing is a prevalent transportation service that facilitates mobility in urban areas. A ride-hailing service system encompasses several research problems, including the operational assignment of on-demand ride requests to vehicles in real time. The literature suggests various approaches to address similar systems, mainly optimizing the system efficiency, but recent studies pointed out that these systems are likely to cause geographical unfairness among passengers. Such unfairness may imply, for example, that requests whose origin or destination is far from centralized locations may suffer from excessive service rejections. In this paper, we suggest a data-driven approach to design an online assignment policy to overcome this phenomenon. We formulate the online ride-hailing problem with fairness that seeks to maximize both efficiency and geographical fairness in the system while achieving an adequate balance between them. To solve this problem, we offer a new general method to develop online assignment policies based on solutions for offline versions of the problem. The new method suggests extracting information from these solutions to guide real-time assignment decisions, which are chosen using a data-driven algorithm. With a simulation study, we examine the performance of our online policies relative to dispatching rules using synthetic random data that represent a real city layout and movement. Some of these rules are commonly used in practice, and some are more sophisticated ones. Our results demonstrate the viability of our approach to designing online policies. Compared with other dispatching rules, the experiments show that the generated policies maintain a better trade-off between efficiency and geographical fairness and preserve stable performance regardless of the instance size in different system settings.

Funding: This research was partially supported by the Israel Science Foundation [Grant 2772/24]. S. Ben-Gal was partially supported by a scholarship from the Shlomo Shmeltzer Institute for Smart Transportation at Tel Aviv University and by The Israeli Smart Transportation Research Center.
Supplemental Material: The online appendix is available at https://doi.org/10.1287/trsc.2023.0068.

Keywords: ride-hailing • data-driven policies • fairness • heuristics

1. Introduction

Ride-hailing is a prevalent transportation service that helps facilitate mobility within urban areas. Specifically, a ride-hailing service offers passengers who want to start traveling within a short time, from one location (origin) to another (destination), a way to order a vehicle for their ride, typically by submitting the request through a mobile application. The service operates a fleet of vehicles. For every ride request, a decision needs to be made regarding which vehicle to assign to it or whether to reject the request in cases in which the system allows it or cannot fulfill the request. These decisions are made by a centrally controlled dispatcher, and they need to be made instantaneously for requests that arrive in a stochastic and unplanned manner.

Commonly, the controllers of these systems apply simple dispatching rules for vehicle assignment decisions. For example, they may assign the vehicle closest to the request's origin or the vehicle that arrives there earliest. As these examples demonstrate, these rules are typically greedy and do not consider that the assigned requests influence the opportunity to serve subsequent requests. Moreover, they aim to achieve efficiency-related goals for the entire system, which may create geographical unfairness among passengers located in different areas.

Such unfairness may imply, for example, that requests whose origin or destination is far from centralized locations may suffer from excessive service rejections. In this context, Pan et al. (2020) showed that the quality of transportation services (measured by the number of trip pickups during peak times) is still spatially different between different areas in New York City. Similarly, the analysis of Jin, Kong, and Sui (2019) demonstrated that only 20% of the population in New York City comprises 95% of Uber customers, suggesting that there are unequal transportation opportunities in different city boroughs. Additionally, Thebault-Spieker, Terveen, and Hecht (2017) argued that geographic principles, such as variation in population density and income levels of different areas in Cook County in the United States, were responsible for structural geographic biases in the effectiveness of Uber services as measured by the average waiting times.

Fairness concerns are naturally of interest to nonprofit organizations (Eisenhandler and Tzur 2019) but also, to profit-driven organizations. One reason is that in unfair systems, customers from undersupplied geographical areas, such as low-density areas, are likely to eventually decrease their usage of the respective system and by that, reduce the system's long-term income (Chen et al. 2023). Moreover, such efforts to maintain an equitable service are aligned with the environmental, social, and governance principle used by socially conscious investors to screen potential investments. For these reasons, even private companies may desire to serve the entire region, including low-density areas. When combined with efficiency, the system controller can achieve an appropriate balance between the two.

In this paper, we present the online ride-hailing problem with fairness, which aims to achieve both efficiency and geographical fairness among passengers. We adopt the objective function introduced by Eisenhandler and Tzur (2019) for distributing food donations to welfare agencies in an efficient and equitable manner. This objective is versatile and supports a variety of definitions for the groups between which one wants to achieve equity according to the different needs of the system. We consider a system in which decisions need to be made instantaneously (rather than after accumulating a batch of requests), a setting that received limited attention in the literature on ride-hailing. This offers a passenger-focused setting in the sense that passengers do not need to wait before they receive the system response. Our model does not incorporate the issue of pricing and profitability. It is reasonable to expect that serving low-density outskirt areas would cost more for the riders who request them, and thus, the system would get compensated for possibly higher costs. Yet, the challenge that we address is making good vehicle assignments to requests so that low-density outskirt areas can be served adequately.

Toward this goal, we develop new assignment policies to manage centralized controlled ride-hailing services. A recent justification to consider centralized systems is provided by Kullman et al. (2022, pp. 775–776), who claim that "[w]hereas fleet control is somewhat centralized today, it is likely to become increasingly centralized as ride-hail companies adopt autonomous vehicles." Our solution approach creates online assignment policies based on solutions to offline versions of the problem. Essentially, our approach creates offline a *reference set*, which organizes information extracted from static solutions that optimize the efficiency and fairness objective. Then, it exploits this information using a data-driven method and makes vehicle assignment decisions in real time based on the system's state.

The contributions of this paper are threefold. First, we introduce a new method of utilizing solutions for offline versions of the problem to guide online decisions. The novelty of this method is in combining the creation of offline solutions that explicitly consider constraints suitable for the online setting with a second stage that makes real-time decisions by developing tailored distance and voting functions. The method is general as it can be applied in a similar way to other problems whose offline versions can be found. Second, we demonstrate how to apply this general method to the online ride-hailing problem with fairness. Part of this contribution includes the introduction of new mixed-integer linear programming (MILP) formulations that are based on the MILP formulation by Bertsimas, Jaillet, and Martin (2019) for a static version of the ride-hailing problem. However, our formulations enable generating vehicle assignments that are applicable online (i.e., they are based on the currently available information and do not assume knowledge of the future). Third, we conduct a numerical study to evaluate the performance of our solution approach, examining problem instances with various system settings and presenting the viability of our approach. Within this examination, our study proposes other *area-based dispatching rules*, suggesting new criteria for vehicle assignment rules based on the spatial distribution of the vehicles. To the best of our knowledge, this is the first time that such rules have been used for ride-hailing vehicle assignments. Finally, we also consider an alternative objective function and demonstrate the superiority of our approach for this case as well.

2. Literature Review

In this section, we review the literature regarding three aspects of the problem: the connection between ride-hailing and other dynamic routing problems, routing problems with fairness, and methodologies to solve real-time vehicle assignment problems. These are described in Sections 2.1–2.3, respectively.

2.1. The Connection Between Dynamic Ride-Hailing and Other Dynamic Routing Problems

Ride-hailing services, sometimes called *ride-sourcing* (Wang and Yang 2019), have been studied extensively in the literature on dynamic pickup and delivery problems (D-PDPs). Specifically, ride-hailing is a variant of the dynamic *dial-a-ride problem* (D-DARP), a family of problems where the routed goods are human passengers (Berbeglia, Cordeau, and Laporte 2010). The D-DARP designs adjustable routes between pairs of pickup and drop-off locations that are either known in advance or revealed during the operation (Cordeau and Laporte 2007). When the problem is fully dynamic, the problem is sometimes referred to as a *ride-splitting* or *ride-pooling* problem (e.g., Alonso-Mora et al. 2017). In ride-hailing services, the requests arrive dynamically, but each vehicle serves one ride request at a time and transports passengers directly from their origin to their destination. When *time windows* exist in ride-hailing, they typically refer only to the pickup time (Bertsimas, Jaillet, and Martin 2019), representing the patience of passengers waiting for service. Few works allow for prebooking of a request in advance (e.g., Bertsimas, Jaillet, and Martin 2019, Duan et al. 2020), so the information about prebooked requests is revealed before the service is needed (i.e., before the start of the pickup time window).

The *ride-sharing* service matches occasional drivers and riders in real time based on similar itineraries and time schedules (Dong et al. 2018, Ausseil, Pazour, and Ulmer 2022). This matching decision is required on short notice, like ride-hailing. Agatz et al. (2012) outlined the characteristics of ride-sharing systems, based on which we can identify several differences between ride-sharing and ride-hailing. The main difference is that drivers in a ride-sharing service are independent participants who perform their trips, so the matching of the driver to a rider is a one-time and nonrecurring event. Thus, the rider's destination is not essential for future services. In addition, the drivers' original routes limit the possible routing decisions, which are represented as deviations from it. Other applications of pickup and delivery problems (PDPs), such as *same-day delivery* (SDD) and *food deliveries*, also decide on vehicle assignments to requests, but they transport objects (not people). These assignment decisions allow more time for the delivery of the requests (e.g., 90 minutes for food deliveries in Reyes et al. 2018), and upon their arrival, only an acceptance (or rejection) decision is necessary (e.g., Ulmer et al. 2021).

The literature on ride-hailing services mostly aims to optimize efficiency measures: maximizing profit (Dickerson et al. 2021) and the number of served requests (Lowalekar, Varakantham, and Jaillet 2018) or minimizing costs (Wang and Bei 2022) and waiting times (Feng, Kong, and Wang 2021). See also Wang and Yang (2019) for a recent review of ride-hailing literature.

2.2. Fairness in Ride-Hailing

Several studies pointed out the existence of unfairness in ride-hailing systems with respect to both the drivers of the vehicles and the passengers. The literature primarily focuses on unfairness toward the former, which is mainly reflected in drivers' differences in earned income (see Bokányi and Hannák 2020). Some works suggested methods for balancing the income more equally among the drivers while preserving the overall system's revenue (see Lesmana, Zhang, and Bei 2019, Sühr et al. 2019, and Sun et al. 2022).

However, the literature regarding passengers' fairness in ride-hailing systems is scarce. Nanda et al. (2020) suggested a parametrized linear program (LP)-based assignment algorithm to balance the system's profit and passengers' fairness, which was defined as the minimum ratio of matches to the number of arrivals among different request types. Similar fairness objectives were suggested by Ma, Xu, and Xu (2021) for a ride-hailing system, where the origin-destination zone pairs of requests defined the groups of requests. Their objective was to maximize the minimum service rate across all groups, which is primarily a fairness objective that promotes efficiency only through the group with the lowest service rate. Promoting fairness and efficiency simultaneously involves a known trade-off in resource allocation problems (Bertsimas, Farias, and Trichakis 2012). The models of the above two studies are inherently different from ours: for example, because of their assumptions that the set of online requests is known in advance, whereas only their sequence is stochastic and the assumption that the agents (vehicles) are offline (i.e., they leave the system after serving a request). Miao et al. (2016) dealt with a street-hailing taxi service, which decides the region that vacant taxis should go to, aiming to serve different regions equally and minimize the idle driving distance. They introduced a supply-demand mismatch penalty for the fairness objective, which measures the difference between the percentage of taxis sent to each region and the percentage of requests that originated there. The trade-off between the two goals was controlled with an arbitrarily chosen parameter.

Geographical aspects of passengers' fairness were accommodated in other PDPs. Regarding the ride-splitting literature, several works suggested adding incentives to promote the service of ride requests from zones with higher rejection rates and examined different implementation schemes (Raman, Shah, Dickerson 2021, Schuller, Fielbaum, and Alonso-Mora 2021, Kumar, Vorobeychik, and Yeoh 2022). Within an offline version of the ride-sharing problem, Lee and Savelsbergh (2015) shed light on the ride characteristics of passengers typically rejected by ad hoc drivers. Their trips were shorter and from areas with a lower participation density. Chen et al. (2023) discovered a similar characterization in the context of the SDD problem. They discussed the geographical unfairness of highly rejected service locations by a central dispatcher. Their goal was to balance the overall acceptance rate and fairness in the system, which was measured by the minimum service acceptance rates across all of the regions. This objective accounts for the worse-off region only. Other problems, such as Neria and Tzur (2024), deal with a D-PDP with fairness while determining both allocation and routing decisions together.

2.3. The Assignment Problem in Ride-Hailing

Several optimization challenges arise in the implementation of a ride-hailing service; among them is the real-time assignment of vehicles to dynamically arriving requests. Real-time vehicle assignment, sometimes called matching or order dispatching, is one of the central components of a successful dynamic ride-hailing system as it indirectly affects the spatial distribution of the vehicles in the service region, which then determines the system's service level. Wang and Bei (2022) categorized two algorithmic approaches for real-time vehicle assignments. The *event-based* approach immediately decides for each arriving request whether it is assigned (and by which vehicle) or rejected. The *batch-based* approach uses a static and offline method to iteratively find the optimal solution for requests gathered in a short time interval. We adopt the former as it better represents real-time decision making and offers a better service to passengers by saving them waiting time until they receive the system's response.

The literature discusses three main methodologies to address the real-time vehicle assignment problem: (i) solving a bipartite graph problem, (ii) reoptimization, and (iii) reinforcement learning. We outline them briefly, highlighting the differences from our solution approach.

Inspired by its two-sided structure, many works represent a ride-hailing service with a bipartite graph, with vehicles on one side and requests on the other. Using this model, the literature offers two main approaches to solving the assignment problem. One approach iteratively solves the *bipartite matching problem* for a batch of requests and available vehicles using an integer linear programming formulation (Maciejewski, Bischoff, and Nagel 2016, Lesmana, Zhang, and Bei 2019, Sühr et al. 2019). However, this is not a fully online method and provides a myopic solution as only one batch of requests is considered at each decision period.

