Annals of Operations Research
https://doi.org/10.1007/s10479-022-04560-3

ORIGINAL RESEARCH

Adaptive forecast-driven repositioning for dynamic ride-sharing

Martin Pouls[1] · **Nitin Ahuja**[2] · **Katharina Glock**[1] · **Anne Meyer**[3]

Accepted: 19 January 2022

Abstract
In dynamic ride-sharing systems, intelligent repositioning of idle vehicles often improves the overall performance with respect to vehicle utilization, request rejection rates, and customer waiting times. In this work, we present a forecast-driven idle vehicle repositioning algorithm. Our approach takes a demand forecast as well as the current vehicle fleet configuration as inputs and determines suitable repositioning assignments for idle vehicles. The core part of our approach is a mixed-integer programming model that aims to maximize the acceptance rate of anticipated future trip requests while minimizing vehicle travel times for repositioning movements. To account for changes in current trip demand and vehicle supply, our algorithm adapts relevant parameters over time. We embed the repositioning algorithm into a planning service for vehicle dispatching. We evaluate our forecast-driven repositioning approach through extensive simulation studies on real-world datasets from Hamburg, New York City, Manhattan, and Chengdu. The algorithm is tested assuming a perfect demand forecast and applying a naïve forecasting model. These serve as an upper and lower bound on state-of-the-art forecasting methods. As a benchmark algorithm, we utilize a reactive repositioning scheme. Compared to this, our forecast-driven approach reduces trip request rejection rates by an average of 3.5 percentage points and improves customer waiting and ride times.

Keywords Repositioning · Ride-sharing · Dial-a-ride · Mobility-on-demand

1 Introduction

While the popularity of mobility-on-demand (MOD) services such as Uber and Lyft has increased significantly in recent years, this growth has also led to increased traffic congestion (Castiglione and Cooper 2018). Several cities have identified this issue and some have even taken countermeasures (Doubek 2018). One way to tackle this problem is to increase the share of dynamic ride-sharing services such as UberPool or MOIA. In these services, multiple

✉ Martin Pouls
pouls@fzi.de

1 FZI Research Center for Information Technology, Karlsruhe, Germany

2 PTV Group, Karlsruhe, Germany

3 TU Dortmund University, Dortmund, Germany

Published online: 16 February 2022

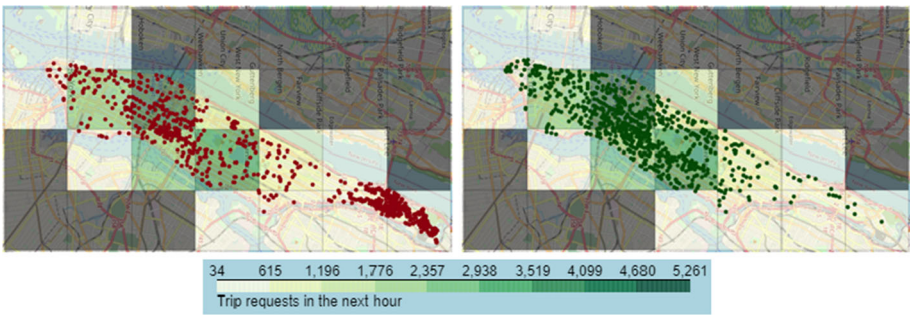

Fig. 1 Points denote vehicle positions without (left) and with (right) repositioning. Grid cells show the number of trip requests by area in the next hour

passengers with different destinations share the same vehicle. Compared to classical MOD services for individual customers, this leads to slight delays due to detours for picking up or dropping off other passengers. Generally, these detours as well as initial waiting times are constrained to a couple of minutes ensuring customer satisfaction. Overall, one maintains the flexibility of MOD services compared to traditional public transport and improves vehicle utilization at the same time.

Planning problems regarding MOD services in general and dynamic ride-sharing, in particular, have generated significant research attention. Most works focus on the vehicle routing aspect, i.e. solving the dynamic dial-a-ride-problem arising in these applications (Alonso-Mora et al. 2017a; Ma et al. 2019). In this work, we focus on the idle vehicle repositioning problem, i.e. the problem of sending idle vehicles to a suitable location in anticipation of future demand.

In general, the overall performance of a ride-sharing system may be impacted significantly by suitable repositioning algorithms. Figure 1 exemplifies this fact by comparing vehicle positions in scenarios without and with repositioning after several hours of service.

Without repositioning vehicles become stuck in low-demand areas. In turn, requests in other areas are rejected due to a lack of nearby vehicles. This is due to the assumption of a maximum waiting time for customers in dial-a-ride problems. A vehicle must reach the customer within this time frame, otherwise, the customer is rejected. Thus, vehicles in low-demand areas cannot reach many new trip requests in time and are consequently rarely assigned a new route. This phenomenon may be avoided by using a repositioning mechanism.

In many practical applications with self-employed drivers, this problem is currently solved decentrally by incentivizing drivers to reposition towards areas with low vehicle supply. For instance, Uber employs so-called “surge pricing” which raises prices in areas with excess demand and thereby offers increased revenue opportunities to drivers (Uber 2020). In this paper, we propose the usage of a central repositioning strategy that may improve system performance in use cases with a central fleet operator.

The main objective of the repositioning procedure is to serve more customers given a fixed vehicle fleet in large-scale MOD services. The main contribution of this paper is threefold. First, we propose an efficient model-based repositioning algorithm that intelligently repositions idle vehicles based on the current system state and a short-term demand forecast. Our algorithm is applicable in real-time even for large-scale and dynamic systems and may be combined with an existing approach for dispatching vehicles to trip requests. Second, we introduce an integrated adaptive parameter tuning technique to adequately consider temporal

and spatial differences in the number of trip requests that vehicles are expected to serve, which is crucial in the case of ride-sharing services. This mechanism ensures that the proposed approaches are usable in real-world settings without requiring extensive parameter tuning or a high effort for data collection and processing and that they can be combined with existing routing and dispatching solutions. Third, we perform an extensive evaluation on four real-world datasets showing that our approach is able to improve the system performance significantly compared to a myopic benchmark strategy.

The main addition compared to our prior work (Pouls et al. 2020) is the introduction of a novel adaptive parameter tuning technique that removes the need for an a priori parameter tuning and enables us to flexibly adjust the parameters to account for the spatial and temporal characteristics of different real-world application scenarios. In addition, we perform an extensive computational evaluation of our algorithm on real-world data. For this purpose we use a simulation-based evaluation framework that enables us to evaluate the repositioning algorithm within the scope of a dynamic ride-sharing application.

The remainder of this work is organized as follows. Section 2 gives an overview of related work regarding idle vehicle repositioning as well as demand forecasting. In Sect. 3 we describe our overall planning service encompassing repositioning as well as dispatching and forecasting. In addition, we cover the discrete-event simulation that is used for evaluations. Our repositioning approach itself is detailed in Sect. 4. Finally, Sect. 5 presents our computational results on real-world datasets while Sect. 6 summarizes our findings and gives some possible directions for future work.

2 Related approaches in repositioning and demand forecasting

To the best of our knowledge, there are relatively few papers dealing with repositioning explicitly in the context of large-scale dynamic ride-sharing applications. However, there exist several closely related fields. There is a large body of work on the dynamic vehicle routing problem with stochastic customers (DVRPSC). In this vehicle routing variant, part of the customers arrive dynamically and there is some form of stochastic knowledge about customer arrivals that may be used during planning. There also exist a large number of works regarding repositioning in other mobility-on-demand services besides ride-sharing. For instance, repositioning is a widely considered problem in car-sharing systems. In these systems, customers rent a vehicle for a desired time period. Repositioning is particularly important for systems in which customers may drop off the vehicle at a different location than their pickup. There are some key differences between car- and ride-sharing. In car-sharing, there are no dedicated drivers and customers drive the rented vehicles themselves. In addition, there is no sharing between different customers, a vehicle is assigned to exactly one customer at a time.

We structure our literature review as follows. We first consider related work regarding repositioning in dynamic ride-sharing systems. These are the most similar to the algorithm presented in this paper. Subsequently, we review literature concerning the DVRPSC as well as repositioning approaches for car-sharing and taxi services. Finally, we discuss existing approaches for short-term travel demand forecasting, a vital component in our repositioning approach.

Table 1 Related work on repositioning for dynamic ride-sharing

Paper	Algorithmic approach	Routing interaction	Instance size
Alonso-Mora et al. (2017a)	Reactive	Separate	460,700 / day
Alonso-Mora et al. (2017b)	Sampling	Integrated	460,700 / day
Jung and Chow (2019)	Repositioning policies	Separate	145,643 / 6 hours
Riley et al. (2020)	MIP	Separate	59,820 / 2 hours
Shah et al. (2020)	ADP	Integrated	19,820 / hour
Lowalekar et al. (2021)	Stochastic program	Integrated	403,770 / day

2.1 Repositioning in dynamic ride-sharing

Most closely related to the algorithm presented in this paper are works dealing with repositioning in the context of dynamic ride-sharing systems. A summary of existing approaches is provided in Table 1. We report the algorithmic approach as well as the instance sizes used for evaluation. Instance sizes are given as the number of trip requests over time. To ensure the comparability of the approaches, we report the largest instances measured by the number of trip requests per unit of time, as far as possible based on the information given. In addition, approaches may be divided by their interaction with the vehicle routing algorithm. Repositioning may either be integrated with routing, i.e. routing and repositioning decisions are taken simultaneously. Alternatively, repositioning decisions are determined by a separate algorithm.

Alonso-Mora et al. (2017a) propose a reactive repositioning policy with the idea of sending idle vehicles to the desired pickup locations of rejected trip requests. Given a batch of rejected requests, idle vehicles are matched to the corresponding pickup locations while minimizing travel times for repositioning movements. We use a similar approach as a benchmark for our solution algorithm. In a follow-up paper, Alonso-Mora et al. (2017b) present a more refined approach similar to the sampling-based algorithms for the DVRPSC presented in Section 2.2. They include predicted trip requests in their vehicle routing algorithm. These requests are served with a lower priority than actual trip requests. The authors show that this approach leads to reduced waiting times and in-car travel delays compared to the reactive repositioning from their previous work (Alonso-Mora et al. 2017a). However, no noticeable improvement in the number of rejected trip requests is achieved. Jung and Chow (2019) present two policies in which vehicles reposition according to historical pickup probabilities. Vehicles either move to a zone or a depot. The probability of selecting a zone or depot is proportional to the historical distribution of trip requests. The authors compare these approaches to a setting without repositioning and show that both repositioning algorithms improve the request acceptance rate at the cost of an increase in the total distance traveled by vehicles. In contrast to our work, the authors do not consider detailed information about supply and demand. In particular, neither the current configuration of the vehicle fleet nor the total demand is considered during repositioning. Riley et al. (2020) propose a MIP-based solution approach similar to the one presented in this work. They use a two-step formulation that first determines the number of vehicles to be repositioned between a pair of zones and subsequently selects specific vehicles. Their algorithm integrates a demand forecast containing the number of trip requests between each pair of zones. Hence, in contrast to this work, they expect a more fine-grained forecast which may not be available in practice. Moreover, they do not allow for the rejection of trip requests but rather penalize long customer waiting times. They evaluate their

approach on data from New York City and show that their repositioning approach decreases customer waiting times compared to performing no repositioning. Shah et al. (2020) propose a learning-based approach that assesses the future value of routing decisions. However, the approach does not explicitly reposition idle vehicles. It merely considers the future value of routing decisions when assigning trip requests to vehicles. They evaluate their algorithm on data from New York City and show that it yields a decrease in rejected trip requests compared to myopic routing approaches. Lowalekar et al. (2021) propose another approach that includes samples of future trip requests to build routes that are suitable to accommodate upcoming requests. The authors evaluate their approach on two real-world datasets from NYC and an undisclosed location and show that the trip request rejection rate is reduced compared to myopic approaches.