Another approach adapts the *online bipartite matching* problem suggested by Karp, Vazirani, and Vazirani (1990), in which one set of vertices is known in advance (vehicles), whereas the other set arrives in an online fashion (requests) along with their edges to specify eligible assignments. For every such request, immediate and irrevocable vehicle assignment is determined using a randomized algorithm. The literature suggests different vehicle sampling schemes, universally based on an offline solution of a benchmark LP formulation (see Nanda et al. 2020 and Ma, Xu, and Xu 2021). Compared with our usage of offline solutions, the benchmark LP is solved once for an "average" scenario using the mean values of the requests arrival process without considering specific instances. Moreover, this model does not assume any notion of passengers' patience, and the system state at the request's arrival time does not affect the algorithm's outcome. It is reflected by the usage of a constant vehicle distribution for all of the requests (implied from the single solution of the benchmark LP) and the limitation of which vehicle can serve the request.

Another methodology, *reoptimization*, aims to overcome the short-sight future of ride-hailing systems and allows for modifying nonimmediate decisions after more information is disclosed. Bertsimas, Jaillet, and Martin (2019) allow for changing the specific vehicle that serves an accepted prebooked request until its service is needed. Tavor and Raviv (2023), who consider the assignments of cars to serve future demand and rebalance automated vehicles in a RoboTaxi ride-hailing system, repeatedly solve a linear program at the beginning of each period with a rolling-horizon procedure that applies its solution to the current period with some adjustments. Miao et al. (2016) and Lowalekar, Varakantham, and Jaillet (2018) combine multiple scenarios of possible request arrivals in the decision process of each period to determine the immediate decisions for the known requests. Bent and Van Hentenryck (2004) introduced the multiple scenario approach (MSA), in which decisions that need to be made in real time are selected based on a *distinguished plan* chosen by a *consensus function* that is applied to several deterministic solutions of the generated scenarios. This concept was later used by, for example, Voccia, Campbell, and Thomas (2019) and Ausseil, Pazour, and Ulmer (2022) with different consensus functions in SDD and ride-sharing applications. The bipartite matching and reoptimization methods may be viewed according to the framework for sequential decision problems proposed by Powell (2019) as look-ahead approximation policies, where the approximation is a result of limiting the horizon.

The last methodology formulates the assignment problem as a *Markov decision process* and solves it with a *deep reinforcement learning* framework, which is a Value Function Approximation (VFA) policy (Powell 2019). This framework is used to approximate the expected future values of an assignment: a vehicle-request pair. These values are later combined as the objective coefficients in the bipartite matching problem to better estimate the effect of an assignment on future gains (Raman, Shah, Dickerson 2021, Beirigo, Schulte, and Negenborn 2022, Kumar, Vorobeychik, and Yeoh 2022, Sun et al. 2022). Alternatively, they are used in a centralized greedy allocation heuristic that incrementally assigns the highest-value vehicle to each arriving request (Sultana et al. 2021, Kullman et al. 2022). Heitmann et al. (2023) address a ride-hailing system where vehicles can be shared among various ride requests. Their solution framework integrates VFA and MSA to determine whether to serve ride requests and the corresponding routes for the accepted requests, respectively.

3. Formal Problem Description

In this section, we formally describe the online ride-hailing problem with fairness. The system operates in a service region divided into a set of nonoverlapping geographical *zones*, $\mathcal{Z}$. We denote by t_{ij} the travel time from zone i to zone j ($i, j \in \mathcal{Z}$) calculated based on the shortest path between them, regardless of the exact location within zone i and zone j. We assume that t_{ij} is time independent.

There are T discrete time periods in which ride requests may randomly arrive to the system according to a stochastic process. The time periods are assumed to be short enough so that at most, one request may enter the system in each of them. A *request* k is for a ride from origin zone o_k to destination zone d_k ($o_k, d_k \in \mathcal{Z}$); it arrives at time a_k ($a_k \in 1, \ldots, T$) and has an expiration time e_k ($e_k > a_k$) that specifies its latest accepted pickup time (after which the passenger cancels the request). We assume that if requests k_1 and k_2 satisfy $a_{k_1} < a_{k_2}$, then $k_1 < k_2$ (i.e., the request indices indicate their order of arrival). Each request is required to be served with no interruptions. Ride-pooling between requests is not allowed.

The system operates a set of vehicles $\mathcal{V}$ that serve the requests. At any given time, each *vehicle* v can be *vacant* when it does not serve any request or *occupied* if assigned to a request. At any point of time and in particular, upon the arrival of request k at a_k, the *vacancy time* and *zone*, τ_v ($\tau_v \geq a_k$) and z_v ($z_v \in \mathcal{Z}$), respectively, identify when and where vehicle v becomes vacant after completing its assigned requests. These values are updated during the system operation because of assignment decisions. The system can clearly assign a newly arriving request to a vacant vehicle, but it can also assign a new request to an occupied vehicle. In the latter case, once the vehicle completes serving its previously assigned requests, it will cruise to the origin and pick up the new request before its expiration time. When completing the last request assigned to it, the vehicle becomes vacant and does not move from the destination zone until the system assigns to it a new request. Notice that we distinguish between the pickup and the service time of a request. The former refers to the point of time following the cruising of the vehicle to the origin of the request (i.e., when the request is picked up). The latter refers to the time duration, starting from the cruising time of the assigned vehicle v and going to the end of the trip time of the request when it is dropped off at the destination zone.

Given a request k, a vehicle v is *available* if it can reach the request's origin zone o_k before the request expiration time e_k. We define $\mathcal{V}_k = \{v \in \mathcal{V} \mid \tau_v + t_{z_v o_k} \leq e_k\}$ as the *set of vehicles available to serve request* k. Upon request k's arrival to the system at time a_k, the system must immediately decide which vehicle v should be assigned to this request among the available vehicles, $v \in \mathcal{V}_k$. If vehicle v serves request k, the system updates its vacancy time and zone to $\tau_v + t_{z_v o_k} + t_{o_k d_k}$ and d_k, respectively. We denote any such decision with a decision variable x_{vk}, which equals one if vehicle v serves request k and zero otherwise. If there is no available vehicle to serve request k ($\mathcal{V}_k = \emptyset$), the system rejects the request, and it leaves the system. Rejecting request k when there are available vehicles to serve it ($\mathcal{V}_k \neq \emptyset$) is allowed in some systems because such an option might be beneficial for serving future requests. This can happen, for example, when the vehicle's current vacancy zone z_v is preferred over the destination zone d_k or when serving the current request occupies a vehicle for a long time. However, this is not allowed in most of our analyses except for one model variation, in which we consider such an extension specifically.

As discussed in Section 1, we focus in this paper on a ride-hailing service whose objective is maximizing its expected efficiency and geographical fairness. A prevalent efficiency measure of a ride-hailing system, which we adopt in this study as well, is the total number of requests served in the entire service region. Therefore, we denote the system's *efficiency* by F, where $F = \sum_k \sum_{v=1}^{|\mathcal{V}|} x_{vk}$.

To define the geographical fairness of the system, we first define a set of requests' groups according to the requests' geographical characteristics: for example, requests from the region's center to the region's outskirts. We denote a *geographical group* G_i ($G_i = \{k_1, k_2, \ldots\}$) as a set of request indices that have entered the system and share similar geographical characteristics, and we denote the size of group G_i as r_{G_i} ($|G_i| \equiv r_{G_i}$). We denote the number of served requests in the group as x_{G_i} ($x_{G_i} = \sum_{k \in G_i} \sum_{v=1}^{|\mathcal{V}|} x_{vk}$) so that the fill rate of a group of requests G_i is x_{G_i}/r_{G_i}, the ratio between the number of served requests in G_i and the number of requests in G_i. Given that there are M groups of requests between which we want to achieve equity, the efficiency measure can also be represented as $F = \sum_{i=1}^{M} x_{G_i}$.

Finally, we measure the *fairness* in the system as one minus the Gini coefficient (a term borrowed from the economics literature used to measure inequity) with respect to the fill rates among the M groups. According to the mathematical representation suggested by Kendall and Stuart (1963) and accounting for the number of requests in each group as proposed by Mandell (1991), the Gini coefficient in our problem (denoted by GI) is equal to $GI = \sum_{i=1}^{M}\sum_{j=i+1}^{M}|q_{G_j}x_{G_i} - q_{G_i}x_{G_j}|/\sum_{i=1}^{M}x_{G_i}$, where $q_{G_i} = r_{G_i}/\sum_{j=1}^{M}r_{G_j}$ is the proportion of requests in group G_i from the total number of requests that entered the system.

Given the expressions for the efficiency and fairness measures in the system, we aim to maximize both measures while achieving an adequate balance between them. For that, we adapt the objective introduced by Eisenhandler and Tzur (2019) in the context of distributing food donations to welfare agencies. As in Eisenhandler and Tzur (2019), we multiply the efficiency and fairness measures and obtain the following expression for the online ride-hailing problem with fairness: $Z = \sum_{i=1}^{M}x_{G_i} - \sum_{i=1}^{M}\sum_{j=i+1}^{M}|q_{G_j}x_{G_i} - q_{G_i}x_{G_j}|$. In the sequel, we aim to find a vehicle assignment policy that maximizes the expected value of Z.

4. Description of Static Versions of the Problem

In this section, we present static formulations of the problem. Unlike the dynamic problem, the static version has complete information in advance regarding the arriving requests. We refer to the realization of requests included in a specific problem as the *set of requests* and denote it with $\mathcal{K}$. Recall that the information regarding each request includes its origin and destination zones and its arrival and expiration times. Knowing the entire future beforehand, the system dispatcher can formulate its problem with mixed-integer linear programming and solve it optimally or heuristically depending on the problem size.

However, in the dynamic version of the problem, the set of requests $\mathcal{K}$ is revealed gradually. Thus, the optimal solution of the static version might not be applicable in the online setting. For example, in the static version, cruising to the origin zone to pick an assigned request can be performed before its arrival time, whereas in the dynamic version, it can only start once the system knows about the request: when it arrives. Considering this as a constraint and similarly other such restrictions, we introduce two MILP formulations with new decision variables and constraints, which define feasible solutions that correspond to applicable online assignments. The difference between the two new mathematical models is the degree of influence that the complete information has on the solution. Specifically, in the first static MILP, requests can be rejected, potentially to serve better future (known in advance) requests. We refer to this problem as *the static problem with elective rejections*. In contrast, in the second static MILP, elective rejections are not allowed. Note that a request is still rejected if no available vehicle can serve it (i.e., $\mathcal{V}_k = \emptyset$ at a_k). This problem is called *the static problem with no elective rejections*.

We based both of our MILP models on a formulation of the offline taxi routing problem suggested by Bertsimas, Jaillet, and Martin (2019). In general, compared with their formulation, our modifications involve restricting the model from using the provided complete information ahead of time and changing the objective function to one that includes fairness.

We use the parameters defined in the previous section and additional parameters that are based on them and are similar to the definitions in Bertsimas, Jaillet, and Martin (2019). They include the set of requests $\mathcal{K}$, which expresses the full information (which is not available in the dynamic problem). Each request $k \in \mathcal{K}$ has a pickup time window (a_k, e_k), and a trip time $T_k = t_{o_k d_k}$. The first set of additional parameters is related to the assignment of the first request on each vehicle. We denote by T_{vk} the travel time between the initial location of vehicle v, z_v, and the origin zone of request k, o_k (i.e., $T_{vk} = t_{z_v o_k}$). These parameters exist for a pair of vehicle v and request k if vehicle v can arrive to pick up request k before its expiration time, assuming that vehicle v is vacant in the first time period. We denote the set of these vehicle-request pairs with $\mathcal{Y} = \{(v,k) | T_{vk} \le e_k\}$ (the number of variables could be reduced by restricting the set $\mathcal{Y}$ to pairs (v,k) that satisfy $a_k + T_{vk} \le e_k$, but the difference in the run time was negligible). The second set of parameters refers to the consecutive assignments of a pair of requests. We denote by $T_{k'k}$ the travel time between requests k' and k, which equals the trip time of request k' and the time to cruise from the destination zone of request k' to the origin zone of request k: that is, $T_{k'k} = T_{k'} + t_{d_{k'} o_k}$. These parameters exist for a pair of requests k' and k if request k arrives after request k' and request k can be served immediately after request k'. The latter requirement means that there is a nonnegative slack between the expiration time e_k and the earliest pickup time of request k when request k is served after request k'. We define these pairs of requests with the set $\mathcal{X} = \{(k', k) | k' < k,\ a_{k'} + T_{k'k} \le e_k\}$.

The solution for each formulation represents a *route* for each vehicle v, which is an ordered sequence of requests that it serves, along with their pickup times. This is described by four sets of decision variables defined similarly to Bertsimas, Jaillet, and Martin (2019). For every $(v,k) \in \mathcal{Y}$, the decision variable y_{vk} equals one if k is the first request assigned to vehicle v (i.e., at the *beginning of the route*) and zero otherwise. For every $(k',k) \in \mathcal{X}$, the decision variable $x_{k'k}$ equals one if request k is served by the same vehicle immediately after request k' (in the sense that no other request is served in between, although idle time may prevail) and zero otherwise. The binary decision variable p_k equals one if request k is served and zero otherwise, and for all k for which $p_k = 1$, the continuous decision variable t_k is the pickup time of request k.