2.2 Related work in dynamic vehicle routing with stochastic customers

In the DVRPSC, part of the trip requests arrive dynamically and there is some form of exploitable stochastic information about these trip requests. The problem has been widely studied in literature. A selection of solution approaches is summarized in Table 2. For more extensive reviews we refer the reader to Ritzinger et al. (2016) and Ulmer et al. (2020). In our review, we adopt the classification by Ulmer et al. (2020) that distinguishes between the following general solution methods:

- *Lookahead algorithm (LA)* Approaches that use a lookahead, for instance, samples of anticipated trip requests, to improve routing decisions.
- *Policy function approximation (PFA)* Algorithms that aim to approximate a policy, often inspired by decision making in practice.
- *Value function approximations (VFA)* Procedures that derive a value function for routing decisions via simulations and related learning techniques.

Among the earliest works in the field of DVRPSCs are waiting strategies (Mitrović-Minić and Laporte 2004; Ichoua et al. 2006; Thomas 2007), which represent examples of PFA algorithms. In these algorithms, the aim is to decide when and where a vehicle should wait while executing a route in order to be well-positioned for dynamically arriving trip requests. While these approaches yield benefits for many application settings such as parcel delivery (Mitrović-Minić and Laporte 2004), the problems arising from these applications are structurally different from the setting of ride-sharing. For instance, they tend to be less dynamic as a large portion of customer requests is known in advance, while in this work we consider purely dynamic trip requests. Additionally, vehicles in ride-sharing tend to have little to no waiting times while executing a route as dynamically arriving customers want to be serviced immediately. Moreover, when transporting passengers, it is difficult to justify waiting times whenever there are customers aboard the vehicle or waiting for its arrival. In contrast, for applications such as parcel logistics, this aspect is less of a problem. Therefore, in our view, such waiting strategies are not directly applicable to the use case considered in this paper.

The most prevalent approach for lookahead algorithms is the inclusion of samples of predicted customers in the routing algorithm. Among these sampling-based algorithms are the multiple scenario approaches (MSA) by Bent and Van Hentenryck (2004); Bent and Hentenryck (2007), in which multiple routing plans are generated based on different samples of future customers and a so-called "distinguished" plan is selected via consensus mechanisms. Ferrucci et al. (2013) also propose a sampling-based approach in the form of a tabu search (TS) that includes sampled future customers. Due to their computational complexity, most

Table 2 Related work on the DVRPSC

Paper	Algorithmic approach	Classification by Ulmer et al. (2020)	Instance size
Bent and Van Hentenryck (2004)	MSA	LA	100 / day
Mitrović-Minić and Laporte (2004)	Waiting strategy	PFA	1,000 / 10 hours
Ichoua et al. (2006)	Waiting strategy	PFA	36 / hour
Bent and Hentenryck (2007)	MSA	LA	100 / day
Thomas (2007)	Waiting strategy	PFA	50 / day
Mes et al. (2010)	Auction	VFA	4.5 / hour
Schmid (2012)	ADP	VFA	90 / day
Ferrucci et al. (2013)	TS with sampling	LA	150 / day
Ulmer et al. (2018)	ADP	VFA	100 / 6 hours
Ulmer et al. (2019)	ADP	VFA	100 / 6 hours
Voccia et al. (2019)	ADP	VFA	192 / day

sampling-based approaches have not been tested on large instance sizes as commonly seen in ride-sharing.

A third major direction of DVRPSC research are VFA algorithms such as approximate dynamic programming (Mes et al. 2010; Schmid 2012; Ulmer et al. 2018, 2019; Voccia et al. 2019). The idea behind these approaches is to determine a value function for routing decisions that incorporates their impact on the handling of future customers. From the practical perspective of ride-sharing applications, these algorithms have two main drawbacks. First, they need a relatively large amount of training data to approximate the value function. Hence, launching a ride-sharing service in a region where no prior data is available becomes a challenge. Second, these VFA approaches are also not tested on large instances and may not deliver the necessary computational performance for processing large numbers of trip requests.

2.3 Repositioning for car-sharing and taxi services

Vehicle repositioning is a widely studied problem in the context of car-sharing services. One may differentiate between operator-based and user-based repositioning. In the former, the car-sharing operator employs personnel to reposition vehicles while in the latter car-sharing users are incentivized to pick up or drop off cars at certain locations. In our literature review, we focus on operator-based repositioning as it shares a closer resemblance with central repositioning approaches for ride-sharing. A summary of recent works in this field is presented in Table 3. A more detailed review may be found in Huang et al. (2020b).

Algorithms are often based on MIP formulations (Nourinejad and Roorda 2014; Repoux et al. 2015; Boyacı et al. 2017; Gambella et al. 2018; Xu and Meng 2019; Huang et al. 2020b), but also include metaheuristics (Bruglieri et al. 2019) and Markov chain based models (Repoux et al. 2019). There are some key differences to the application setting of

Table 3 Related work on repositioning for car-sharing

Paper	Algorithmic approach	Instance size
Nourinejad and Roorda (2014)	MIP	200 / day
Repoux et al. (2015)	MIP	200 / day
Boyacı et al. (2017)	MIP	300 / day
Gambella et al. (2018)	MIP	50 / day
Bruglieri et al. (2019)	ALNS	100
Repoux et al. (2019)	Markov chain model	400 / day
Xu and Meng (2019)	MIP	125 / day
Huang et al. (2020b)	MIP	125,000 / day

ride-sharing. Most importantly, in car-sharing systems, there is a one-to-one relation between a customer and vehicle. In contrast, ride-sharing allows for multiple customers to share the same vehicle. Therefore, one key challenge when making repositioning decisions in a ride-sharing system is to estimate how many vehicles are needed to serve a given number of trip requests. As this depends on the structure of the trip requests, it may vary between datasets but also between locations or the time of day within the same dataset. On the other hand, repositioning approaches for car-sharing systems focus on specific aspects of that application domain. These include scheduling the personnel that carries out the repositioning movements, considering different operation modes (two-way, one-way, free-floating) and different reservation schemes. Hence, we believe that due to the structural differences between car- and ride-sharing, specific solution approaches are needed for both applications.

Repositioning approaches have also been proposed for ride-hailing and classical taxi services. In these services, no sharing takes place and only one group of customers uses the vehicle at a time. Both Li et al. (2011) and Powell et al. (2011) use GPS traces of taxis to identify profitable regions. However, as with car-sharing, in classical taxi services, a vehicle serves at most one customer at a time. In addition, these solution approaches take the viewpoint of a taxi driver and optimize the profit, while our goal is to optimize the system-wide performance.

2.4 Short-term travel demand forecasting

In our solution approach, we utilize a short-term forecast of the anticipated trip requests. The general field of short-term travel demand forecasting and related spatio-temporal prediction problems has been studied extensively. For reviews we refer the reader to Vlahogianni et al. (2004, 2014). Most approaches provide information in an aggregated form, i.e. they offer a forecast of the total number of trip requests for a given area and interval of time rather than predicting individual trip requests (Vlahogianni et al. 2014). Classical approaches include time series models such as ARIMA or Kalman filters (Li et al. 2012; Lippi et al. 2013) as well as statistical learning (Huang et al. 2020a). More recently, convolutional neural networks (CNN) and long short-term memory (LSTM) recurrent neural networks have emerged as suitable techniques for modeling the complex spatial and temporal dependencies typically found in these forecasting problems (Zhang et al. 2018; Liao et al. 2018; Yao et al. 2018; Ke et al. 2017; Yao et al. 2019). For instance, Yao et al. (2018); Yao et al. (2019) combine a CNN for modeling spatial dependencies with a LSTM architecture that reflects the temporal

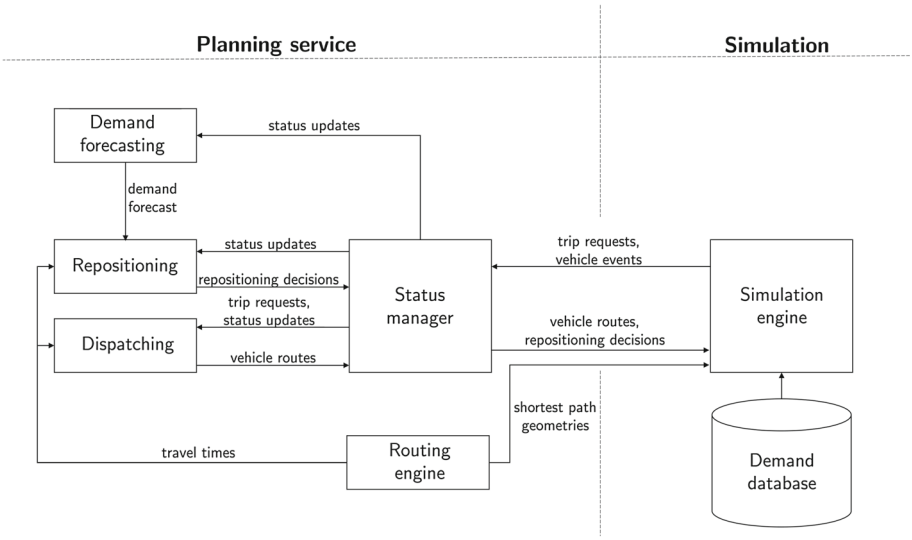

Fig. 2 Planning service, simulation and relevant communication

aspects. They evaluate their approach on New York City taxi data and achieve a significant improvement over a simple historical average as well classical machine learning algorithms such as gradient boosting.

In the remainder of this work, we will not focus on the forecasting methodology itself but rather on the usage of a forecast to improve planning results. However, the assumptions of our algorithm regarding the structure of the forecast conform to state-of-the-art forecasting techniques, i.e. they expect the predicted number of trip requests per area and interval of time as an input. This way, we ensure that these approaches can be used in combination with our algorithm.

3 A system design for dynamic ride-sharing

This section provides an overview of the larger system for dynamic ride-sharing into which the proposed repositioning mechanism is embedded. Similar to several recent approaches (see Sect. 2.1), we use separate components for the main planning tasks of dispatching and repositioning. Consequently, the proposed repositioning procedure can be integrated with existing routing mechanisms for ride-sharing services. This setup also ensures that each component is less complex compared to an integrated approach and that routing and repositioning decisions can be understood and accepted by human operators interacting with the planning system. Moreover, the system design ensures that no component is dependent on prior information or extensive training. This enables the usage of the proposed method in new service areas where a large amount of training data that is necessary for learning-based approaches may not be available. The overall system and communication between components is depicted in Fig. 2.

It consists of a set of modular components separated into two parts, the planning service and a simulation that emulates the behavior of real-world customers and vehicles. By strictly separating planning and simulation components, the planning service could be directly

transferred to a real-world use case. The different components that make up this system are discussed in the following.

3.1 Planning service

All planning and forecasting functionality is consolidated in the planning service, which consists of separate decoupled components.

Status manager The status manager handles external communication and maintains the current status of vehicles and trip requests. It processes incoming events regarding new trip requests, pickup and delivery of customers, start and end of repositioning movements as well as regular vehicle location updates. All this information is consolidated and provided as an up-to-date system state to the planning and forecasting components. The status manager also triggers planning actions in other components such as the dispatching of new trip requests.

Dispatching Vehicle routing is handled by the dispatching module. Essentially it solves a dynamic dial-a-ride problem with the typical constraints: vehicle capacity, waiting time, and customer ride time (Cordeau and Laporte 2007). The waiting time is limited to a fixed maximum and corresponds to the time a customer has to wait after submitting a request until a vehicle arrives at the requested pickup location. The ride time on the other hand corresponds to the total travel time of a customer inside the vehicle. The maximum allowed ride time of a single trip request may for instance be calculated as the direct travel time between the desired pickup and delivery locations plus a permissible detour. Each trip request is processed immediately upon arrival to guarantee quick response times. A request is either accepted and inserted into the route of a vehicle or rejected if no feasible solution is found. For this process, we utilize an insertion heuristic inspired by the works of Ma et al. (2015, 2019). Our modified dispatching algorithm can be roughly outlined as follows.

1. *Vehicle search* Based on a spatial index data structure, vehicles near the pickup location of the new trip request are identified. We consider all vehicles that may reach the new request within the specified maximum acceptable waiting time of the request.
2. *Vehicle sorting* The selected vehicles are sorted based on their suitability for the new trip request. In this step, we perform a rough approximation regarding the impact of inserting the new request into a vehicle route. No detailed checking of constraints and in particular no expensive shortest path calculations are performed.
3. *Insertion check* Sorted vehicles are iteratively checked for a feasible insertion. Feasible insertions are rated based on the resulting increase in vehicle travel time and the best feasible insertion is performed. If no feasible insertion is found, the trip request is rejected. The prior sorting of vehicles allows us to prematurely abort the search after evaluating at least 50 vehicles if a feasible insertion was found. Otherwise, the search continues until either an insertion is found or all eligible vehicles have been evaluated. The premature abortion allows for faster running times and is particularly useful when working with large datasets such as in New York City with a peak demand of up to 20, 000 trip requests per hour. In this case, individual requests must be processed within 100 - 200 ms. Additionally, response times for the customer are quicker, improving customer satisfaction.

Repositioning The repositioning module monitors the current vehicle fleet configuration and demand forecast. Based on this information it regularly repositions vehicles to new locations. Repositioning decisions are made in real-time in order to ensure efficient usage in combination with the dispatching algorithm. As seen in Sect. 2.1, some approaches combine repositioning and vehicle routing decisions. We opted to decouple these decisions for the

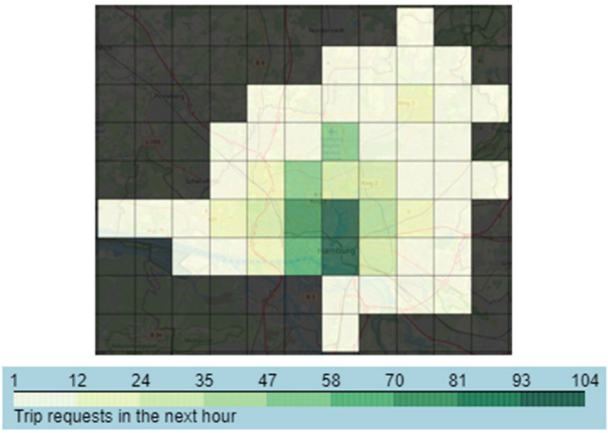

Fig. 3 Demand forecast for Hamburg

following reasons. First, it enables us to flexibly combine the repositioning algorithm with any vehicle routing algorithm. For instance, a different vehicle routing approach could be used depending on the specific dataset or special constraints for a use case. Second, the repositioning approach may be used with existing demand forecasting models whereas many integrated approaches require more detailed demand information. Our algorithmic approach for vehicle repositioning is detailed in Sect. 4.

Routing engine The main task of the routing engine is to answer shortest path queries and provide travel times as well as route geometries to other components. For this purpose, we work with road networks derived from OpenStreetMap (OSM) data and use a state-of-the-art open-source routing engine (Dibbelt et al. 2016).

Demand forecasting The repositioning approach requires information on the anticipated number of trip requests within a set of areas over a given forecast horizon. Figure 3 illustrates an exemplary demand forecast for Hamburg utilizing a partitioning of the map into grid cells. This structure conforms to state-of-the-art forecasting algorithms as outlined in Sect. 2.4. Thus, any forecasting approach providing information in this form could be integrated into our system in the form of a forecasting module. The forecasting modes that are used for evaluation purposes in this paper are discussed in Sect. 5.4.

3.2 Simulation

The simulation part of our system consists of the necessary input data in form of a trip demand database and the simulation engine itself.

Demand database The simulation operates on a database of historic trip requests. Each trip request contains the request time, pickup and destination coordinates as well as the number of passengers. Throughout the remainder of this paper, we work with trip requests obtained from real-world taxi services. However, if no real-world data is available, one could also use

other data sources such as macroscopic traffic simulations or public transport data to generate trip requests.

Simulation engine The discrete-event simulation is responsible for simulating the behavior of vehicles and customers. On startup, the simulation obtains all trip requests in the simulated time period from the demand database. In addition, a vehicle fleet of a given size is created. Initial vehicle positions are sampled from historical pickup locations of trip requests and therefore roughly mirror the expected spatial distribution of demand. The simulation now replays all trip requests and enriches them with scenario-specific parameters such as the maximum waiting time or the maximum ride time. Each simulated trip request is sent to the planning service for dispatching. If the trip request is accepted, the simulation obtains a new vehicle route as a response. Combined with repositioning assignments, these vehicle routes form the basis for the simulation of vehicle behavior. A vehicle moves immediately to its next target which either corresponds to the next stop in its route or its repositioning target. In order to realistically model vehicle movement, we work with paths on a road network obtained from our routing engine. For this study, we assume free-flow travel times on the arcs of the road network. Upon arrival at its target, a vehicle generates relevant events and sends them to the plannings service. These include, for instance, the pickup and delivery of customers or the end of a repositioning movement. Additionally, while traveling to a new location, each vehicle regularly emits updates regarding its current location.

4 Forecast-driven repositioning

In this section, we provide a detailed description of our forecast-driven repositioning algorithm (FDR) and additionally introduce a reactive repositioning strategy (REACT) as a benchmark.

4.1 Forecast-driven repositioning algorithm

The core idea of our algorithm FDR is to cover forecasted trip request demand by intelligently repositioning idle vehicles. The central component of our approach is a MIP model (FDR-M) that is solved at regular intervals. In this model, we aim to balance supply and demand by maximizing the sum of covered demand and minimizing the number of repositioning movements as well as vehicle travel times. Anticipated demand is given by a forecast that outputs the expected number of trip requests for a given set of areas and a forecast horizon. Supply on the other hand is provided by the vehicle fleet. We assume that vehicles may cover demand in the neighborhood of their current or assigned location. This neighborhood is intuitively defined as the set of areas that may be reached within the maximum allowed waiting time of a customer. A single vehicle is assumed to cover multiple trip requests over the forecast horizon. However, the precise number varies drastically by dataset, location, and time of day. For instance, at night, when demand is rather low, vehicles tend to serve fewer customers in the forecast horizon as fewer trip requests can be combined in a single vehicle route. A similar behavior may be observed when comparing low and high demand areas with vehicles in high-demand areas serving more customers within the forecast horizon. Thus, we integrate an adaptive parameter tuning technique into our approach that estimates the expected number of served trips for a vehicle located in a certain area. As this parameter is updated each time we solve FDR-M, it reflects the spatial and temporal differences in

the utilization of vehicles. Based on this adaptive parameter tuning and the current vehicle schedules, we may calculate the supply provided by our vehicle fleet. By repositioning idle vehicles, we are now able to shift supply from low-demand areas to areas with excess demand.

FDR-M is embedded into a planning process that we explain in the following section. Subsequently, we introduce the MIP model itself alongside the necessary notation and the adaptive parameter tuning technique.

4.1.1 Planning process

FDR-M is embedded into a rolling horizon planning process which is triggered at a regular interval f (e.g. every three minutes). The four main steps of the planning process are as follows:

1. Obtain an up-to-date demand forecast.
2. Perform adaptive parameter tuning as detailed in Sect. 4.1.4.
3. Solve FDR-M as given in Sect. 4.1.3.
4. Determine an optimal assignment of vehicles to repositioning targets.

In the first step, a demand forecast is obtained that yields the number of expected trip requests for a set of areas within a forecast horizon (e.g. 1 hour). Combined with the current state of the vehicle fleet, this forecast serves as an input for FDR-M. In a second step, parameters for FDR-M are updated based on the fleet's performance in a short period of time before the repositioning algorithm is triggered. This enables us to adjust to varying spatial and temporal conditions. Subsequently, we solve FDR-M which determines repositioning assignments on an aggregated level. As an output, we obtain the number of vehicles repositioned between any pair of areas i and j, denoted as x_{ij}. Lastly, these aggregated values are translated into actionable decisions. First, we randomly sample x_{ij} repositioning targets from the set of feasible target locations L_j in the target area for all pairs $i, j \in A$ where $i \neq j$. This set of feasible targets is determined based on prior trip requests, i.e. each past pickup location is a feasible repositioning target. Afterward, we determine an optimal assignment of available idle vehicles to the sampled target locations via a network flow formulation. This assignment minimizes the total travel time necessary for repositioning movements. In the remainder of this section, we will take a detailed look at the second and third steps of the planning process which constitute the core portion of our algorithm.