4.1. The Static Problem with Elective Rejections

In our first static formulation, we modified the decision variables and constraints by Bertsimas, Jaillet, and Martin (2019) to exclude two options made available by having complete information. First, we enforce serving requests assigned to the same vehicle in the same order as their arrival by performing the assignment immediately as the system knows about them. Therefore, if requests k' and k are assigned to the same vehicle v and $k' < k$, then request k' is served earlier than request k. This requirement restricts the set of decision variables $x_{k'k}$. Second, we require that a vehicle may start serving request k no earlier than time a_k. Thus, the cruising time from request k' to the subsequent request k cannot be performed before the latter arrives at time a_k. Finally, we changed the objective function to maximizing our objective function Z instead of maximizing the system's revenue as used in Bertsimas, Jaillet, and Martin (2019). We refer to this model as the *static problem with elective rejections* because it enables rejecting requests for which there are available vehicles as opposed to the assumption of our dynamic system, which does not allow for rejecting such requests.

With this setup, we now present the mathematical formulation of our model for the static problem with elective rejection:

$$\max\ Z = \sum_{k\in\mathcal{K}} p_k - \sum_{i=1}^{M}\sum_{j=i+1}^{M}\left| q_{G_j}\left(\sum_{k\in G_i} p_k\right) - q_{G_i}\left(\sum_{k\in G_j} p_k\right)\right| \quad (1)$$

subject to

$$p_k = \sum_{v\in\mathcal{V}} y_{vk} + \sum_{k':(k',k)\in\mathcal{X}} x_{k'k} \qquad \forall k\in\mathcal{K} \quad (2)$$

$$\sum_{k:\,(k',k)\in\mathcal{X}} x_{k'k} \le p_{k'} \qquad \forall k'\in\mathcal{K} \quad (3)$$

$$\sum_{k:\,(v,k)\in\mathcal{Y}} y_{vk} \le 1 \qquad \forall v\in\mathcal{V} \quad (4)$$

$$t_k \le e_k \qquad \forall k\in\mathcal{K} \quad (5)$$

$$t_k \ge a_k + \sum_{k':\,(k',k)\in\mathcal{X}} x_{k'k}(T_{k'k} - T_{k'}) \qquad \forall k\in\mathcal{K} \quad (6)$$

$$t_k \ge a_k + T_{vk} y_{vk} \qquad \forall (v,k)\in\mathcal{Y} \quad (7)$$

$$t_k \ge t_{k'} + T_{k'k} x_{k'k} + (a_k - e_{k'})(1 - x_{k'k}) \qquad \forall (k',k)\in\mathcal{X} \quad (8)$$

$$y_{vk}\in\{0,1\} \qquad \forall (v,k)\in\mathcal{Y} \quad (9)$$

$$x_{k'k}\in\{0,1\} \qquad \forall (k',k)\in\mathcal{X} \quad (10)$$

$$p_k\in\{0,1\} \qquad \forall k\in\mathcal{K} \quad (11)$$

$$t_k \ge 0 \qquad \forall k\in\mathcal{K}. \quad (12)$$

The objective function (1), to be maximized, adequately balances the efficiency and geographical fairness of the system as discussed in Section 3. Constraints (2)–(4) define a sequence of served requests for each vehicle v. Unlike Bertsimas, Jaillet, and Martin (2019), we restrict the sequence to follow the order of the requests' arrival as explained above. Constraint (2) verifies that request k is not served, served first on one of the vehicles, or served after request k' that arrived before it on the same vehicle. Constraint (3) assigns at most one subsequent request (i.e., k) to request k' if the latter is served and none otherwise. Constraint (4) validates that for each vehicle v, at most one request is the first request assigned to it. Constraints (5) and (6) address the pickup time windows. Constraint (5) limits the latest pickup time of request k to its expiration time e_k. Constraint (6) ensures that a vacant vehicle v starts serving request k no earlier than time a_k, which is the arrival time of request k plus the cruising time from the destination of request k' (the previous request), $d_{k'}$, to the origin zone of request k, o_k, which equals $(T_{k'k} - T_{k'})$ by definition. Similarly, Constraint (7) ensures the same but for a vacant vehicle v at the beginning of its route. Constraint (8) schedules the pickup time of the requests in accordance with the vehicle's route. It sets the pickup time of request k that is served after request k' by an occupied vehicle. When $x_{k'k}$ equals one, the pickup time of request k is set to be after vehicle v drops off request k' and the cruising to the origin of request k, which is $t_{k'} + T_{k'k}$ by definition. When $x_{k'k}$ equals zero, the constraint is not active and verifies that a nonnegative number (i.e., $t_k - a_k$) is larger than a nonpositive one (i.e., $t_{k'} - e_{k'}$). Constraints (9)–(12) are integrality and nonnegativity constraints of the decision variables.

The absolute values in the objective function (1) can be linearized, so the static problem with elective rejections formulation is a mixed-integer linear program that can be solved with a standard solver. We define continuous decision variables $W_{G_iG_j}$ to represent the absolute values. The objective function becomes a linear expression, $\sum_{k\in\mathcal{K}} p_k - \sum_{i=1}^{M}\sum_{j=i+1}^{M} W_{G_iG_j}$, and we add the following constraints:

$$W_{G_iG_j} \ge q_{G_j}\left(\sum_{k\in G_i} p_k\right) - q_{G_i}\left(\sum_{k\in G_j} p_k\right) \quad \forall i = 1,\dots,M,\ j = i+1,\dots M \quad (13)$$

$$W_{G_iG_j} \ge q_{G_i}\left(\sum_{k\in G_j} p_k\right) - q_{G_j}\left(\sum_{k\in G_i} p_k\right) \quad \forall i = 1,\dots,M,\ j = i+1,\dots M \quad (14)$$

$$W_{G_iG_j} \ge 0 \quad \forall i = 1,\dots,M,\ j = i+1,\dots M. \quad (15)$$

4.2. The Static Problem with No Elective Rejections

The ability to electively reject requests is another option made possible by having the complete information about the set of requests $\mathcal{K}$. Rejecting a request might be beneficial for serving a known future request, which might provide a better contribution to the objective function. As this option exists in the system analyzed by Bertsimas, Jaillet, and Martin (2019) and our first model, we introduce new decision variables and constraints to dismiss it and add them to the static problem with elective rejections model. These extensions ensure that a solution is only feasible when there is no unserved request k for which there exists an available vehicle to serve it ($\mathcal{V}_k = \emptyset$ at a_k for every unserved request).

Generally, to prevent an elective rejection, this formulation verifies for each unserved request that it cannot be served given the vehicles' routes determined by the solution. Checking this needs the definition of two new sets of constraints. The first set ensures that every vehicle deviates from the expiration time e_k of an unserved request k if it would be assigned to the vehicle's determined route according to the request's arrival time. The second set determines the pickup times of served requests to the earliest time possible. A delayed pickup time of a served request could make the vehicle deliberately miss the expiration time of some subsequent less attractive requests to satisfy the former set of constraints.

We refer to this mathematical model as the *static problem with no elective rejections* and present its complete formulation in Online Appendix A. Note that in our preliminary study, we obtained similar results when using this formulation for guiding the dynamic decisions as obtained with the formulation with elective rejection. In addition, the computational time to solve this formulation was larger because of the additional variables and constraints. For these reasons, we eventually used in our experiments the first formulation of the static problem (i.e., *with* elective rejections) (see Section 6).

5. Online Solution Approach

In this section, we describe our online solution approach based on the solutions for the MILP formulations presented in the previous section. The idea is to extract information from the MILP static solutions, which are solved offline, to design a good real-time dispatching (vehicle assignments) policy for newly arriving requests. The approach suggests a method to transform MILP solutions (decisions) into dynamic policies for the online ride-hailing problem with fairness, but it is a general concept that can be adapted to other problems in which an MILP formulation is available or other online problems whose static solutions can be obtained in a reasonable time.

The method includes two phases. In the first, which is performed offline, we solve the static formulation of the problem for multiple instances and construct a *reference set* based on their solutions. Next, we use a data-driven method that exploits this set to determine vehicle assignments for newly arriving requests in real time. We now describe these two components of our method; a pseudocode appears in Online Appendix B.4.

To build the reference set, we define the *system state* as all of the information necessary to make an optimal decision (a vehicle assignment) each time a new request, say request k, arrives at the system. The system state consists of three elements—the vacancy time and zone of every vehicle $v \in \mathcal{V}$; the attributes of the new request, including the arrival and expiration times and the origin and destination zones; and the current fill rates of the geographical groups, which represent the service history of each group until the arrival time of the new request. Recall that these fill rates are part of our objective function Z.

The dimension of the above state is very large; therefore, we suggest using a reduced state representation, which will include only part of the complete information, ideally the most relevant for the vehicle assignment decision. The reduced state that we use is the following. First, we aggregate the zones into areas. An *area* is a set of several neighboring zones in $\mathcal{Z}$ joined to create a continuous geographical region. That is, the aggregation is a partition of $\mathcal{Z}$ so that every zone $z \in \mathcal{Z}$ belongs to exactly one area. We denote the area of zone z by $A(z)$.

Given this geographical reduction, we represent the system state at time a_k, at which request k arrives, by three new parts. The *geography part* is an integer vector that counts the number of vehicles in each area according to their vacancy zones z_v at time a_k. For the purpose of this count, vacant vehicles or occupied ones that become vacant in the area later on are considered the same. The *time part* is another vector that given the vacancy times τ_v for all $v \in V$ at time a_k, indicates for each area how long it will take until a vehicle becomes vacant there if a vehicle from this area serves the current request. Otherwise (if no vehicle from this area can serve the current request), it is the time duration until the next vehicle becomes vacant. The third part, named the *origin part*, indicates the origin zone o_k of the request that arrived at a_k, which we emphasize because it is the location that the vehicle is dispatched to. We use the request's destination zone, d_k, to correct the geography and time parts of the reduced state to reflect the system after the service of request k. The specific corrections to the state are provided in Online Appendix B.1. We omit from the reduced representation the rest of the elements of the system state, which include the request's expiration time e_k and the current geographical groups' fill rates. We omit the latter as we believe that their influence is insignificant as also shown numerically by Neria and Tzur (2024) for a related system.

Next, we explain how the solutions of the optimized MILPs can be applied in an online setting. This is done by simulating the same problem instances as those solved by the MILPs in order to find the reduced system state at each decision point (i.e., at each request arrival) and match the MILP solution (decision) to each such state. The decision is the area from which the assigned vehicle is dispatched. Note that the simulation keeps track of the zone z_v from which the vehicle is dispatched, and this is "translated" into $A(z_v)$, the corresponding area. Figure 1(a) illustrates a point in time (when request k arrives) with respect to the vehicles' vacancy zones, the request's origin and destination zones, and the vehicle assignment according to the MILP. In Figure 1(b), we show the corresponding item in the reference set containing a reduced system state and its decision (an area from which the vehicle assignment is made).

In Figure 1(a), there are four areas, each containing six zones, and there are six vehicles. Then, request k arrives whose origin and destination zones are within areas 4 and 1, respectively. At the captured point in time, there are no vehicles in area 1, and there are two vehicles, three vehicles, and one vehicle in areas 2, 3, and 4, respectively. However, because the destination of the current request is area 1, we add to it a count of one so that the geography part of the presented item is $(1, 2, 3, 1)$ as observed in Figure 1(b). The time part of this item is calculated as explained above based on data that are not shown in Figure 1(a). The last part of the system state shows the origin zone indicated by the indices of the matrix that represents the city layout (fourth row and fifth column in Figure 1(a)). Based on the MILP solution, vehicle 6 has been assigned to serve this request, resulting in a corresponding decision of area 2 for this item because vehicle 6 is in area 2.

Performing the above-described process to every request arrival in the simulation outlines the construction of the reference set by grouping together the records of all of the MILP solutions into a tabular data set. The reference set can be viewed as a training set in a supervised learning framework composed of input features and a label, which are the reduced system state and the selected area, respectively. See Online Appendix B.2 for further motivation for using the reference set and other aspects of its construction process.

In the online problem, when a new request arrives in the system, we encounter a system state that may not be identical to any of the system states included in the reference set. For this new state, as opposed to the offline situation, we do not have a decision and need to determine the vehicle assignment in real time. To do this, we search for similar reference states encountered by the offline problem and use their corresponding decisions as a basis for the current decision. Besides the fact that we may not find the exact same state in the reference set, it is important to rely on the decision of more than one reference state to avoid dependency with respect to a single problem instance. Thus, we define a *distance function* between the new system state and a reference state; see Online Appendix B.3.1. We look for the N reference states with the smallest distance from the new system state. We refer to these states as the *nearest states* and use all of their corresponding decisions to determine the vehicle assignment by a voting function (see below).

We measure the distance of the new system state only from relevant reference states (i.e., those that recommend using a vehicle from an area that can provide request k with a feasible vehicle assignment). Feasibility is satisfied if there exists at least one vehicle in the set of available vehicles, $\mathcal{V}_k$, that belongs to the area recommended by the decision of the reference state.