4.1.2 General notation

All relevant notation used throughout this section is summarized in Table 4. K denotes the set of all vehicles. This set may be further subdivided into idle vehicles K^{id}, active vehicles serving a route K^{act} and vehicles on a repositioning trip K^{re}. In addition, A represents a set of areas. For the remainder of this work, we utilize a partitioning of the region under study into square grid cells. The size of these cells is given by g (e.g. 3000 m) and denotes the length of the sides. Figure 4 illustrates the utilized grid among other key concepts that will be explained throughout this section. For the purpose of travel time calculations, we assume that these areas are represented by their center and calculate a travel time t_{ij} between the centers of i and j. Centers are not the exact centroid of an area but rather the closest node on the road network. Consequently, travel times are calculated from the shortest paths on the road network. The maximum travel time between any pair of areas is denoted as $T^{max} = \max_{i,j \in A} t_{ij}$. We may now further divide the sets of vehicles by area $i \in A$ as K_i^{id},

Table 4 Notation for forecast-driven repositioning

Sets	
A	Areas
A^{re}	Valid target areas for repositioning
K	Vehicles
$K^{id} \mid K^{act} \mid K^{re}$	Idle | active | repositioning vehicles
$K_i^{id} \mid K_i^{act} \mid K_i^{re}$	Idle | active | repositioning vehicles per area $i \in A$
K^{N_i}	Vehicles in the neighborhood of i at the start of h^-
L_i	Feasible repositioning target locations in $i \in A$
N_i	Neighborhood of i
Parameters	
α_k	Active percentage of vehicle k
$\hat{d}_i$	Demand forecast for $i \in A$
$\hat{D}$	Cumulated demand forecast for all areas $i \in A$
d_k^-	Performed delivery operations by vehicle k in h^-
d_k^+	Planned delivery operations by vehicle k in h^+
$\hat{e}_i$	Expected number of requests for a vehicle in $i \in A$ in h^+
e_k	Potential number of requests served for vehicle k in h^-
f	Interval of time in which algorithm FDR is run
g	Size of grid cells
h	Forecast horizon in minutes
$h^+ \mid h^-$	Time periods covering the previous and next h minutes
k^{min}	Minimum number of vehicles for neighborhood calculations
p_k^-	Performed pickup operations by vehicle k in h^-
p_k^+	Planned pickup operations by vehicle k in h^+
S_i^{act}	Supply provided by active vehicles in area $i \in A$
S_i^{re}	Supply provided by repositioning vehicles to area $i \in A$
t_{ij}	Travel time from area i to j
T^{max}	Maximum travel time $\max_{i,j \in A} t_{ij}$
t^c	Coverage radius
w^{cov}	Objective function weight for covered demand
w^{mov}	Objective function weight for repositioning movements
w^t	Objective function weight for coverage travel times
w_i	Objective function weight for coverage of demand in area $i \in A$
Decision variables	
$c_{ij} \in \mathbb{R}_0^+$	Provided demand coverage from area i to j
$x_{ij} \in \mathbb{N}_0^+$	Number of vehicles repositioned from area i to j

K_i^{act}, K_i^{re}. Note that the sets K_i^{id} and K_i^{act} contain those vehicles currently situated in area i. K_i^{re} on the other hand consists of vehicles currently repositioning towards i. Vehicles may only be repositioned to valid target areas $A^{re} \subseteq A$. In this study, we limit A^{re} to areas with at least one prior pickup. This is necessary as we sample specific repositioning targets L_i

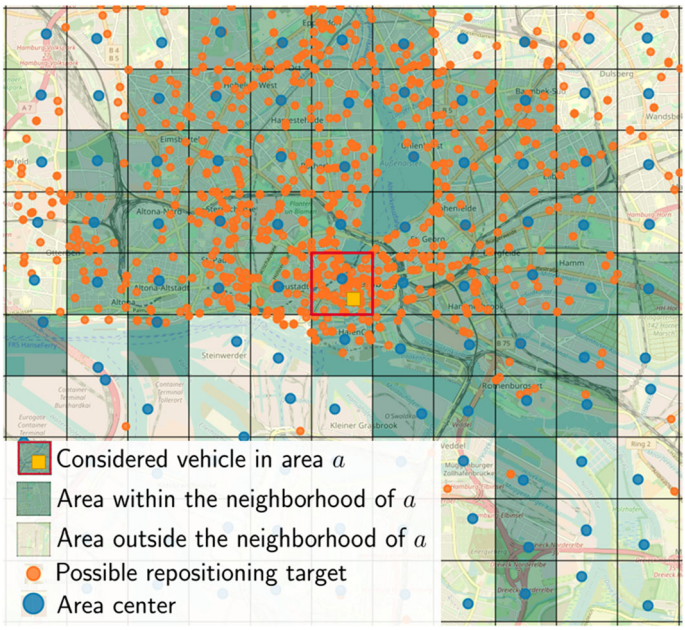

Fig. 4 Illustration of our grid-based map partitioning, neighborhood of the considered area a, area centers, and possible repositioning targets

for each $i \in A^{re}$ from past pickup locations. An example of possible repositioning targets may be seen in Fig. 4. In practical applications, A^{re} might be determined based on suitable waiting spots for vehicles. Our model works on a demand forecast denoted as $\hat{d}_i$ for each $i \in A$. This forecast gives us the expected number of trip requests originating in an area i within a forecast horizon of h minutes. h^+ and h^- denote time periods covering the previous and next h minutes respectively.

4.1.3 Repositioning model

The complete model FDR-M is given in equations (1) – (8). In the following, we will first describe the decision variables and subsequently, we take a detailed look at the model, its objective function, and constraints. In Sect. 4.1.4, we present the details of our adaptive parameter tuning process that is utilized when building the model.

Decision variables Our model contains two sets of decision variables. Integer variables $x_{ij}|i, j \in A$ correspond to our actual repositioning decisions and denote the number of vehicles repositioned from area i to j. Variables $c_{ij}|i, j \in A$ denote the coverage that is provided by vehicle resources located in or assigned to i for forecasted trip demand in j. This concept of provided coverage is central to our model. We assume that any vehicle may serve multiple trip requests over the forecast horizon in the neighborhood N_i of its current or assigned location in area i. This neighborhood is defined as the set of areas $j \in A$ that lie within a given coverage radius t^c of i. This coverage radius corresponds to the maximum allowed waiting time of a trip request as this means that a vehicle situated in i could reach the request on time. Thus, N_i is defined as $N_i = \{j \in A | t_{ij} \leq t^c\}$. Figure 4 shows an example in the city of Hamburg with a coverage radius of 480 seconds. Given these variable definitions,

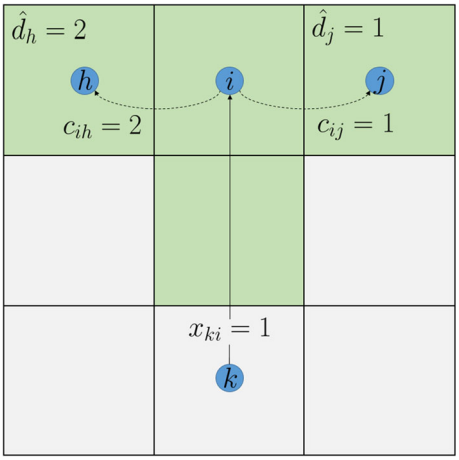

Fig. 5 Illustration of decision variables. Green areas form the neighborhood of i. Only non-zero demands and relevant area centers are shown for simplicity. In this example one vehicle is repositioned from area k to area i to cover forecasted demand in areas j and h

our model will reposition vehicles to provide coverage in areas with lacking supply. Figure 5 shows a simple example in which one vehicle is repositioned from area k to area i in order to cover demand in the two neighboring areas h and j. The precise number of requests that a single vehicle may cover depends on its location and the time of day. Therefore, we adaptively determine a parameter $\hat{e}_i$ denoting the expected number of trip requests that a vehicle located in area i may cover over the horizon h. The details of this supply calculation as well as the adaptive parameter tuning technique will be explained later in this section.

Model The objective function (1) follows three hierarchical goals which are reflected in the terms of the objective function:

1. Maximize the sum of covered demand, weighted by w^{cov} and w_i.
2. Minimize the number of repositioning movements, weighted by weight w^{mov}.
3. Minimize travel times for repositioning movements and demand coverage. The latter are penalized with weight w^t.

Objective precedence is ensured by weights $w^{cov} > w^{mov} > 1$. These weights may be determined based on the maximum overall travel time between two areas T^{max}. We use $w^{cov} = 10 \cdot T^{max}$ and $w^{mov} = T^{max}$. With this choice of weights, we ensure that one unit of additional covered demand is prioritized over minimizing movements and travel times. The primary objective is to maximize the acceptance rate of future requests by covering predicted demand. Empirically, it has proven beneficial to prioritize coverage in high-demand areas. Therefore, we add weights per area corresponding to the fraction of total estimated demand, i.e., $w_i = \frac{\hat{d}_i}{\sum_{i \in A} \hat{d}_i} = \frac{\hat{d}_i}{\hat{D}}$ rewarding coverage in high-demand areas. The secondary objective stems from the operational concern that we want to move as few vehicles as possible. Particularly, we do not want to move any vehicles at all, if the current fleet configuration can cover all forecasted demand. Thus, we penalize the movement of vehicles and ensure that repositioning only takes place if it leads to additional covered demand. The tertiary objective ensures that overall travel times are minimized and leads to suitable vehicles being selected for repositioning. Two travel time factors are taken into account. First, we consider travel times incurred by repositioning decisions x_{ij}. Second, we consider anticipated travel times attached to c_{ij} variables. The assumption is that a vehicle located at $i \in A$ will have to move to $j \in A$ when a request arises. These anticipated travel times are penalized by a factor $w^t \geq 1$ which rewards moving vehicles closer to the predicted demand. This tends to be

beneficial as it reduces customer waiting times and improves vehicle utilization.

$$\text{(FDR-M)} \quad \sum_{i \in A} \sum_{j \in A} w^{cov} \cdot w_j \cdot c_{ij} - \sum_{i \in A} \sum_{j \in A \setminus \{i\}} w^{mov} \cdot x_{ij} - \sum_{i \in A} \sum_{j \in A} x_{ij} \cdot t_{ij} - \sum_{i \in A} \sum_{j \in A} w^{t} \cdot c_{ij} \cdot t_{ij} \rightarrow \max \tag{1}$$

$$\text{s.t.} \quad \sum_{j \in A} x_{ij} \leq |K_i^{id}| \qquad i \in A \tag{2}$$

$$\sum_{j \in A} c_{ji} \leq \hat{d}_i \qquad i \in A \tag{3}$$

$$\sum_{j \in A} c_{ij} \leq \sum_{j \in A} x_{ji} \cdot \hat{e}_i + S_i^{re} + S_i^{act} \qquad i \in A \tag{4}$$

$$c_{ij} = 0 \qquad i \in A, j \notin N_i \tag{5}$$

$$x_{ij} = 0 \qquad i \in A, j \notin A^{re}, i \neq j \tag{6}$$

$$x_{ij} \in \mathbb{N}_0^+ \qquad i, j \in A \tag{7}$$

$$c_{ij} \in \mathbb{R}_0^+ \qquad i, j \in A \tag{8}$$

Constraints (2) guarantee that the number of vehicles repositioned from $i \in A$ does not exceed the number of available idle vehicles. Note that only idle vehicles are available for repositioning. Vehicles that are currently performing a repositioning movement cannot be reassigned to a new target. This is due to operational concerns as we do not want to frequently change repositioning targets of individual vehicles and drivers. A repositioning movement may however be interrupted by the dispatching algorithm as presented in Sect. 3.1. In that case, the vehicle is assigned a route with trip requests and starts serving these. Constraints (3) ensure that the maximum provided coverage for a given area i is capped by the forecasted demand $\hat{d}_i$. Inversely, Constraints (4) limit the provided coverage from area i to the available supply. This supply is equivalent to the number of trip requests that may be served within the forecast horizon h by vehicle resources in i. In order to calculate the available supply, we consider three types of vehicles. Active vehicles located in i contribute to the active supply S_i^{act}, while vehicles currently repositioning to i provide the repositioning supply S_i^{re}. Lastly, supply provided by idle vehicles is considered through the x_{ji} variables in the right-hand side of the equation. This term includes vehicles staying idle at i as well as vehicles being selected for repositioning to i. These vehicles are multiplied by a factor $\hat{e}_i$ denoting the expected number of trip requests that a vehicle in i will serve within the forecast horizon. This parameter as well as the active and repositioning supply may differ significantly depending on the considered area, current vehicle schedules, and time of day among other factors. Therefore, we introduce an adaptive parameter tuning approach, which will be detailed in Sect. 4.1.4. Constraints (5) limit the spatial extent of provided coverage to the given neighborhood N_i as explained earlier. As repositioning is only allowed to a set of target areas A^{re}, Constraints (6) ensure that we only send idle vehicles to such areas or leave them at their current location. Lastly, variable domains are given by Constraints (7) and (8).

4.1.4 Adaptive parameter tuning

As mentioned previously, the number of trip requests that a vehicle is expected to serve in the horizon h varies depending on several factors such as its current location, the time of day, or its current schedule. For instance, vehicles tend to be able to serve more requests in high-demand areas as it is possible to build more efficient vehicle routes. To reflect these facts, we propose an adaptive parameter tuning process to estimate the supply provided by the vehicle fleet over the next horizon h^+.

For this purpose, we consider the performance of our vehicle fleet within the previous horizon h^-. As a first step given in Equation 9, we determine the potential number of requests that a vehicle k could have served in h^-, denoted as e_k.

$$e_k = 0.9 \cdot \frac{1}{\alpha_k} \cdot \frac{p_k^- + d_k^-}{2} \quad k \in K \tag{9}$$

p_k^- and d_k^- denote the number of pickup and delivery operations performed by k in h^-. Thus, dividing the sum of these values by two in the right-hand side of the equation gives us the number of trip requests served in h^-. However, k may have been idle for a portion of h^-, therefore this number does not reflect the potential number of served requests at full utilization. To account for this fact we multiply by a factor of $\frac{1}{\alpha_k}$ where α_k is the percentage of time that k was active in h^-. This value tends to be an upper bound on the number of trip requests that the vehicle could have realistically served. In general, a 100 % usage rate of vehicles is not achievable in practical scenarios. Based on preliminary studies, we estimate that an average 90 % usage rate is realistic and therefore correct our result by a factor of 0.9.

Based on this historical vehicle performance, we may now estimate the expected number of served trip requests $\hat{e}_i$ for each area i as defined in Equation 10. This value yields an estimation of how many trip requests a vehicle starting in area i will serve in the time period h^+.

$$\hat{e}_i = \frac{\sum_{k \in K^{N_i}} e_k}{|K^{N_i}|} \quad i \in A \tag{10}$$

To calculate $\hat{e}_i$, we consider all vehicles K^{N_i} that were located in the neighborhood of i at the start of the previous horizon h^-. Recall that this neighborhood is defined as the set of areas within the coverage radius of i. To achieve a reliable estimate, we furthermore ensure that K^{N_i} contains at least k^{min} vehicles. If this is not the case, the neighborhood N_i is grown iteratively by including the next closest area until k^{min} vehicles are reached. We may then calculate $\hat{e}_i$ as the average of e_k for all vehicles in K^{N_i}.

Based on $\hat{e}_i$, we are now able to estimate our available supply in any area. Supply provided by repositioning vehicles, i.e. vehicles that have been selected for repositioning in previous runs of the repositioning algorithms, is calculated as defined in Equation 11. We assume that each vehicle repositioning to area i will serve approximately $\hat{e}_i$ requests over the coming forecast horizon h^+. Thus, we merely multiply the number of repositioning vehicles by $\hat{e}_i$.

$$S_i^{re} = |K^{re}| \cdot \hat{e}_i \quad i \in A \tag{11}$$

Supply provided by active vehicle is calculated in a similar fashion in Equation 12. However, in this case we need to consider the current vehicle schedules. Thus, for each vehicle, we adjust the number of expected served trip requests based on the current planned pickup

(p_k^+) and delivery (d_k^+) operations.

$$S_i^{act} = \sum_{k \in K_i^{act}} \hat{e}_i - \frac{p_k^+ + d_k^+}{2} \quad a \in A \tag{12}$$

Finally, supply is also provided by idle vehicles that are either selected to reposition to another location in the subsequent period or that stay at their current position. This is expressed through the decision variables x_{ji} in Constraints (4) of our model.

4.2 Reactive repositioning algorithm (REACT)

As a benchmark algorithm, we implement a reactive repositioning approach. The algorithm is an adapted version of the repositioning approach presented by Alonso-Mora et al. (2017a). The central idea behind it is to reposition idle vehicles to the locations of rejected trip requests. This tends to be beneficial as trip requests are highly spatially and temporally correlated. Therefore, it is likely that additional trip requests will arise in the near future in the vicinity of rejected requests. We modify the proposed algorithm to reflect the fact that we process each trip request individually upon arrival whereas Alonso-Mora et al. (2017a) work with batches of requests gathered over a given batching time period. Thus, after rejecting a trip request, we may immediately select an idle vehicle for repositioning. Given a rejected request r and its pickup location p_r, we greedily reposition the nearest idle vehicle to p_r, i.e. the vehicle with the shortest travel time from its current position to p_r.

5 Computational evaluation

In this section, we evaluate FDR on several real-world datasets. We first discuss our experimental design and afterward our datasets and describe the specific scenarios considered in this computational study. Subsequently, we present algorithmic settings as well as relevant performance indicators. Finally, Sect. 5.6 presents the computational results and findings.

5.1 Experimental design and setup

In this study, we investigate the performance of our approach on several real-world instances. Our main goals are (1) to evaluate the applicability of the repositioning approach for large-scale instances, (2) to compare its performance to the reactive repositioning approach of Alonso-Mora et al. (2017a), (3) to assess the robustness of the repositioning approach relative to errors in the forecasting process, and (4) to evaluate how well the adaptive method performs in different settings and scenarios.

To assess the performance of the algorithms under circumstances, we use four real-world data sets for urban and metropolitan areas that differ substantially in terms of structure. These data sets as well as the scenarios derived from them are introduced in Section 5.2 in more detail. Note that our main interest lies in the evaluation of the repositioning algorithm rather than in the development of demand prediction methods. However, the quality of the repositioning decisions depends on the quality of the forecasts that are used. Therefore, we combine our algorithm with both a naive and a perfect demand forecast, which, together, provide reasonable lower and upper bounds on the quality of forecasts that can be expected in

practical applications. These forecast modes are discussed in Sect. 5.4. Finally, performance measures are defined in Sect. 5.5.

Our planning algorithms as well as the discrete-event simulation described in Sect. 3 are implemented in C++. As a routing engine, we use RoutingKit (Dibbelt et al. 2016) which operates on data extracted from OpenStreetMap covering the respective areas under study. Gurobi 8.1.0 serves as a MIP solver for model FDR-M. All experiments were run on the same machine with an Intel i7-6600U CPU and 20 GB of RAM.

5.2 Datasets and scenarios

In the following we describe the datasets that were used in this study (Sect. 5.2.1). From these datasets we derive several simulation scenarios that are described in Sect. 5.2.2.

5.2.1 Dataset description

We evaluate our repositioning approach on four real-world datasets. Firstly, a dataset containing taxi trip records from the city of Hamburg (HH)[1]. Secondly, we use the publicly available data of taxi trips in New York City[2] to build two datasets. On the one hand, we consider the complete area of New York City (NYC). Additionally, we also build another dataset as a subset of the previous one, containing only taxi trips within Manhattan (MANH). We do this to evaluate our algorithm under different circumstances. The MANH dataset reflects a highly urbanized area with very dense demand in a rather small geographical region while the NYC dataset also includes less densely populated regions and covers a significantly larger area. Lastly, we utilize a dataset from Chengdu (CH)[3] made available by the Chinese MOD provider Didi Chuxing.

All of these datasets have the same basic structure and contain the pickup time as well as the desired pickup and delivery coordinates of each trip request. Moreover, the NYC and MANH datasets also contain the number of passengers per request. As this information is missing from the other two datasets, we derive a distribution over the number of passengers based on the NYC dataset and randomly sample from this distribution for the HH and CH datasets. Additionally, we perform some basic filtering to eliminate obvious outliers and erroneous records. For this purpose, we remove records where pickup or delivery coordinates are missing or lie outside the respective region of study. We only consider trip requests with a passenger count of at most two. This is common practice in ride-sharing services like UberPool (Uber 2021) as larger groups are often requested to use classical taxi services.

5.2.2 Scenario generation

Based on the datasets described above, we generate a set of scenarios. As the spatial and temporal distribution of demand varies significantly between working days and weekends, we consider a scenario on a Wednesday as well as Sunday for each dataset. This way, we evaluate our repositioning approach under different demand patterns. The precise dates and number of requests are shown in Table 5. We use a shorthand notation to denote our scenarios, e.g., "HH-Wed" corresponds to the Wednesday scenario for the HH dataset. One thing to

[1] Provided by PTV Group, Haid-und-Neu-Str. 15, 76131 Karlsruhe, Germany.

[2] https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page.

[3] https://outreach.didichuxing.com/appEn-vue/KDD_CUP_2020.

Table 5 Temporal scenario settings per dataset with dates and trip requests

	HH		NYC	
	Date	Requests	Date	Requests
Wednesday	20 Mar 2019	13,556	16 Mar 2016	376,526
Sunday	24 Mar 2019	10,669	20 Mar 2016	368,508
	MANH		CH	
	Date	Requests	Date	Requests
Wednesday	16 Mar 2016	297,457	16 Nov 2016	239,037
Sunday	20 Mar 2016	269,346	20 Nov 2016	237,037

Table 6 Fleet size per dataset

	HH	NYC	MANH	CH
Fleet size	90	1300	900	1700

note is that we use a simulation warm-up time of 6 hours in simulated time. Thus, when evaluating a scenario on 20 March 2019, the simulation is started on 19 March 2019 at 18:00, but the collection of statistics is only performed after 20 March 2019 at 00:00. This way, we do not start with an empty system state in which all vehicles are idle.

As we have no information regarding real-world vehicle fleets, we use preliminary tests to determine a fleet size for each dataset as given in Table 6.

This fleet size should lead to a trip request acceptance rate of roughly $90 - 95$ % with our benchmark algorithm REACT and is used throughout most experiments in the following sections. The reasoning behind our fleet size choice is as follows. Firstly, it seems reasonable to assume that a real-world provider of a MOD service would be able to accept almost all trip requests. We allow for some rejections due to trip requests that are very difficult to reach or due to excess demand during peak hours. Secondly, we vary fleet sizes in a limited range around our base size to evaluate scenarios with a slight over- and undersupply of vehicles in order to assess the algorithm performance under these conditions (Section 5.6.3). Scenarios with extremely small or large fleet sizes relative to the given demand are less interesting for repositioning. In these cases either no repositioning is necessary or all vehicles are occupied and, thus, no idle vehicles are available for repositioning.

5.3 Repositioning settings

All scenarios are run with different repositioning modes: no repositioning at all (NONE), reactive repositioning as described in Sect. 4.2 (REACT), and our forecast-driven repositioning algorithm from Sect. 4.1 (FDR). FDR has a set of parameters that need to be configured appropriately. Concerning the repositioning interval (f), grid cell size (g) and forecast horizon (h), we perform experiments to assess the impact of these parameters and determine suitable values. These are described in detail in Sect. 5.6.1. Values for all remaining parameters are given in Table 7.