The next step is to use a *voting function* to combine the corresponding decisions of the N nearest states. For each of these N states, we know its distance from the new state and the area from which the assigned vehicle was dispatched according to the MILP solution (corresponding decision). The voting function goes through these N states and assigns to their associated areas a value that is a function of their distance from the new state. Finally, the area from which the solution method decides to dispatch the available vehicle is the one that scored the highest according to the voting

Figure 1. (Color online) (a) A Point in Time When Request k Arrives and (b) An Illustration of One Item in the Reference Set

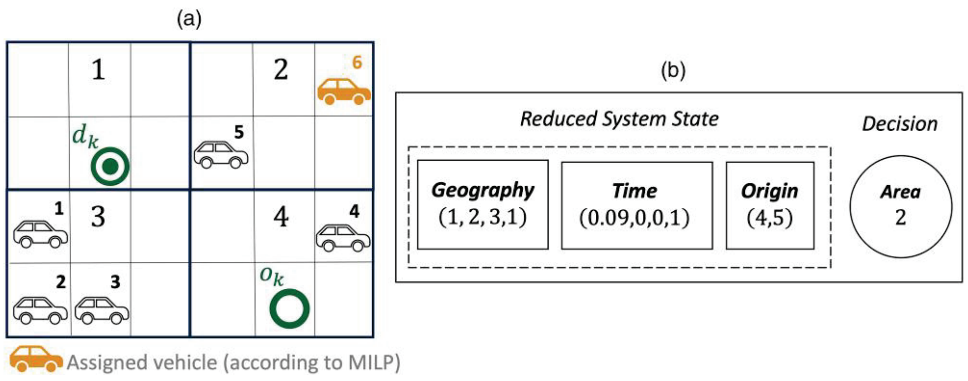
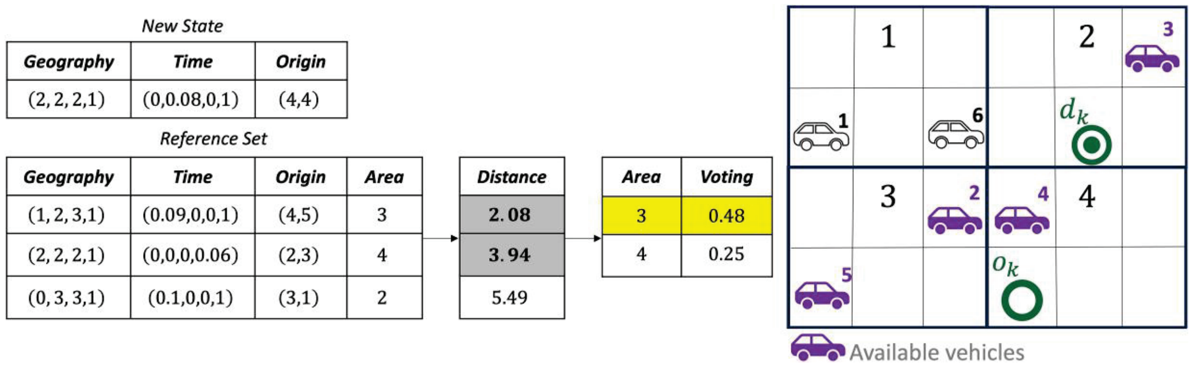


Figure 2. (Color online) Illustration of the Process of Determining the Assigned Vehicle to a New Request (k) in Real Time



New State

Geography	*Time*	*Origin*
(2, 2, 2,1)	(0,0.08,0,1)	(4,4)

Reference Set

Geography	*Time*	*Origin*	*Area*	*Distance*
(1, 2, 3,1)	(0.09,0,0,1)	(4,5)	3	2.08
(2, 2, 2,1)	(0,0,0,0.06)	(2,3)	4	3.94
(0, 3, 3,1)	(0.1,0,0,1)	(3,1)	2	5.49

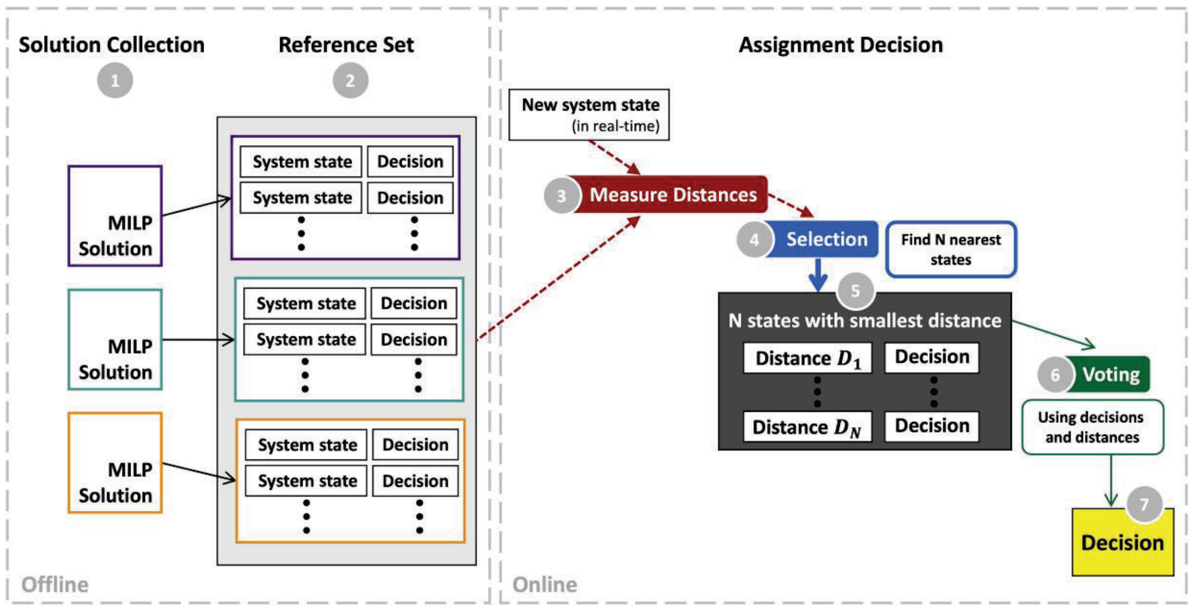
Area	*Voting*
3	0.48
4	0.25

function. See Online Appendix B.3.2 for further details, including the voting functions that we used.

Figure 2 illustrates the process of determining the vehicle to assign to a newly arriving request (request k) in real time. Assuming in the example that there are three items in the reference set, we aim to find the two nearest ones (setting N to two) to the new state. Thus, we calculate the distance of each relevant reference state from the new state (see Online Appendix B.3.1). This calculation reveals that the first two states in the shown reference set are the nearest. Subsequently, we use their distance to apply the voting function (see Online Appendix B.3.2) and determine that for this example, the chosen area is area 3 (the details of the calculations are omitted). Then, we assign vehicle 2 to serve request k because it is in area 3 and closer than vehicle 5 to the origin zone of the new request. Notice that this decision differs from choosing the earliest arriving vehicle, which is vehicle 4.

Figure 3 illustrates the overall solution approach, where the circled numbers indicate the order of the steps in the process. We present the offline phase in the left panel of Figure 3. From a collection of MILP solutions for multiple instances (step 1 of Figure 3), we derive the reference set (step 2 of Figure 3), which includes reduced system states and their corresponding selected areas (decisions, which are represented by the light gray rectangle in Figure 3). Then, as shown in the right panel of Figure 3, which corresponds to the online phase, a new request arrives in real time, for which an assignment decision is needed. Thus, we consider the new system state, compute its distance from all relevant reference states (represented by the dashed arrow into step 3 of Figure 3), and find the N nearest states (in the dark gray rectangle; steps 4 and 5 of Figure 3). Then, the voting function (denoted with the thin solid arrow; step 6 of Figure 3) takes the distances and decisions of the N nearest states to produce

Figure 3. (Color online) Illustration of Our Online Solution Approach to Transform MILP Solutions into Dynamic Policies



the decision (step 7 of Figure 3) (i.e., an area to dispatch the vehicle from).

As indicated above, our policy suggests the area from which to assign the available vehicle. However, there might be several available vehicles in that area. So, to determine the final vehicle assignment (choosing one vehicle among those in the chosen area), we use one of the vehicle-based dispatching rules introduced in Section 6.2.

Our use of the reference set bears some similarities to the concept of *MSA* and *consensus function* (Bent and Van Hentenryck 2004, Voccia, Campbell, and Thomas 2019) (see Section 2). However, in our case, the reference set includes a very large number of states (not necessarily identical), among which we choose only a limited number of closest states on which we base the decision. Then, the decision is obtained by applying weights to the distances between the new state and the states in the reference set. Therefore, our approach can be thought of as a generalization of the MSA. Moreover, the creation of our reference set is done completely offline so that the decisions in real time can be made instantaneously rather than solving deterministic problems of scenarios repeatedly. The creation of the reference set in a way that considers constraints derived from the online setting and its combination with the use of a weighted distance function as described above forms the novelty of our approach.

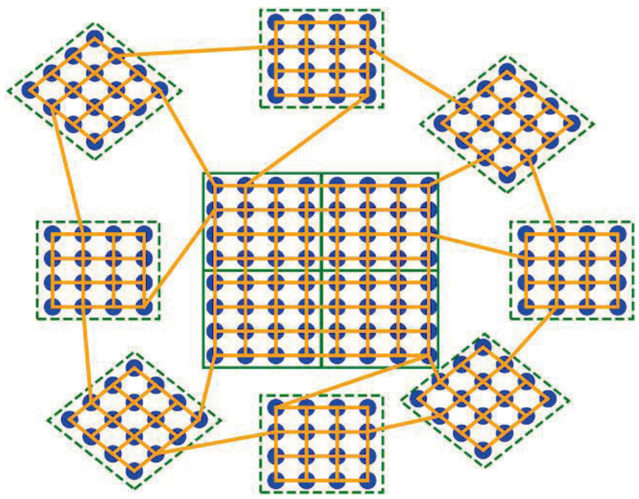
6. Numerical Experiments

In this section, we present the experimental study that we performed to evaluate the performance of our online data-driven policy compared with other dispatching rules. First, we describe how we generated the problem instances by explaining the city network structure, the request arrivals, and the different system settings. Then, we present the other dispatching rules, which were compared with our policy. Finally, we specify the experiment schemes that we performed for every examined system and show their results.

6.1. Problem Instances Generation

We used a synthetic city network composed of 192 zones and 640 bidirectional arcs as Bertsimas, Jaillet, and Martin (2019) suggested for their offline version of the problem. We aggregated the zones into 12 areas. Four represent the city center, and the remaining eight are the city outskirts. In Figure 4, each dot is a zone, and each line is a bidirectional arc. Each area contains 16 zones and is marked with a rectangle surrounding its zones (solid lines indicate city center areas and dashed lines indicates city outskirt areas in Figure 4). The travel times are slower inside the areas and faster between areas and on the perimeter of the city center.

Figure 4. (Color online) The Layout of the Asymmetric City Network



The average travel time between all pairs of zones equals 7.6 minutes, and the maximum is 18 minutes. Although all networks that we considered have the same general structure, in Figure 4, the distances between the outskirts and the city center areas are not identical; therefore, it represents an *asymmetric* system.

We modeled the arrivals of ride requests as a Bernoulli process, where at each time period t, a request arrives with probability p (we sometimes refer to time period t as time t). The probability was set to $p = 0.025$. The arrival time of request k, a_k, is the time when the kth request arrives to the system; recall that if $k_1 < k_2$, then $a_{k_1} < a_{k_2}$. The origin and destination zones of each request are o and d, respectively, with probability p_{od} (i.e., independent of the time that it arrived). These probabilities are chosen so that most requests are from/to the city center areas. Specifically, the probability equals 0.25 to each of the following pairs: requests between zones within the city center areas, from a zone in a city center area to a zone in a city outskirts area (and vice versa), and between zones within city outskirts areas. All of the above percentages were uniformly divided between the different zone pairs. Because there are twice as many zones in the city outskirts compared with the center, there are twice as many requests to/from each zone in the city center compared with the outskirts. The duration between the arrival time and the expiration time ($e_k - a_k$) was set to six minutes for every request k.

To build the reference set from the MILP solutions, we selected the number of time periods T to be 2,400, representing 40 minutes when each time period is one second. Thus, on average, 60 requests arrive in 2,400 seconds when $p = 0.025$. The number of vehicles $|\mathcal{V}|$ was set to 15. The initial zone z of each vehicle was randomly generated according to the probability that a request will originate from zone z (factoring the distribution of p_{od} over all of the possible destinations, $\sum_{d\in Z} p_{zd}$). In the dynamic settings, the same parameters were used, and it was run for 60 minutes.

The above parameter values were chosen to create a system where the demand for requests exceeds the supply of vehicles (reflected by the incapability to serve all of the requests, even according to the MILP solutions) so that the consideration of fairness is meaningful (otherwise, the system would manage to serve all of the requests). In addition, the size of the instances was limited according to the ability to obtain an optimal solution for all of the MILP formulations that we used.

The above values of the parameters characterize the basic dynamic system that we investigated. However, we also examined our approach with other systems, varying the network structure, the arrival probability of request arrivals, the travel time between zones, and the expiration time of the requests (as detailed in Section 6.4).

6.2. Other Dispatching Rules

We evaluate the policies generated by our approach against several dispatching rules. These dispatching rules follow the same characteristics as the policies developed by our approach concerning the framework of the online problem presented in Section 3. Specifically, each dispatching rule should produce a fast (immediate) decision without any knowledge regarding future demands, namely when requests are coming next and from which origin zone to which destination zone.

Generally, a dispatching rule determines at time a_k of the vehicle assignment of request k. However, the rules can be classified into two types regarding the decision that they produce. *Vehicle-based criteria* select a vehicle directly, considering the available vehicles' vacancy times and zones. These rules are relatively common in the literature and in practice for ride-hailing services (see, for example, Maciejewski, Bischoff, and Nagel 2016 and Wang and Yang 2019, respectively). The second type, *area-based criteria*, has more sophisticated rules, which consider the current geographical distribution of the fleet between the areas and aim to balance them (Lin et al. 2018). These rules recommend an area to dispatch the vehicle from, and the specific vehicle is chosen from the available vehicles in that area according to additional considerations. The policies generated by our approach can be viewed as area-based criteria.