The objective function weights are selected in accordance with the prioritization of the different goals as discussed in Sect. 4.1.3. We use $w^{cov} = 10 \cdot T^{max}$ and $w^{mov} = T^{max}$. Covered demand is weighted with a factor of $w_i = \frac{\hat{d}_i}{\hat{D}}$, while travel times for coverage

Table 7 FDR parameter settings

Description	Values
Objective weight for total coverage (w^{cov})	$10 \cdot T^{max}$
Objective weight for vehicle movements (w^{mov})	T^{max}
Objective weight for coverage travel time (w^{t})	1.3
Objective weight for areas (w_i)	$\frac{\hat{d}_i}{\hat{D}}$
Min. vehicles in neighborhoods (k^{min})	5
Coverage radius (t^{c}) [min]	4 / 8 (HH)

are penalized by a factor of $w^t = 1.3$. Altogether, these weights prioritize the coverage of additional demand over the minimization of vehicle movements. If not all demand may be covered, this parameter setup prioritizes trip requests in high-demand areas. Lastly, travel times for coverage of demand are penalized slightly, encouraging moving vehicles closer to the covered demand. When running our adaptive parameter tuning process, we set the minimum number of considered vehicles for calculation of $\hat{e}_i$ to $k^{min} = 5$. One additional parameter is configured depending on the dataset: the coverage radius t^c. It is set to 8 minutes for the HH dataset and 4 minutes for all other datasets. Recall that this parameter is derived from the maximum waiting time of a customer. We assume a longer acceptable waiting time for the Hamburg dataset where trip request demand is relatively sparse and the vehicle fleet is consequently much smaller compared to the other datasets. Otherwise, we would need an excessively large vehicle fleet in the Hamburg scenarios to serve a reasonable fraction of the arising trip requests.

5.4 Forecast modes

As described above, we combine the forecast-driven repositioning algorithm (FDR) with two different forecasting modes to evaluate its performance relative to the quality of the predicted information. In order to base our study on forecasting benchmarks that can be easily replicated, we do not use dedicated forecasting methods but seek to represent the margin of error that may be associated with real-world forecasts. We first employ a perfect forecast that assumes complete knowledge of future trip requests. It is used in this study to examine how well the proposed method can exploit the available information, since it includes, for example, demand spikes due to public events. The corresponding setting is denoted as FDR (P) in the following. Secondly, we use a naive forecast that assumes that the demand in the next period equals that of the previous one, i.e. the forecast for a given area i and a forecast horizon h is merely the number of trip requests that originated in i within the previous horizon h^-. This setting is denoted as FDR (N). It is reasonable to assume that any forecast achieves at least this quality in practice, as mechanisms that perform worse can easily be replaced by the naive forecast. Similar to REACT, it can adjust to demand surges with a small delay. Moreover, it is a method that can be used without any additional data input and can therefore also be used for new systems without a data history.

These two forecasts serve as an upper and lower bound on realistic state-of-the-art forecasting models and can be easily replaced by application-specific methods. They serve to illustrate that our algorithm is relatively robust to forecasting errors and may already be successfully utilized with a relatively simple forecast. We performed some preliminary tests

Table 8 Performance indicators

KPI	Unit	Description
Rej	%	Trip request rejection rate
Wait	s	Avg. customer waiting time
Ride	s	Avg. customer ride time
TT^v	min	Avg. total travel time per vehicle
TT^v_{rep}	min	Avg. repositioning travel time per vehicle
TT^v_{req}	s	Avg. vehicle travel time per served trip request
RT	min	Total running time
RT_r	min	Total running time for the repositioning algorithm

regarding the performance of our naive forecast in order to compare its performance to results from literature and estimate how a more refined forecasting model would perform. In particular, we replicated the evaluation of Yao et al. (2019) on the NYC dataset. They achieve a root-mean-squared error (RMSE) of 24.10 with their best model and report a RMSE of $26.07 - 28.51$ with classical machine learning approaches such as gradient boosting or multi-layer perceptrons. In comparison, our naive forecast achieves a RMSE of 35.54. For our experiments, we used the same spatial granularity (1x1 km grid cells), test data period (11 February 2015 - 03 March 2015), and forecast interval of 30 minutes. Therefore, these results should be roughly comparable despite minor differences like the data preprocessing and cleaning procedure. Based on these evaluations, we may conclude that our naive and perfect forecasts establish a reasonable range regarding the performance of our repositioning approach.

5.5 Performance indicators

To assess the performance of our repositioning algorithm, we introduce a set of key performance indicators (KPI) as summarized in Table 8. The trip request rejection rate is our primary measure for the effectiveness of our repositioning algorithms. Customer waiting and ride times are used as a means to assess the impact on customer satisfaction. The total travel time per vehicle TT^v serves as an estimator of the increased system cost by vehicle movements. However, it is distorted due to the differences in served trip requests. An increased average travel time may be caused by two main factors: First, a higher vehicle utilization due to serving more trip requests, and second, the repositioning movements themselves. Thus, we also consider the average travel time per served request TT^v_{req} which corresponds to the total travel time of all vehicles divided by the total number of served trip requests. We also separately measure the travel time for repositioning movements TT^v_{rep}. Lastly, we report the total running time of a scenario. This includes the running time of the repositioning algorithm as well as dispatching and simulation. The running time of only the repositioning algorithm is also reported separately as RT_r.

5.6 Computational results

Throughout the following sections we present our main computational results and findings. First, we study the impact of several algorithmic parameters in Section 5.6.1. Subsequently,

Table 9 FDR parameter evaluation, default values used in this section are denoted in *italic* while the best values determined in our tests are denoted in **bold**

Description	Values
Repositioning interval (f) [s]	**30**, 60, *120*, 180, 300
Grid cell size (g) [m]	750, 1000, *2000*, **3000**, 5000
Forecast horizon (h) [min]	5, 15, *30*, **60**, 120

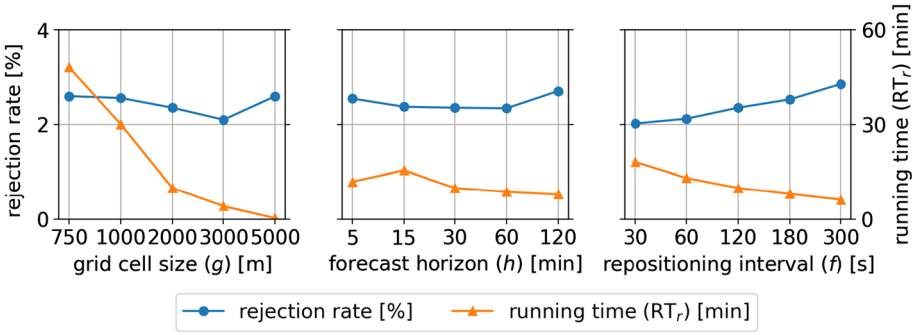

Fig. 6 Influence of the grid cell size (g), forecast horizon (h) and repositioning interval (f) on the trip request rejection rate and the total running time of FDR

Section 5.6.2 presents an overview of our results while Sections 5.6.3 - 5.6.6 present a more detailed analysis of several relevant factors such as fleet sizes, the parameter tuning process and the influence of the demand forecast.

5.6.1 Parameter influence

In this section, we study the influence of three main parameters of FDR: the grid cell size (g), the repositioning interval (f), and the forecast horizon (h). For each parameter, we test five different settings as given in Table 9.

Due to the involved computational effort, we are unable to perform a full grid search for all possible parameter combinations. Hence, we vary each parameter individually, setting the other two parameters to their default values indicated in italic in Table 9. The best found values for each parameter that will be used for the remainder of this study are highlighted in bold. We test each setting on the scenarios as described in Section 5.2.2. We mainly focus on the impact of each parameter on the overall trip request rejection rate as well as the running time for FDR. Average results across all scenarios and datasets are depicted in Figure 6. We show the trip request rejection rate as well as the total running time of FDR (RT_r) for each parameter value.

The grid cell size has a large impact on the total running time of FDR. This is mainly due to the fact that the size of our mathematical model FDR-M scales with the number of grid cells. On the other hand, the effect on the trip request rejection rates is minor. Empirically, the best rejection rates are achieved at a size of $g = 3000$ m. Therefore, we will use this value in subsequent sections. At his grid cells size, the running times of FDR are negligible with one iteration taking an average of around 80 ms.

Concerning the forecast horizon, the performance of FDR is relatively stable for values between 15 and 60 minutes. Above or below these values we see an increase in rejected trip requests. This meets our expectations as a very short horizon leaves little time to react

Table 10 Average results for each combination of dataset and repositioning mode

Data	Mode	Rej [%]	Wait [s]	Ride [s]	TT^v [min]	TT^v_{req} [s]	RT [min]	RT_r [min]
HH	NONE	15.44	405.58	530.87	773.83	409.54	1.06	0.00
	REACT	6.22	387.43	529.58	876.42	417.56	1.37	0.00
	FDR (P)	2.50	334.27	525.78	960.43	439.06	4.99	1.47
	FDR (N)	**2.33**	**328.22**	**525.17**	959.04	437.85	3.93	1.53
NYC	NONE	73.98	237.67	477.46	311.23	251.04	65.49	0.00
	REACT	6.98	221.96	426.94	1009.48	227.30	68.63	1.93
	FDR (P)	4.07	**199.80**	**414.99**	1052.42	229.78	86.86	4.83
	FDR (N)	**4.00**	199.99	415.77	1051.97	229.50	84.03	3.95
MANH	NONE	34.63	248.03	368.81	567.14	165.40	50.98	0.00
	REACT	2.25	228.28	353.35	842.99	164.54	34.16	0.24
	FDR (P)	**0.33**	204.17	**342.38**	868.82	166.35	85.42	1.00
	FDR (N)	0.38	**204.11**	342.39	864.20	165.56	137.37	0.90
CH	NONE	45.65	208.89	679.41	436.13	363.95	30.05	0.00
	REACT	6.25	195.11	629.08	741.98	358.56	25.62	1.82
	FDR (P)	**0.91**	174.89	617.67	825.63	377.48	107.13	7.84
	FDR (N)	1.00	**174.49**	**617.31**	819.18	374.90	103.44	6.41
ALL	NONE	42.42	275.04	514.14	522.08	297.48	36.89	0.00
	REACT	5.43	258.20	484.74	867.72	291.99	32.44	1.00
	FDR (P)	1.95	228.28	475.21	926.83	303.17	71.10	3.78
	FDR (N)	**1.93**	**226.70**	**475.16**	923.59	301.95	82.19	3.20

The last rows denoted as "ALL" contain averages across all four datasets. For KPIs concerning the customer experience, the best values for each dataset are indicated in bold

to changing demand and a long horizon includes forecasted demand that is not relevant for the short-term decision making. For all subsequent evaluations, we select the best value of $h = 60$ min.

Lastly, the repositioning interval has a relatively strong impact on the rejection rate as well as the running time. As expected, shorter intervals lead to fewer rejected trip requests at the cost of an increase in running time as FDR is run more often. However, even running FDR every 30 seconds is still manageable as the average running time for one iteration is merely around 80 ms. Hence, subsequently we select an interval of $f = 30$ sec.

5.6.2 Results overview

Table 10 provides an overview of the computational results. It contains average values for each combination of dataset and repositioning mode. Additionally, we also calculate averages across all datasets, denoted in the table as "ALL".