6.2.1. Vehicle-Based Criteria.

These rules determine which available vehicle $v \in \mathcal{V}_k$ to assign to request k. The *earliest arriving vehicle*, denoted *EA*, chooses the vehicle v^* that arrives at the origin zone of request k, o_k, the earliest among all available vehicles in $\mathcal{V}_k$. That is, $v^* = \arg\min_{v\in\mathcal{V}_k}(\tau_v + t_{z_v o_k})$, where τ_v is the vacancy time of vehicle v and $t_{z_v o_k}$ is the cruising (travel) time from its vacancy zone to the origin of request k. The motivation for this rule is to minimize the waiting time of requests in a greedy manner. We use this rule as our baseline, comparing the performance of our policies and dispatching rules relative to it.

Motivated by greedily minimizing the empty travel time of vehicles, another common rule is the *closest vehicle*, denoted *CV*, which selects the vehicle v that has the shortest cruising time to the request's origin zone o_k (i.e., $v^* = \arg\min_{v\in\mathcal{V}_k} t_{z_v o_k}$). The last rule is *random*, denoted *R*, which randomly chooses a vehicle v from the set of available vehicles $\mathcal{V}_k$.

6.2.2. Area-Based Criteria.

The rules in the area-based criteria determine from which area an available vehicle $v \in \mathcal{V}_k$ should be assigned to serve request k. This requires mapping the set of available vehicles into areas. For that purpose, we use the aggregation of the zones into areas as introduced in Section 5. We define $\mathcal{A}_k = \{A(z_v) \mid v \in \mathcal{V}_k\}$ as the *set of available areas*, where z_v is the vacancy zone of vehicle v, $A(z)$ is the area of zone z, and $\mathcal{V}_k$ is the set of available vehicles to serve request k. Thus, $\mathcal{A}_k$ is the set of areas that contain at least one available vehicle $v \in \mathcal{V}_k$, and the rules need to select an area $A \in \mathcal{A}_k$. If there is more than one vehicle in the chosen area A^*, the specific vehicle can be selected with a vehicle-based criterion.

The first rule that we consider in this category is *most crowded*, denoted *MC*, which chooses the area with the largest number of vehicles, relying on the assumption that if there are already many vehicles in that area, the assigned vehicle might be unnecessary for near-future requests. Note, however, that when counting the number of vehicles in an area, it may be desirable to consider not only vehicles that are vacant there at time t but also, vehicles that are known to arrive there in the future. Thus, we denote the number of vehicles at area A at time t by N_{tA}, and we define $N_{tA} \equiv |\{v \in \mathcal{V} \mid \tau_v \geq t,\ A(z_v) = A\}|$. Other definitions of N_{tA} may be used: for example, $\tau_v = t$. Then, the rule is $A^* = \arg\max_{A\in\mathcal{A}_k} N_{a_k A}$.

The second rule is *most crowded normalized*, denoted *MC-N* (a variation of MC), which chooses the area with the highest supply-demand ratio (rather than considering only the supply). We denote by μ_A the expected number of requests per time period originating at area A: that is, $\mu_A = \sum_{o\in Z:A(o)=A}\sum_{d\in Z} p_{od}$. Then, the rule is $A^* = \arg\max_{A\in\mathcal{A}_k} N_{a_k A}/\mu_A$. Both MC and MC-N greedily account for the passengers' geographical fairness objective and attempt to keep vehicles in areas with insufficient supply for near-future demand. Indeed, these rules are particularly useful for serving requests from outskirt areas because such requests typically can be served by vehicles from a limited number of areas, so it is essential not to leave any area without any vehicle. This is in contrast to requests originating from city center areas that typically can be served by vehicles from more than one area. Therefore, MC and MC-N are both rules that promote fairness. Similar concepts were suggested by Fagnant and Kockelman (2014) and Ackermann and Rieck (2023) as a repositioning strategy of idle vehicles for a shared autonomous vehicle service and a ride-hailing system, respectively. However, to the best of our knowledge, this is the first time that it has been used for the decision of vehicle assignments.

6.3. Experiment Steps for Each System

Our numerical study included an examination of several systems that differ from each other in the model assumptions and/or parameters as described in Section 3. The numerical study for each system included two parts. In the first part, performed offline, we solved the MILP formulations and created the online policy based on their solutions. In the second part, we ran a simulation in which we implemented the online policy in real time on new problem instances.

In the first part, we solved the MILP formulations for randomly chosen problem instances. We used a standard commercial optimization engine (IBM ILOG CPLEX 20.1.0.0 solver) and limited the running time to 12 hours for each instance. Then, we used the MILP solutions to build the reference set for our online policy as described in Section 5. Finally, we chose the values of the parameters that define the specific policy: for example, the weights used to calculate the distance between two states. We examined approximately 1,000 parameter combinations, and for each combination, we simulated 12,000 problem instances of the system. We then picked the parameter combination that yielded the highest mean increase in the objective function Z normalized by the value of EA, the baseline rule.

In the second part, we simulated the online policy on 12,000 new problem instances with a 30-, 60-, 90-, or 120-minute duration using Python 3.7. For one of the systems, we also evaluated the performance of the online policy on longer instances of 10 hours of duration (see Section 6.4.4). In all cases, the same instances were also solved with the dispatching rules described in Section 6.2. We set for all rules a system warm-up duration of 600 seconds at the start of the simulation, during which EA determined the vehicle assignments, and the requests that arrived during that time were omitted from the results. In Section 6.4, we present the results of the instances with a duration of 60 minutes, which were similar to those for the other durations.

6.4. Results

In this section, we present the results of our experiments for seven system settings. System 1 is the basic (asymmetric) system with the parameters mentioned in Section 6.1. For this system, we also consider alternative objective functions, which put an even higher emphasis on fairness relative to efficiency, and we analyze the difference in the results. In System 2, the travel time from city center zones to outskirt zones is doubled compared with the basic system, and the expiration time of requests originating from outskirt zones is delayed. System 3 and System 4 are identical to System 1 except that the value of p is increased to 0.028 or the time duration of the instances is increased to 10 hours, respectively. System 5 has a different city network layout where all of the outskirt areas surround the city center areas symmetrically. In System 6, we examine the robustness of our approach by using different parameters or a different network layout in the offline and online phases. Finally, in System 7, we modify the arrival characteristics of requests so that it is more difficult to fulfill them. Table 1 outlines the differences between the system settings.

In Sections 6.4.1–6.4.4, 6.4.6, and 6.4.7 (Section 6.4.5) below, when we indicate that the results are statistically significant, it is based on a paired sample t test with an upper-tailed (lower-tailed) alternative hypothesis for which the p-value was no higher than 0.01. That is, with a probability of at least 0.99, we can reject the null hypothesis that the two mean values considered are equal, and therefore, the one with the higher (lower) average is significantly better. We used paired samples because in the comparison of two rules, the same instance was always matched. As will be observed in the next sections, the statistical tests were significant even for a small-scale difference in the mean values because we tested 12,000 instances.

6.4.1. System 1: The Basic System. In this system, we used the parameters mentioned in Section 6.1. For the definition of the geographical groups in the objective function, we account for the area of the origin zone of the requests. That is, if $k', k \in G_i$, then only their origin zones, $o_{k'}$ and o_k, are necessarily in the same area (i.e., $A(o_{k'}) = A(o_k)$) and their destination zones, $d_{k'}$ and d_k, can be in different areas. We choose this definition as opposed to others because it creates a smaller number of groups, where each geographical group is bigger; so, there are fewer groups that have a single request. Moreover, this definition promotes equal service to requests starting from different parts of the service region.

With the two MILP formulations, shown in Section 4, we can create several online policies; see Online Appendix A. In our preliminary runs, we used the two MILP formulations and examined the performance of the

Table 1. Summary of the Differences Between the Examined Systems

System no.	Network structure/other	Travel times between zones	Expiration times of requests ($e_k - a_k$)	p-value	Duration (hours)
1 (Basic)	Asymmetric	Closer outskirts	Constant (6 minutes for all k)	0.025	1
2	Asymmetric	Farther outskirts	Extended (8 minutes) if o_k is in the city outskirts	0.025	1
3	Asymmetric	Closer outskirts	Constant (6 minutes for all k)	0.028	1
4	Asymmetric	Closer outskirts	Constant (6 minutes for all k)	0.025	10
5	Symmetric	Farther outskirts	Extended (8 minutes) if o_k is in the city outskirts	0.025	1
6	Asymmetric/different offline and online phases	Closer outskirts	Constant (6 minutes for all k)	0.025	1
7	Asymmetric/harder to fulfill	Closer outskirts	Constant (6 minutes for all k)	0.025	1

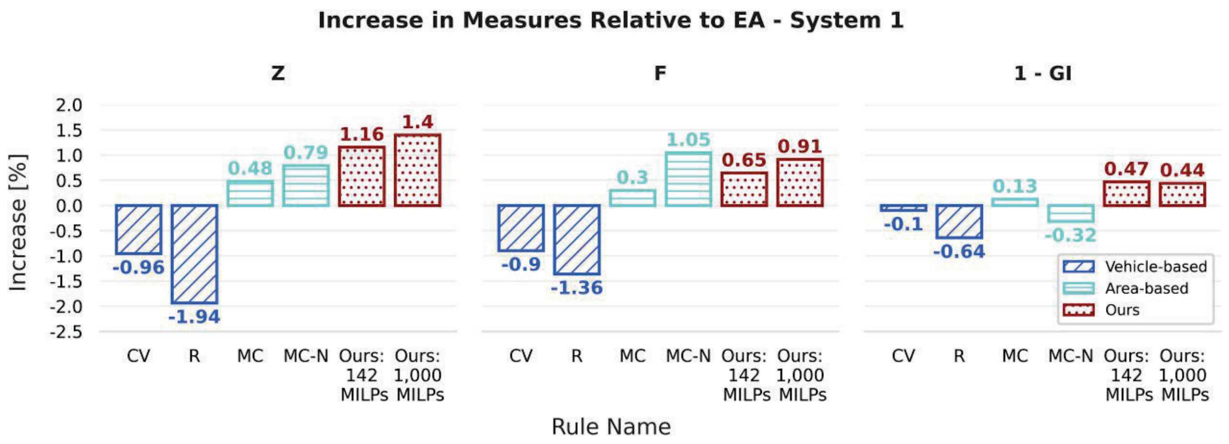
generated policies. The results showed that the MILP formulation with elective rejections had a significantly lower computation time (five seconds on average) than the MILP formulation with no elective rejections (eight hours on average). However, the performance of the two policies, each constructed upon the solutions of a different formulation, was relatively close. Thus, in our experiments, we only run the MILP formulation with elective rejections and build our reference set upon its solutions. This enables us to base the reference set on more MILP solutions (see the effect of the reference set later).

We ran a parameter search to find the best combination for our online policy. For MC and MC-N, we used the variant where N_{tA} takes into consideration vehicles at or on their way to the area, and EA selects the specific vehicle after the area selection because these settings use more information and performed best in preliminary runs. Figure 5 depicts the average increases (in percentage) of our policy and the other dispatching rules relative to EA for the measures Z, F, and $1 - GI$. That is, each measure of each instance according to each rule was normalized by the value achieved by EA for the same instance. Table 2 presents the mean values of each measure over all instances according to each of the rules. For the fairness measure, we multiply $1 - GI$ by 100 to obtain values on a 0–100 scale. In Tables 2–6, for each measure, we highlight in bold the highest value. Recall that our objective is a maximization function; therefore, high positive values are desirable.

From Figure 5, we can see that our policy outperformed all of the dispatching rules with respect to the Z and $1 - GI$ values, both relative to the baseline and with respect to the average value. The average Z value of our online policy was significantly larger than that of MC-N, which was the best dispatching rule. This is achieved because of an improvement in the fairness measure $1 - GI$, as expected from the MILPs objectives, on the account of a slight deterioration in the efficiency measure, F.

To evaluate the effect of the size of the reference set, we repeated the steps of the experiment for System 1

Figure 5. (Color online) Increases (in Percentage) in the Z Value (Left Panel), the Efficiency Measure F (Center Panel), and the Fairness Measure $1 - GI$ (Right Panel) of Each Rule Normalized by EA and Averaged over 12,000 Instances of 60 Minutes in System 1



Notes. Diagonal line bars indicate the vehicle-based dispatching rules, horizontal line bars indicate the area-based dispatching rules, and dotted bars indicate our policies. For clarification, in all subsequent experiments (except System 4), the figures show the average of 12,000 test instances of 60 minutes and differentiate between the types of dispatching rules and our online policy with the same patterns as in Figure 5.

Table 2. Mean Measures of Each Rule Averaged over 12,000 Instances of 60 Minutes in System 1

	EA	CV	R	MC	MC-N	Ours: 142 MILPs	Ours: 1,000 MILPs
Z	60.53	59.89	59.21	60.64	60.82	**61.11**	**61.25**
F	69.50	68.84	68.46	69.60	**70.11**	69.88	70.06
$(1-GI)\cdot 100$	87.05	86.94	86.45	87.11	86.73	**87.42**	**87.39**

Note. Bold indicates the highest value.

(except for the search of the parameter combination), now solving 1,000 offline problem instances with the formulation with elective rejections. Recall that the time required to solve one instance of this MILP formulation is smaller. In Figure 6, we measure the size of the reference set by the number of MILP solutions that it is constructed from (the x axis) and demonstrate its effect on the performance (the Z value on the y axis) of our online policy. For every reference set size specified in Figure 6, the simulation that applies our policy was run, and the average Z value of 4,000 instances is reported.

Figure 6 demonstrates that generally, as the reference size increases, the performance of the online policy improves. Another important observation obtained from Figure 6 is that relying on a reference set to create a vehicle assignment rule is indeed beneficial, which provides a justification for our approach. We solved the same online instances with the online policy with the reference set based on 1,000 MILP solutions (see the rightmost values in Figure 5 and Table 2). The improvement from the additional solutions in the reference set is shown by comparing the performance of the two online policies. Compared with the performance of the policy with the smaller reference set, the Z value is significantly greater and equals 61.25. Examining the measures composing it, F has increased to 70.06, and $(1-GI)\cdot 100$ has a slight decrease to 87.39. A similar change is observed in the ratio measures relative to EA. Thus, in the next experiments, we solve and construct the reference set based on 1,000 solutions. We report only the measures of our 1,000 MILPs policy as the results for the rest of the rules are not affected by the reference set size.

In the above experiment, one can see that our policy gained around a 1% improvement of the objective function Z, which balances the efficiency and fairness measures. However, this objective function $Z = F(1-GI)$ considers fairness in a relatively moderate way. When a higher emphasis on fairness is desired, one may want to consider an objective, such as $Z2 = F(1-2GI) = F-2FGI$ or even $Z3 = F(1-3GI) = F-3FGI$. Indeed, when calculating $Z2$ and $Z3$ based on the assignments made by our online policy for the basic system, we obtained improvements over EA (whose performance was also calculated according to $Z2$ and $Z3$) of 2.19% and 3.69%, respectively. This demonstrates that our method provides an even more significant improvement compared with EA when a larger emphasis on fairness is expressed in the overall objective function because EA does not consider fairness at all.

6.4.2. System 2: Farther Outskirts with Longer Expiration Time.

Compared with System 1, we made two changes in System 2. First, we doubled the travel times from the zones in the perimeter of the city center areas to the zones in the city outskirts areas (and vice versa). The average travel time between all pairs of zones was 9.4 minutes, and the maximum was 23 minutes. This change yields a more restrictive system than System 1 because the number of vehicles remained 15, but it took more time to travel to the outskirts areas. Contrary to that, we defined the expiration time e_k of request k depending on the area of its origin zone $A(o_k)$. If request k originates from an area in the city outskirts (city center), the duration from the arrival time a_k to the expiration time e_k is set to eight minutes (six minutes). The delayed expiration time allows for additional time to pick up passengers in distant locations, which acts as opposed to the more restrictive travel time. We repeated the experiment steps; Figure 7 presents the increases (in percentage) of our online policy and dispatching rules relative to EA, and Table 3 summarizes the mean measures in System 2.

With longer travel time and delay in expiration time of the distant originating requests, all average

Table 3. Mean Measures of Each Rule Averaged over 12,000 Instances of 60 Minutes in System 2

	EA	CV	R	MC	MC-N	Ours
Z	53.26	52.78	51.56	52.66	53.65	**53.80**
F	62.18	61.76	60.62	61.56	62.50	**62.60**
$(1-GI)\cdot 100$	85.51	85.30	84.91	85.43	85.74	**85.81**

Note. Bold indicates the highest value.

Table 4. Mean Measures of Each Rule and Each System Averaged over 12,000 Instances of 60 Minutes in System 3

	EA	CV	R	MC	MC-N	Ours
Z	64.25	63.58	62.58	64.00	64.14	**64.77**
F	74.42	73.66	73.07	74.21	74.78	**74.83**
$(1-GI)\cdot 100$	86.29	86.26	85.61	86.22	85.76	**86.54**

Note. Bold indicates the highest value.

Table 5. Mean Measures of Each Rule and Each System Averaged over 2,000 Instances of 10 Hours in System 4

	EA	CV	R	MC	MC-N	Ours
Z	637.90	632.34	623.34	637.90	634.55	**644.26**
F	689.24	680.13	679.25	691.97	**697.74**	696.21
$(1-GI)\cdot 100$	92.55	**92.98**	91.77	92.19	90.95	92.54

Note. Bold indicates the highest value.

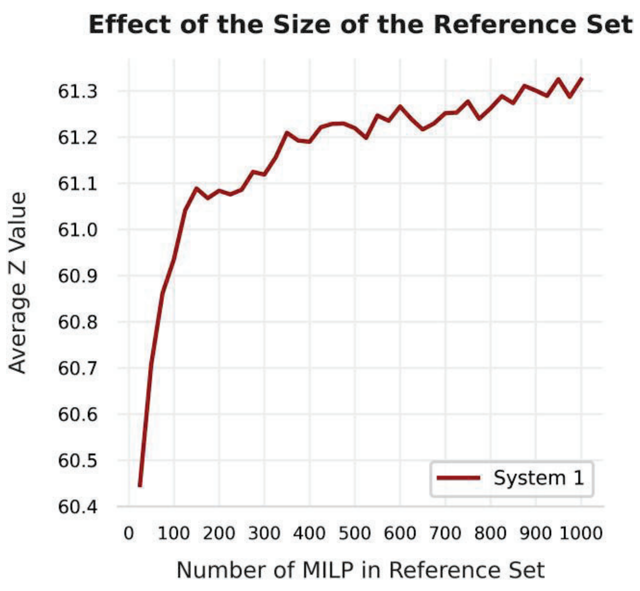
measures by all rules have decreased compared with the performance in System 1; see Table 3. However, when comparing their performance, our online policy outperformed all of the other dispatching rules with respect to all of the measures, both relative to the baseline and in the average values. Still, the average Z value of our online policy was significantly larger than that of the best dispatching rule, which is MC-N.

Regarding the measures constructing the objective Z, our online policy improved both, including the efficiency measure F, for which previously, the MC-N achieved the best result. In the geographical fairness measure $1-GI$, the fill rates of requests originating in the outskirts areas have approached the ones originating in the center areas (that suffered a decrease). This result highlights that more travel time should be allocated to pick up passengers from areas in the city outskirts for managing a ride-hailing service in a geographically fair manner.

However, we observe that the MC rule had deteriorated and failed to benefit from the later expiration times in the existence of longer travel time. The mean measures of MC achieved were lower than even two of the vehicle-based dispatching rules (EA and CV) in the mean value of Z and F. MC dispatches a vehicle from an area with the maximum number of vehicles without considering their need for any future demand. Thus, unlike our online policy and MC-N, MC lacks a component of looking ahead to the near future, which results in performance deterioration. This is a general limitation of MC, which is magnified when the expiration times are longer and a longer future is considered.

6.4.3. System 3: Impact of a Busier System. In this section, we aim to examine the performance of our online policy in larger instances of the problem. Therefore, we increased the arrival probability of the requests to $p = 0.028$ in System 1, which resulted in more frequent arrivals of requests (every 36 seconds on average instead

Figure 6. (Color online) Average Z Value over 4,000 Problem Instances of 60 Minutes in System 1



of every 40 seconds when $p = 0.025$). We repeated the experiment steps but used the same parameter combination for the creation of the online policy. We show the results of System 3 relative to EA in Figure 8, and the mean values are presented in Table 4.

Similar to the less busy system (i.e., System 1) (see Figure 5 and Table 2 in Section 6.4.1), our online policy outperformed all of the other dispatching rules and performed the best with respect to all of the measures relative to the baseline and in the mean values. The average Z value of our online policy was significantly larger than that of the best dispatching rule, which is EA for System 3. According to the results, MC-N, which previously was the best dispatching rule, was deteriorating when implemented on a busier system, whereas the performance of the rest of the dispatching rules remained in a similar trend. It is observed with the low increase in the Z value relative to EA in Figure 8 and with the mean Z value by MC-N in Table 4, which was significantly lower even than the mean by EA.

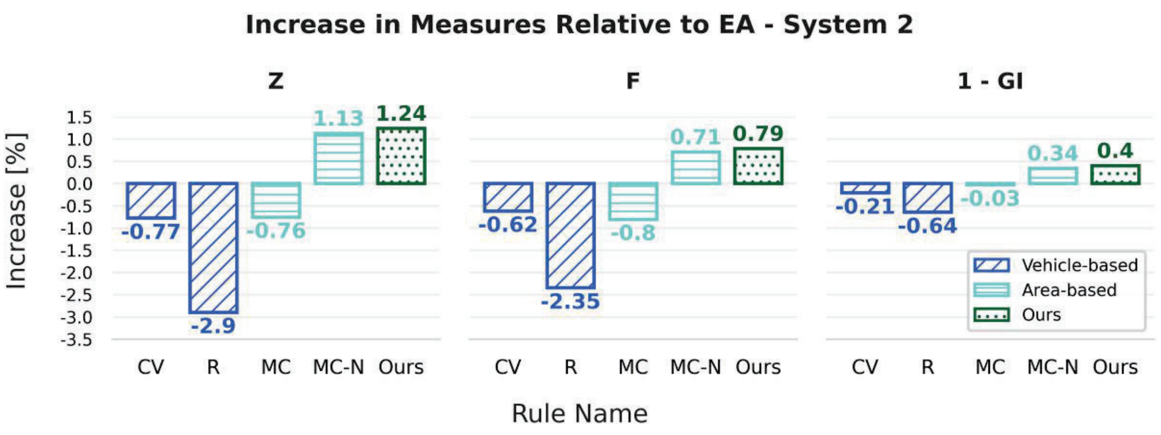
6.4.4. System 4: Impact of Longer Instances. In this section, we aim to examine the performance of our online policy in longer instances of the problem and refer to it as System 4. Therefore, we used the settings of System 1 but increased the duration of the instances

Table 6. Mean Measures of Each Rule Averaged over 12,000 Instances of 60 Minutes in System 4

	EA	CV	R	MC	MC-N	Ours
Waiting time (seconds)	**244.3**	256.1	287.6	278.8	277.1	252.5
Rejection rate (%)	29.88	30.34	31.58	30.52	**29.34**	29.35

Note. Bold indicates the highest value.

Figure 7. (Color online) Increases (in Percentage) in the Z (Left Panel), F (Center Panel), and $1 - GI$ (Right Panel) Measures Normalized by EA in System 2



to 10 hours. We repeated the experiment steps except that we used the same static MILP solutions to create the reference set and the same parameter combination for the creation of the online policy. We show the results of System 4 relative to EA in Figure 9, and the mean values are presented in Table 5.

The results obtained for System 4 show that the objective function Z of our online policy outperformed all of the other dispatching rules (see the left panel of Figure 9 and the first line of Table 5). In particular, our policy gained a significantly better mean Z value compared with the best benchmark, which was MC. Other dispatching rules gained the best values for the F and $1 - GI$ values, however, in low margins compared with the values by ours.

6.4.5. System 5: The Symmetric City Network. In this section, we aim to examine the effect of geography symmetry on the performance of our online policy. As mentioned in Section 6.1, the city network graph of the systems examined thus far had a geographical asymmetry in the location of areas around the city center such that some outskirts areas were closer to the center, whereas others were farther. In System 5, we changed this assumption and used a city network in which the distances of all of the outskirt areas surrounding the city center areas are symmetric (see Figure 10 for a comparison between the two networks). In the new network, the travel times on the arcs connecting an outskirt area and a city center area are all equal, and the travel times of all arcs connecting a pair of adjacent outskirt areas are equal. Compared with the average travel times of the same arcs in the previous network, the average travel time between two outskirt areas has increased, and it has decreased between city center and outskirts areas.

The results for System 5 demonstrate that there is no significant difference between the performance of our online policy (that achieved an average Z value of 54.75) and MC-N, which was the best benchmark (that achieved an average Z value of 54.69). When analyzing the efficiency and fairness measures, we see that our

Figure 8. (Color online) Increases (in Percentage) in the Z (Left Panel), F (Center Panel), and $1 - GI$ (Right Panel) Measures Normalized by EA in System 3

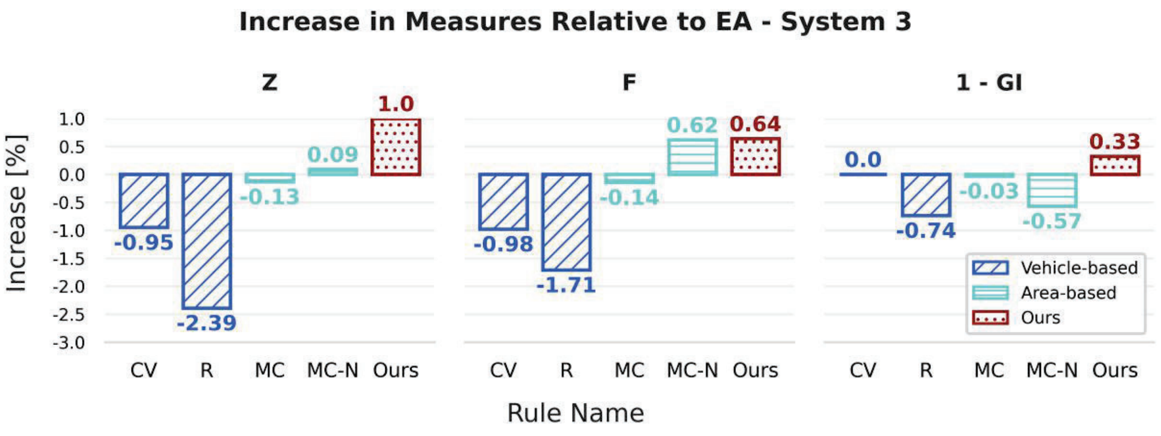
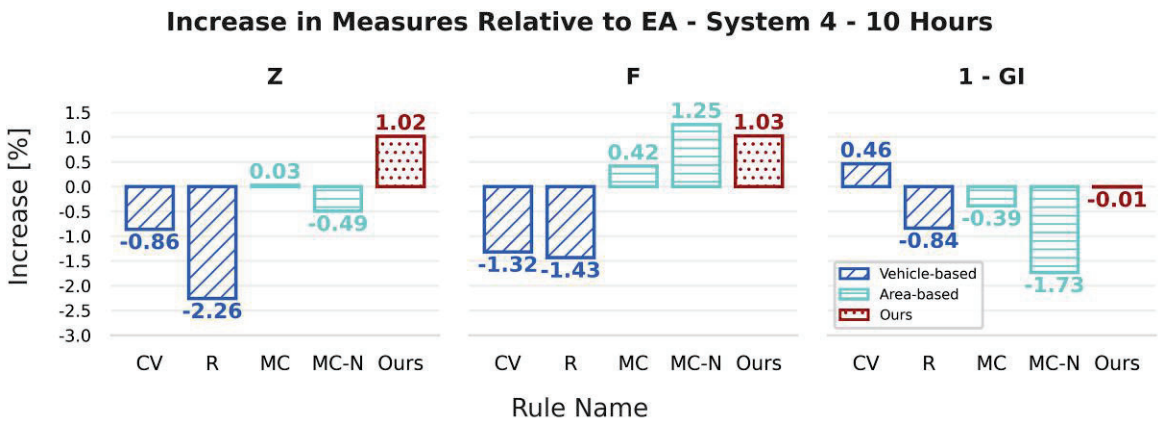