Concerning the customer rejection rate, FDR achieves the best results with an average improvement of around 3.5 percentage points compared to our benchmark algorithm REACT. One thing to note is that the scenarios without repositioning (NONE) are not competitive and exhibit high rejection rates. Therefore, we will not discuss these values in detail and will generally use algorithm REACT as a baseline for comparison. When comparing FDR (P) and FDR (N), we observe similar rejection rates. FDR (N) even performs better in several

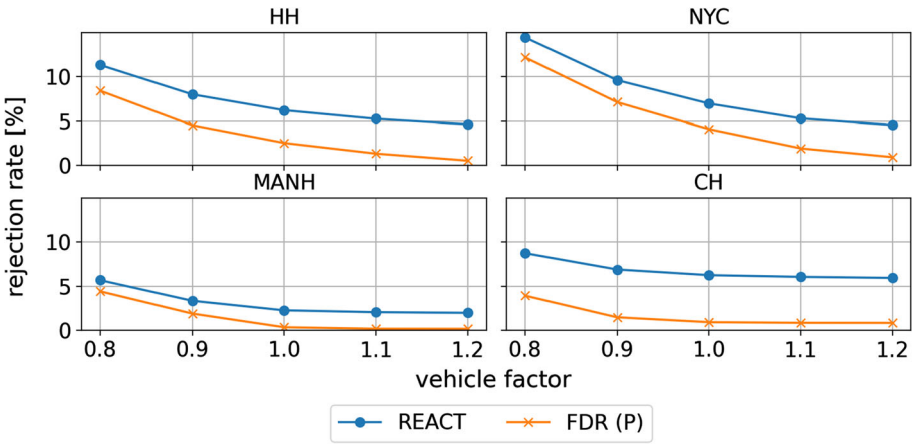

Fig. 7 Impact of the fleet size on rejection rates with REACT and FDR (P)

scenarios. As this behavior may also be observed for the other KPIs, we will focus our analysis on FDR (P) for most of this section. In Sect. 5.6.5 we will take a more detailed look at the differences between the two demand forecasting modes.

In addition to lowering rejection rates, our forecast-driven repositioning algorithm also leads to reduced customer waiting and ride times. Throughout all scenarios, it achieves a significant reduction compared to REACT. On average the reduction in waiting time amounts to around 30 s (11.6 %). This is a side-effect of repositioning as vehicles are better positioned to serve arising requests. The travel time per vehicle is increased when using repositioning mechanisms. However, this is to be expected as the number of served requests rises. As the travel time per served request illustrates, algorithm FDR only leads to slightly worse results than REACT.

Overall, these results show that our approach does not perform unnecessary repositioning movements in most cases. The observed running times are suitable for real-time usage across all datasets. Repositioning itself only takes up a minor fraction of the running time compared to the computational effort for dispatching and simulation. On average, one iteration of FDR takes 79 ms and even on the most challenging dataset (CH), this value only goes up to 163 ms. Hence, the algorithm may be used in real-time even on large datasets. The fast running times of the repositioning algorithm are important to prevent conflicts with the dispatching algorithm. If the repositioning algorithm would take too long, the system state would deviate while it is running due to incoming trip requests. This would potentially invalidate the decisions made by the repositioning algorithm. The overall running time is in all cases lower than the simulated time equivalent of 1440 min. Hence, FDR can be applied in real-time on large-scale datasets. In fact, there is still a large gap between the total running times and the simulated time equivalent. This available computational time could for instance be used to employ more complex vehicle routing algorithms.

5.6.3 Impact of fleet sizes

In Fig. 7 we illustrate the impact of the fleet size on the rejection rate with algorithms REACT and FDR (P).

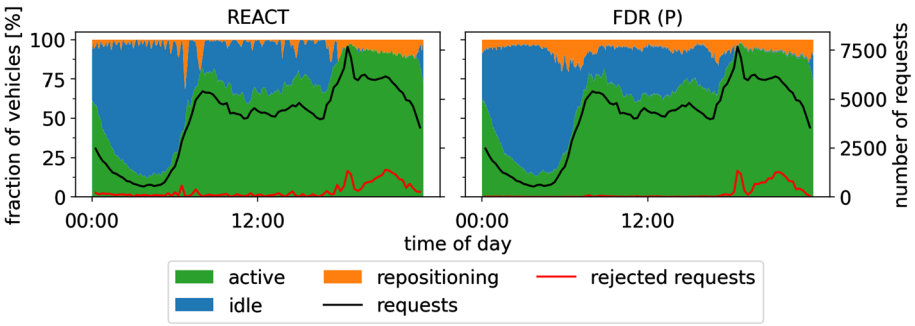

Fig. 8 Vehicle utilization throughout the day for scenario NYC-Wed with REACT and FDR (P)

For this purpose, we alter the base fleet size as determined in Sect. 5.2.2 by a fixed vehicle factor ranging from 0.8 to 1.2. The actual fleet size is obtained by multiplying the vehicle factor with the fleet size. Across all datasets and fleet sizes, FDR improves the rejection rate compared to REACT by an average of 3.36 percentage points. The level of improvement varies between 1.27 and 5.44 percentage points depending on the specific dataset and fleet size. For instance, in the MANH dataset REACT performs remarkably well and we see only minor improvements by employing FDR. We assume that this is mainly due to the geographical concentration of trip requests in the downtown Manhattan area and the small overall area. This makes repositioning less impactful as vehicles are always relatively close to high-demand areas. For all datasets, we observe some level of diminishing returns as the fleet size is increased. This effect is most noticeable on the MANH and CH datasets where we see only very minor reductions in the rejection rate when raising the vehicle factor above 1.0. Another trend across all datasets is that the difference between FDR and REACT grows as the number of vehicles is increased. FDR is better suited to exploit larger fleet sizes where almost all trip requests may be served, even ones in remote areas. In the case of small vehicle fleets, the complete fleet may be occupied during peak hours, therefore leaving little room for improvement by repositioning.

5.6.4 Vehicle utilization & impact of datasets, weekdays and fleet sizes

Besides aggregated KPIs, it is also interesting to take a detailed look at the utilization of vehicles throughout the day for different scenarios.

Figure 8 shows the vehicle utilization compared between the different repositioning modes for one scenario. Colored areas illustrate the fraction of vehicles in a specific state over time. Possible states are idle, active, and repositioning. Lines indicate the number of total and rejected requests over time.

With FDR (P), most of the fleet is left idle during low-demand times and only minimal repositioning is performed (particularly at night between 02:00 and 04:00). Before the morning peak, a significant portion of the fleet is repositioned. In comparison, REACT only starts to reposition notable numbers of vehicles after a spike in rejected requests at around 07:00. Overall, when using FDR, the number of rejected requests is almost zero throughout most of the day. Only during the evening peak after approximately 18:00, when the complete fleet is occupied, requests are rejected. REACT on the other hand exhibits a small number of rejected trip requests even at times when the vehicle fleet is not fully utilized.

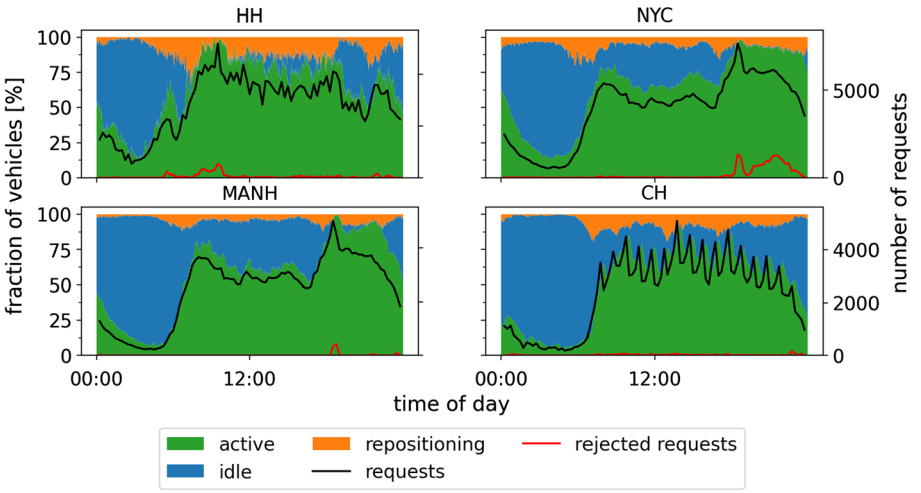

Fig. 9 Vehicle utilization throughout the day for scenarios HH-Wed, NYC-Wed, MANH-Wed and CH-Wed with FDR (P)

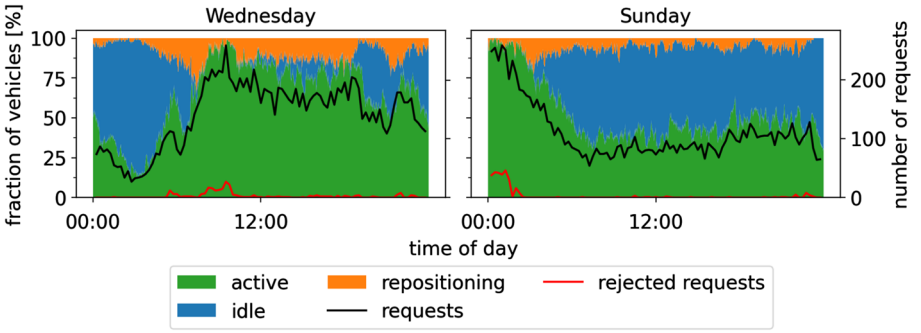

Fig. 10 Vehicle utilization throughout the day for scenarios HH-Wed and HH-Sun with FDR (P)

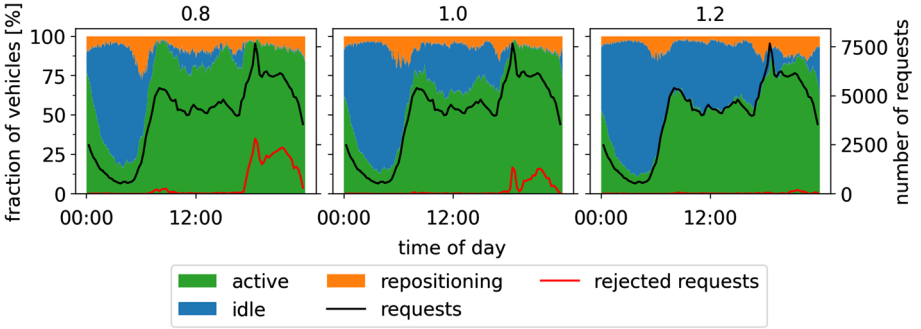

Fig. 11 Vehicle utilization throughout the day for scenario NYC-Wed, FDR (P) and vehicle fleet factors 0.8, 1.0 and 1.2

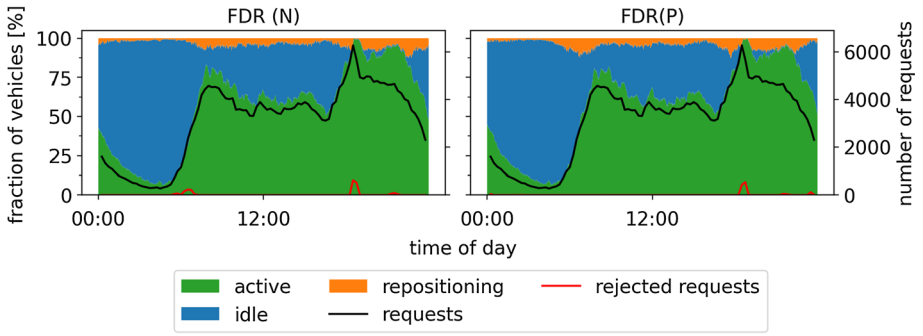

Fig. 12 Vehicle utilization throughout the day compared for one scenario MANH-Wed and two repositioning modes FDR (N) and FDR (P)

The behavior of FDR (P) is consistent and robust over a wide variety of scenarios. Figure 9 illustrates the vehicle utilization on our four different datasets. Although the demand patterns are different, the algorithm achieves reliable results in all scenarios. The same may be concluded for the behavior when comparing scenarios HH-Wed and HH-Sun as seen in Fig. 10. Although the temporal development of demand is completely different in both scenarios, FDR manages to keep rejection rates low and only rejects a larger number of trip requests in times where the complete vehicle fleet is utilized. Finally, FDR (P) also works consistently with different fleet sizes as shown in Fig. 11. Repositioning becomes less impactful in the scenario with a 0.8 vehicle fleet factor, as the complete fleet is occupied throughout large portions of the day and only a few vehicles are available for repositioning. Conversely, in the scenario with a 1.2 vehicle fleet factor, FDR (P) manages to serve almost all trip requests.