Figure 9. (Color online) Increases (in Percentage) in the Z (Left Panel), F (Center Panel), and $1 - GI$ (Right Panel) Measures Normalized by EA Averaged over 2,000 Instances of 10 Hours in System 4



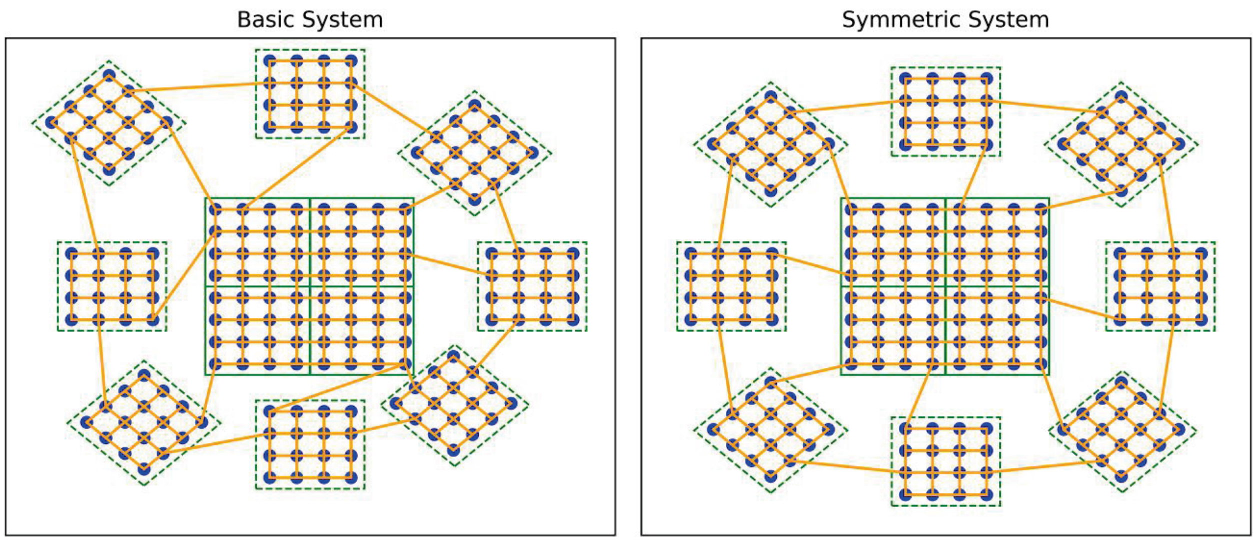
online policy performed significantly better than MC-N in the average F measure, whereas there was no significant difference between the mean values of $1 - GI$ obtained by the two. It appears that it is easier to balance the service between the areas in the symmetric city network so that MC-N facilitates equal service, but the improvement in the fairness measure comes at the expense of efficiency.

Thus, for this system, we tested another objective function to verify that our approach, which emphasizes the objective function of the MILP formulation that they are based on, is beneficial to another function, different from the Z function that was used earlier. For that purpose, considering the problem's parameters defined in Section 3, we changed the objective function to minimize the waiting time of requests, which is another common objective for ride-hailing services (Lowalekar, Varakantham, and Jaillet 2018, Feng, Kong, and Wang 2021). The new objective function is $\sum_k (t_k - a_k)$, which is the difference between the pickup time t_k and the arrival time a_k of served requests k.

We modified the formulation with elective rejections for this experiment. First, we defined the formulation as a minimization problem. We replaced the objective function with $\sum_{k\in\mathcal{K}}(t_k - a_k) + \sum_{k\in\mathcal{K}} T(1 - p_k)$. This function considers the waiting time of served requests, where the waiting time of request k is $(t_k - a_k)$, and it penalizes with T any unserved request k, for which $p_k = 0$. The function operates hierarchically, first minimizing the number of unserved requests and then minimizing the waiting time of the served ones. Notice that this objective determines the pickup time of unserved request k to a_k. Therefore, the first expression represents exactly the waiting time of served requests, leading to the set of constraints remaining the same as presented in Section 4.

We first repeated the experiment steps based on this modified formulation while using the exact same

Figure 10. (Color online) Comparison of City Networks

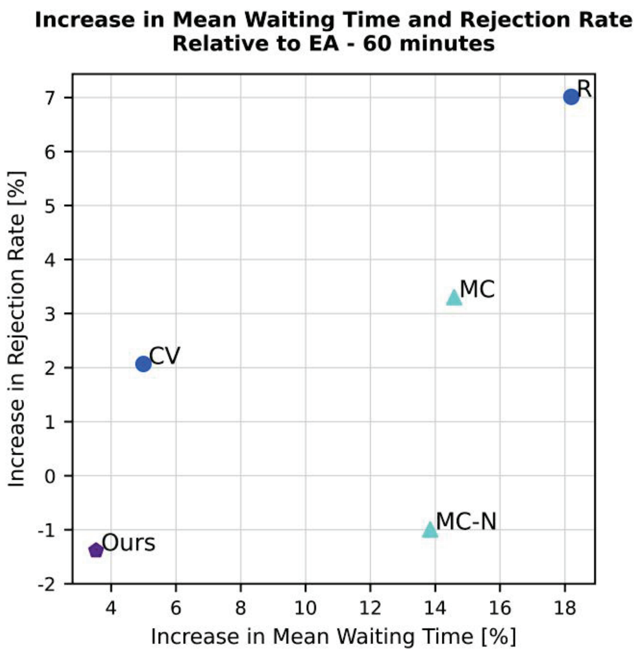


parameters of the online policy that are aimed to maximize the objective function Z. Then, for each instance, we measured the waiting time of requests and the rejection rate by each rule, and we normalized them by the value achieved by EA for the same instance. Figure 11 shows these results jointly (the waiting time of requests on the x axis and the rejection rate on the y axis) as there is no function balancing their values. This is a minimization problem, so the performance of the rule is better if it is located toward the bottom left of Figure 11. Table 6 shows the mean for the two measures of each rule, where lower values are better.

Among the rules, only our policy and MC-N decreased the average rejection rate relative to EA by 1.38% and 1.00%, respectively. There was no significant difference between the mean rejection rate of our policy and MC-N. However, the increase in the average waiting time of requests compared with EA was the smallest with our online policy (3.54%). In accordance, the average waiting time of requests achieved by our policy was significantly lower than the average waiting time gained by MC-N (see Table 6).

Next, we aim to verify that the objective function of the MILP formulation contributes to the measures achieved by the online policy. Thus, we used the online policy with the reference set built upon the solutions of the objective Z maximization and measured its performance with respect to the requests' waiting time and

Figure 11. (Color online) Increases (in Percentage) in the Mean Waiting Times of Requests (x Axis) and the Mean Rejection Rates (y Axis) Normalized by EA in System 4



Note. Circles indicate the vehicle-based dispatching rules, triangles indicate the area-based dispatching rules, and the pentagon indicates our policy.

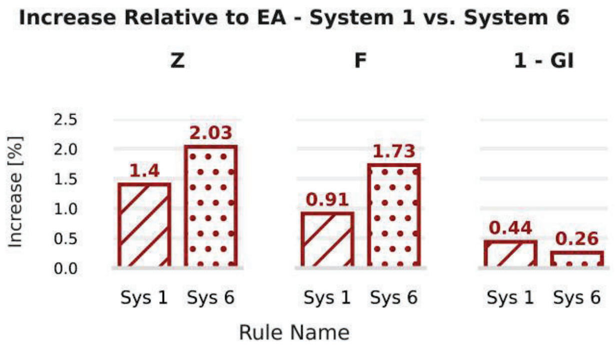
the rejection rate measures. As expected, this policy achieved a significantly higher value for the waiting time of the requests compared with the one gained by the policy with the reference set constructed upon the MILP formulation of the waiting time minimization. Note that the opposite case showed the same results; namely, we used the solutions of the waiting time minimization as the reference set basis, which gained a lower average of the objective Z and the $1-GI$ measures. These outcomes confirm that the objective function of the MILP formulation is influential to the performance of the online policy in real time.

6.4.6. System 6: Robustness. In the previous experiments, the same system settings were used to create the problem instances for building the reference set and for evaluating the policy in real time. To test the robustness of our solution approach, we used a different system setting to build the reference set than the one used in real time. Such a robustness can be beneficial for industry practitioners who might experience changes between the historic data that they used to build the reference set and the real-time scenarios occurring when operating the system. We focused on two parameters of the system settings—the probabilities p_{od} and the city network layout.

First, we examined the case where there is a difference in the probability distribution of the four pairs of requests (from an area in the center/outskirts to an area in the center/outskirts). Recall that these distributions are later uniformly divided between all of the corresponding origin-destination zones pairs to determine the probabilities p_{od}. Specifically, for the reference set, the probability for requests from a zone in a city center area to a zone in a city outskirt area was 0.55, and it was 0.15 for all other pairs of areas. For the online simulation, we used the same online instances as in System 1, where the probability for all of the pairs was 0.25 and the values of all of the rest of the parameters were also the same as in System 1. The results of this setting appear in Figure 12, where the percentages of improvements in Z, F, and $1-GI$ over EA appear in the left panel for System 1 (for ease of comparison) and System 6 (the current system), both when implementing our method, and the corresponding absolute values appear in the right panel.

Compared with the performance of our policy in the basic system (System 1), one can see that not only the performance of our policy did not deteriorate when we used a reference set with unmatched settings (System 6), but it actually improved from 1.4% to 2.03% relative to EA for the Z value. The improvement relative to EA increased from 0.91% to 1.73% for the F value and slightly deteriorated from 0.44% to 0.26% for the $1-GI$ value. These trends are also clearly reflected in the absolute values of the two systems

Figure 12. (Color online) Comparison Between Systems 1 and 6



	System 1	System 6
Z	61.25	**61.62**
F	70.06	**70.61**
$(1 - GI) \cdot 100$	**87.39**	87.23

Notes. (Left panel) Increases (in percentage) in the Z, F, and $1 - GI$ measures (from left to right) normalized by EA in Systems 1 and 6. (Right panel) Mean measures averaged over 12,000 instances of System 6. Sys, system.

(shown in the right panel of Figure 12). Although these results were very surprising at first, we believe that this is because the new probability values of the origin-destination pairs' requests in the reference set result in many vehicles traveling to the outskirts' areas to satisfy the large number of requests from the city center to the outskirts' areas. Then, the reference set has many more states that include the availability of vehicles in the outskirt areas, which leads to a better decision of which vehicle to choose for requests that can be served by vehicles in the outskirts. Indeed, we observed that when it was possible to select a vehicle from the outskirts, the percentage of cases in which such a vehicle was selected increased from 69% to 74.4% because of the above change. By that, these vehicles are less likely to keep waiting in far areas and are better utilized to serve more requests, which increases the efficiency measure F and the objective function Z. We obtained similar improvements when we increased the probabilities from the center to the outskirt areas to 0.7 and 0.85 because of the same reason mentioned above.

Second, we examined the case where the network layout of the city changed. We used the settings of the symmetric city network (System 5) for the reference set creation (with the same expiration time for all of the requests) and used for the online instances the same system setting as System 1. This resulted in a difference in the travel times t_{ij} between the system used to create the reference set and the system used in real time. In this case, we observed a deterioration in the system performance. Compared with the basic system, the mean increases of our policy normalized by EA were 0.74%, 0.34%, and 0.37% for the Z, F, and $1 - GI$ measures, respectively. However, such a change in the city layout is not likely to occur unexpectedly. Thus, this result demonstrates the importance of using a reference set that matches the correct network characteristics.

6.4.7. System 7: A More Challenging System. The objective of our last experiment is to assess the performance of our policy for a more challenging system (i.e., when the requests are harder to fulfill because of the system characteristics). To explore this, we modified the probability distribution of the four pairs of requests as follows. The probability for requests from an outskirt area to a center area was set to 0.55, whereas each of the remaining three probabilities was set to 0.15. These adjusted probabilities were applied both in the offline instances for the reference set construction and in the online instances to evaluate the policy in real time. We believe that it will be generally harder to fulfill requests in this system relative to the basic system because it has many requests that originate from the outskirts' areas, which are harder to reach within a limited time. This was confirmed by considering performance measures of the MILP solutions of this system. The MILPs' average objective value and fill rates were 47.82 and 0.87, respectively, for System 7 compared with 52.05 and 0.92, respectively, for the basic system.

With respect to the real-time policy for System 7, we observed that compared with EA, it improved by 2.28%, 1.26%, 0.94%, 4.09%, and 8.68% for the Z, F, $1 - GI$, $Z2$, and $Z3$ measures, respectively, and we note that these improvements are higher than those obtained for the basic system (see, e.g., Figure 12 in the previous section). It also significantly outperformed the best benchmark, MC-N (whose Z value outperformed EA by only 1.23%), and surpassed all other dispatching rules. This demonstrates that our policy is likely to have a greater advantage over other online policies when the system is harder to operate. It highlights the importance of leveraging information from the reference set to make better decisions on vehicle assignments when each decision has important implications for the future operations. Thus, our policy may be particularly attractive for even more complex systems (e.g., when considering also vehicle balancing or ride-sharing), which are interesting directions to consider as future research.

6.5. Discussion of the Numerical Results

Our experiments demonstrate the viability of the suggested approach to designing online policies that optimize a selected objective function, such as the Z function and the waiting time of requests. Despite the significance of the statistical tests that validate the superiority of our online policies, it is evident that their performance is sometimes not far from that of other dispatching rules. However, we demonstrated that a careful construction of the reference set can improve the performance of our approach even further and that the performance is robust to changes in the problem's input. Moreover, no one other rule is good for all systems and objective functions.

Considering the objective function Z, the generated policies maintain geographical fairness with a plausible efficiency loss for most systems. System 2 shows that postponing the expiration time of some requests is leveraged by our online policy to manage the efficiency-fairness trade-off. This system also highlights that allocating more travel time to pick up passengers from areas in the city outskirts is a convenient method to achieve geographically fair ride-hailing services. Beyond that, our online policy performs well for different instance sizes, whereas the other competing benchmarks, specifically MC-N, tend to accomplish lower measures with instances of a bigger size. The rule of MC-N seems to be a good heuristic to balance the efficiency-fairness trade-off, with a priority to the former measure, especially in the symmetric city network (System 5) and when the travel time is longer (System 2). When putting a higher emphasis on fairness in the objective function, the improvement of our policy becomes more significant. Concerning the objective of waiting time minimization, our policy has shown superiority compared with all of the other dispatching rules. It is the only rule that manages to maintain the smallest increase in the waiting time of requests while improving the rejection rate. We conclude that industry practitioners can adjust the definition of their MILP formulation's objective function to generate a policy that emphasizes their desired intention. For example, if only efficiency (the measure F) is desired, one can change the objective function to represent this preference and proceed according to the framework as long as the appropriate MILP formulation can be solved efficiently. We applied our framework with this objective and found that our method outperformed all other rules when we used the no elective rejection MILPs, which are more suitable for this objective, or when we used the probability distribution as reported for System 6 to generate a more diverse reference set. Thus, our method can replace the search and usage of intuitive decision rules, which might be hard to define for a general objective.

Finally, our experiments highlight an interesting phenomenon. The dispatching rules based on area criteria as well as our online policy perform better than rules based on vehicle criteria in all of the examined systems. It stresses the main role that geography has on the decision of vehicle assignments within the broad set of considerations of the problem. It also demonstrates the benefits of dividing the decision into several steps. First, choose the area from which a vehicle should be dispatched, and then, determine the specific vehicle, where its exact location might be less critical given the selected area. However, we note that the number of vehicles in our experiments was set to 15, which sometimes lead to the presence of a single available vehicle in the chosen area so that an additional decision on a specific vehicle was not needed.

7. Conclusion

The online ride-hailing problem with fairness addresses the challenge of assigning vehicles to dynamically and stochastically arriving requests in ride-hailing services. Unlike the common goal of optimizing only the efficiency of the service, this problem's objective is to achieve both efficiency and geographical fairness among passengers.

For that purpose, we suggested a solution approach that offers a new general method to develop online assignment policies based on solutions for offline versions of the problem. The new method suggests extracting information from these solutions to guide real-time assignment decisions chosen using a data-driven algorithm and distance measures. Because these principles are general, they can be used for other online problems as well. A limitation of this approach is the dependence of the offline solution on a known future of one instance. However, the generated online policy determines the assignment after considering solutions of multiple offline instances, which mitigates this dependency in a certain way. The numerical study presented in this paper demonstrates the viability of our approach to designing online policies. It outperformed all dispatching rules in all our experiments, including the fairness-oriented rule MC-N.

For future research, it would be interesting to use the solution approach for other dynamic and online problems. As indicated in Section 6.4.7, it may be particularly beneficial for even more complex systems, either because requests are harder to fulfill or because of additional complexities, such as vehicle balancing or ride-sharing. This approach could be implemented for any problem for which we have a method to solve its offline version (for example, an MILP formulation as we used) and possibly, other methods. However, implementing our approach to new problems will require adaptation of the state representation to account for the relevant information necessary for its decisions. Another interesting research topic is investigating how to build a good reference set that will provide good guidance for the real-time decisions.

References

Ackermann C, Rieck J (2023) A novel repositioning approach and analysis for dynamic ride-hailing problems. *EURO J. Transportation Logist.* 12:100109.

Agatz N, Erera A, Savelsbergh M, Wang X (2012) Optimization for dynamic ride-sharing: A review. *Eur. J. Oper. Res.* 223(2):295–303.

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.

Ausseil R, Pazour JA, Ulmer MW (2022) Supplier menus for dynamic matching in peer-to-peer transportation platforms. *Transportation Sci.* 56(5):1304–1326.

Beirigo BA, Schulte F, Negenborn RR (2022) A learning-based optimization approach for autonomous ridesharing platforms with service-level contracts and on-demand hiring of idle vehicles. *Transportation Sci.* 56(3):677–703.

Bent RW, Van Hentenryck P (2004) Scenario-based planning for partially dynamic vehicle routing with stochastic customers. *Oper. Res.* 52(6):977–987.

Berbeglia G, Cordeau JF, Laporte G (2010) Dynamic pickup and delivery problems. *Eur. J. Oper. Res.* 202(1):8–15.

Bertsimas D, Farias VF, Trichakis N (2012) On the efficiency-fairness trade-off. *Management Sci.* 58(12):2234–2250.

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

Bokányi E, Hannák A (2020) Understanding inequalities in ride-hailing services through simulations. *Sci. Rep.* 10(1):6500.

Chen X, Wang T, Thomas BW, Ulmer MW (2023) Same-day delivery with fair customer service. *Eur. J. Oper. Res.* 308:738–751.

Cordeau JF, Laporte G (2007) The dial-a-ride problem: Models and algorithms. *Ann. Oper. Res.* 153(1):29–46.

Dickerson JP, Sankararaman KA, Srinivasan A, Xu P (2021) Allocation problems in ride-sharing platforms: Online matching with offline reusable resources. *ACM Trans. Econom. Comput.* 9(3):13.

Dong Y, Wang S, Li L, Zhang Z (2018) An empirical study on travel patterns of internet based ride-sharing. *Transportation Res. Part C Emerging Tech.* 86:1–22.

Duan L, Wei Y, Zhang J, Xia Y (2020) Centralized and decentralized autonomous dispatching strategy for dynamic autonomous taxi operation in hybrid request mode. *Transportation Res. Part C Emerging Tech.* 111:397–420.

Eisenhandler O, Tzur M (2019) The humanitarian pickup and distribution problem. *Oper. Res.* 67(1):10–32.

Fagnant DJ, Kockelman KM (2014) The travel and environmental implications of shared autonomous vehicles, using agent-based model scenarios. *Transportation Res. Part C Emerging Tech.* 40:1–13.

Feng G, Kong G, Wang Z (2021) We are on the way: Analysis of on-demand ride-hailing systems. *Manufacturing Service Oper. Management* 23(5):1237–1256.

Heitmann R-JO, Soeffker N, Ulmer MW, Mattfeld DC (2023) Combining value function approximation and multiple scenario approach for the effective management of ride-hailing services. *EURO J. Transportation Logist.* 12:100104.

Jin ST, Kong H, Sui DZ (2019) Uber, public transit, and urban transportation equity: A case study in New York City. *Professional Geographer* 71(2):315–330.

Karp RM, Vazirani UV, Vazirani VV (1990) An optimal algorithm for on-line bipartite matching. *Proc. Twenty-Second Annual ACM Sympos. Theory Comput.* (ACM, New York), 352–358.

Kendall M, Stuart A (1963) *The Advanced Theory of Statistics Volume 1 Distribution Theory* (Griffin, London).

Kullman ND, Cousineau M, Goodson JC, Mendoza JE (2022) Dynamic ride-hailing with electric vehicles. *Transportation Sci.* 56(3):775–794.

Kumar A, Vorobeychik Y, Yeoh W (2022) Improving zonal fairness while maintaining efficiency in rideshare matching. *Proc. Twelfth Internat. Workshop Agents Traffic Transportation (ATT' 2022) (Vienna, Austria)*, 77–90.

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

Lesmana NS, Zhang X, Bei X (2019) Balancing efficiency and fairness in on-demand ridesourcing. *Advances in Neural Information Processing Systems*, vol. 32 (Curran Associates, Inc., Red Hook, NY), 477.

Lin K, Zhao R, Xu Z, Zhou J (2018) Efficient large-scale fleet management via multi-agent deep reinforcement learning. *KDD'18 Proc. 24th ACM SIGKDD Internat. Conf. Knowledge Discovery Data Mining (London)*, 1774–1783.

Lowalekar M, Varakantham P, Jaillet P (2018) Online spatio-temporal matching in stochastic and dynamic domains. *Artificial Intelligence* 261:71–112.

Ma W, Xu P, Xu Y (2021) Group-level fairness maximization in online bipartite matching. Preprint, submitted May 21, https://arxiv.org/abs/2011.13908.

Maciejewski M, Bischoff J, Nagel K (2016) An assignment-based approach to efficient real-time city-scale taxi dispatching. *IEEE Intelligent Systems* 31(1):68–77.

Mandell MB (1991) Modelling effectiveness-equity trade-offs in public service delivery systems. *Management Sci.* 37(4): 467–482.

Miao F, Han S, Lin S, Stankovic JA, Zhang D, Munir S, Huang H, He T, Pappas GJ (2016) Taxi dispatch with real-time sensing data in metropolitan areas: A receding horizon control approach. *IEEE Trans. Automation Sci. Engrg.* 13(2):463–478.

Nanda V, Xu P, Sankararaman KA, Dickerson J, Srinivasan A (2020) Balancing the tradeoff between profit and fairness in rideshare platforms during high-demand hours. *Proc. AAAI Conf. Artificial Intelligence* 24(2):2210–2217.

Neria G, Tzur M (2024) The dynamic pickup and allocation with fairness problem. *Transportation Sci.* 58(4):821–840.

Pan R, Yang H, Xie K, Wen Y (2020) Exploring the equity of traditional and ride-hailing taxi services during peak hours. *Transportation Res. Record* 2674(9):266–278.

Powell WB (2019) A unified framework for stochastic optimization. *Eur. J. Oper. Res.* 275(3):795–821.

Raman N, Shah S, Dickerson J (2021) Data-driven methods for balancing fairness and efficiency in ride-pooling. *Proc. Thirtieth Internat. Joint Conf. Artificial Intelligence* (ijcai.org), 363–369.

Reyes D, Erera A, Savelsbergh M, Sahasrabudhe S, O'Neil R (2018) The meal delivery routing problem. *Optim. Online* 1–70.

Schuller P, Fielbaum A, Alonso-Mora J (2021) Towards a geographically even level of service in on-demand ridepooling. *2021 IEEE Internat. Intelligent Transportation Systems Conf. (ITSC)* (IEEE, Piscataway, NJ), 2429–2434.

Sühr T, Biega AJ, Zehlike M, Gummadi KP, Chakraborty A (2019) Two-sided fairness for repeated matchings in two-sided markets: A case study of a ride-hailing platform. *Proc. 25th ACM SIGKDD Internat. Conf. Knowledge Discovery Data Mining* (ACM, New York), 3082–3092.

Sultana NN, Baniwal V, Basumatary A, Mittal P, Ghosh S, Khadilkar H (2021) Fast approximate solutions using reinforcement learning for dynamic capacitated vehicle routing with time windows. *Proc. Adaptive Learning Agents Workshop Twentieth Internat. Conf. Autonomous Agents Multiagent Systems (AAMAS 2021)*.

Sun J, Jin H, Yang Z, Su L, Wang X (2022) Optimizing long-term efficiency and fairness in ride-hailing via joint order dispatching and driver repositioning. *KDD'22 Proc. 28th ACM SIGKDD Conf. Knowledge Discovery Data Mining* (ACM, New York), 3950–3960.

Tavor S, Raviv T (2023) Anticipatory rebalancing of RoboTaxi systems. *Transportation Res. Part C Emerging Tech.* 153:104196.

Thebault-Spieker J, Terveen L, Hecht B (2017) Toward a geographic understanding of the sharing economy: Systemic biases in UberX and TaskRabbit. *ACM Trans. Comput-Human Interaction* 24(3):21.

Ulmer MW, Thomas BW, Campbell AM, Woyak N (2021) The restaurant meal delivery problem: Dynamic pickup and delivery with deadlines and random ready times. *Transportation Sci.* 55(1):75–100.

Voccia SA, Campbell AM, Thomas BW (2019) The same-day delivery problem for online purchases. *Transportation Sci.* 53(1):167–184.

Wang H, Bei X (2022) Real-time driver-request assignment in ridesourcing. *Proc. Thirty-Sixth AAAI Conf. Artificial Intelligence (AAAI-22)* 36(4):3840–3849.

Wang H, Yang H (2019) Ridesourcing systems: A framework and review. *Transportation Res. Part B Methodological* 129:122–155.