5.6.5 Comparison of naive and perfect forecasts

In this section, we take a detailed look at the performance differences of FDR when comparing the perfect and naive forecasting models, thereby assessing the robustness of the algorithm with respect to forecasting errors. As we saw from the aggregated results in Sect. 5.6.2, the overall difference between the two forecasting modes is relatively small. Somewhat surprisingly, FDR (N) even led to better rejection rates in several scenarios. Looking at the vehicle utilization in Fig. 12, we can see some minor differences between the two modes.

FDR (N) underestimates demand in the morning, leading to some additional rejected requests. On the other hand, it overestimates demand in the evening, incurring additional repositioning movements. The reason for these differences is the structure of the naive forecast, which assumes that the demand over the next forecast horizon is equal to the demand observed throughout the last forecast horizon. Hence, when we observe a rising demand, for instance during the morning rush hour, the naive forecast lags behind and only catches the peak with some delay. On the other hand, during times of sharply decreasing demand, such as in the late evening, the naive forecast overestimates the level of demand and causes additional repositioning movements.

To illustrate this effect more concisely, we generate two additional scenarios for the HH and MANH datasets. The time slices selected for these scenarios are depicted in Fig. 13. One scenario covers the morning rush hour while the other covers the evening hours during which the level of demand decreases. The results for these scenarios are summarized in Table 11.

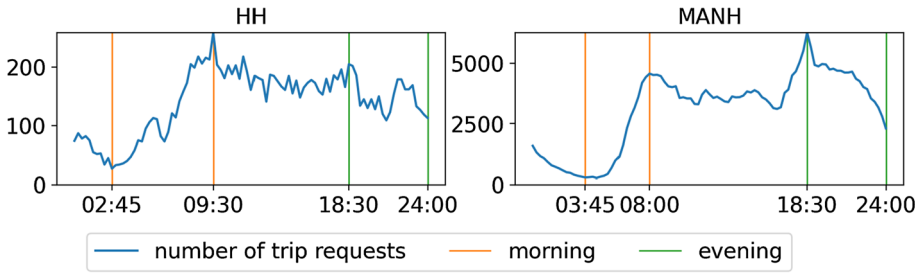

Fig. 13 Time slices used to illustrate performance during changes in demand

Table 11 Average results for the morning and evening scenario with a FDR (P) and FDR (N)

Scenario	Data	Mode	Rej [%]	TT^v [min]	TT^v_{rep} [min]
morning	HH	FDR (P)	3.57	265.07	43.77
		FDR (N)	4.71	254.64	33.97
	MANH	FDR (P)	0.00	89.53	15.11
		FDR (N)	2.30	80.96	6.05
evening	HH	FDR (P)	0.82	242.64	27.96
		FDR (N)	0.52	249.89	34.79
	MANH	FDR (P)	1.03	302.45	20.82
		FDR (N)	1.30	304.80	24.77

In the morning scenario, our naive forecast underestimates the trip request demand and causes less repositioning movements as seen by the repositioning time TT^v_{rep}. However, this also leads to an increased number of rejected trip requests for both datasets. Conversely, in the evening scenario, FDR (N) overestimates the demand level and repositions additional vehicles. In the case of the HH dataset, this leads to an improvement in rejected trip requests. The reasoning for this is that FDR aims to minimize repositioning movements. In some cases, we may overestimate the capability of the vehicle fleet to serve the forecasted demand. Hence, repositioning additional vehicles can be beneficial.

Overall, we can conclude that FDR can be successfully utilized even when combined with structurally biased demand forecasts. Assuming that our two forecasting modes provide reasonable bounds on demand forecasts used in practice (see Sect. 5.4), we expect that the effects observed in practical applications follow a similar pattern.

5.6.6 Impact of adaptive parameter tuning

In Sect. 4.1.4 we introduced an adaptive parameter tuning process with the goal of estimating the number of trip requests that a specific vehicle may serve within the next forecast horizon. This approach serves two main purposes. Firstly, it is intended to remove the necessity of tuning parameters a priori. Hence, we could start a ride-sharing service in a new area of operations without needing any data in advance to tune our system parameters. Secondly, it should more accurately reflect spatial and temporal differences in the number of trip requests that a vehicle may serve compared to using a single static parameter. This should in turn lead to improved system performance.

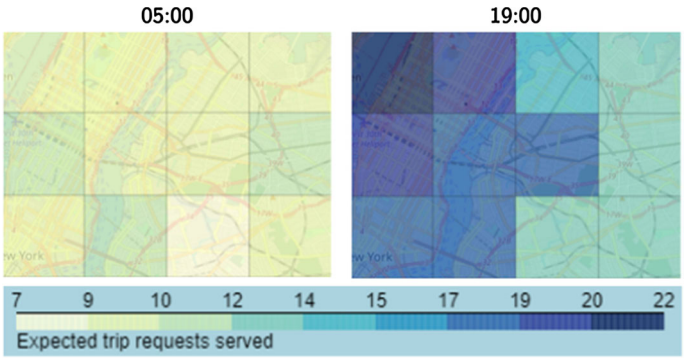

Fig. 14 Expected requests served ($\hat{e}_i$) for an excerpt of NYC (parts of Manhattan and Brooklyn) on 17 Mar 2016 at 05:00 and 19:00

Table 12 Average results with and without adaptive parameter tuning for each dataset. The last rows denoted as "All" contain averages across all four datasets

Data	Adaptive	Rej [%]	TT^v [min]
HH	yes	1.87	1077.11
	no	2.17	1043.66
NYC	yes	4.62	1011.16
	no	5.24	994.14
MANH	yes	0.43	884.85
	no	0.47	882.41
CH	yes	0.91	820.74
	no	1.24	808.58
ALL	yes	1.96	948.47
	no	2.28	932.20

The usefulness of estimating the number of served trip requests depending on the time of day and geographical location is illustrated in Fig. 14. It shows the value for $\hat{e}_i$ for an area covering parts of Manhattan and Brooklyn at two different times of the day. As we can see, there are indeed large differences in $\hat{e}_i$. At 05:00, the overall demand level is lower, which makes combining multiple requests into an efficient route more challenging. Hence, the overall level of $\hat{e}_i$ is lower. In contrast, during peaks times, a vehicle can serve more than twice the number of requests. In addition, there are geographical differences. In both images, we see Manhattan towards the left and Brooklyn towards the right. The demand density is higher in the Manhattan area, leading to more efficient vehicle routes and hence a higher value of $\hat{e}_i$.

To evaluate whether the adaptive tuning mechanism can adequately adjust to these effects, we ran additional scenarios in which we fixed $\hat{e}_i$ to a single value that was estimated for each individual dataset based on historical data. The average results are summarized in Table 12. As we can see, the adaptive tuning process reduces the trip request rejection rates in all datasets at the cost of a slightly increased vehicle travel time. Based on these results we may conclude that the adaptive parameter estimation both eases the application of the repositioning algorithm to a new area and improves the overall system performance regarding rejected trip requests.

6 Conclusion and outlook

In this paper, we have presented a new approach for the idle vehicle repositioning problem. The central component of our algorithm is a MIP model that aims to balance supply provided by the vehicle fleet and expected demand given by a forecast. We include an adaptive parameter tuning process in order to reflect temporal and spatial changes in the number of trip requests a vehicle is expected to serve. Our approach is embedded into a larger framework for running dynamic ride-sharing applications and evaluated through extensive simulation studies on four real-world datasets. In this study, we use a perfect and a naive demand forecast that serve as an upper and lower bound on realistic forecasting models. The results with both forecasts show that our algorithm improves trip request rejection rates as well as customer waiting times and ride times across all datasets and scenarios. Thus, our algorithm may be used in real time on large real-world datasets and already performs well with a simple forecast.

In the future, we would like to study other potential applications for our forecast-driven repositioning approach such as the repositioning of autonomous guided vehicles (AGVs) in a large-scale industrial manufacturing or warehousing setting. In addition, we would like to investigate whether our approach could be transferred to applications with decentrally controlled vehicles such as MOD services like Uber or Lyft with self-employed drivers. In these settings, our model and algorithm could be adapted to modify prices and incentivize drivers to reposition to certain areas. Moreover, we aim to integrate additional forecasting components into our system in order to adequately consider other sources of uncertainty. One big factor here would be the inclusion of a forecasting model for travel times, as these vary substantially depending on the current time of day, location, and traffic situation. There are also several minor extensions and improvements that could be made for our MIP model. Despite our consideration of the current system state and the adaptive parameter tuning process, we could still add more details to our algorithm. For instance, it could prove beneficial to consider the current vehicle routes which contain information regarding the future vehicle location and occupation. Moreover, our parameter tuning itself currently only considers information about the immediate past and could be replaced by a more complex approach. For example, one could use a trained machine learning model that forecasts the expected number of served trip requests per vehicle. Besides modifying the current algorithm, we also intend to experiment with completely different algorithmic approaches. While the running times are good, the need for a commercial-grade MIP solver can be problematic in real-world use cases. Therefore, we intend to study different approaches such as network flow models which may be able to solve a similar repositioning model considering current supply and forecasted demand.

Funding Open Access funding enabled and organized by Projekt DEAL. Funding was partially provided by PTV Group (http://ptvgroup.com, Karlsruhe, 76131 Germany).

Availability of data and material The utilized evaluation datasets are partially publicly available and the relevant sources are referenced in the paper.

Code Availability The code utilized in this paper is not publicly available.

Declarations

Conflicts of interest There are no conflicts of interest to declare.

References

Alonso-Mora, J., Samaranayake, S., Wallar, A., Frazzoli, E., & Rus, D. (2017a). On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment. Proceedings of the National Academy of Sciences, 114(3), 462–467. https://doi.org/10.1073/pnas.1611675114

Alonso-Mora, J., Wallar, A., & Rus, D. (2017b). Predictive routing for autonomous mobility-on-demand systems with ride-sharing. In 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), (pp. 3583–3590). Vancouver, BC: IEEE. https://doi.org/10.1109/IROS.2017.8206203.

Bent, R. & Hentenryck, P. V. (2007). Waiting and relocation strategies in online stochastic vehicle routing. In M. M. Veloso (Ed.), IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007, (pp. 1816–1821). http://ijcai.org/Proceedings/07/Papers/293.pdf.

Bent, R. W., & Van Hentenryck, P. (2004). Scenario-based planning for partially dynamic vehicle routing with Stochastic customers. *Operations Research, 52*(6), 977–987. https://doi.org/10.1287/opre.1040.0124

Boyacı, B., Zografos, K. G., & Geroliminis, N. (2017). An integrated optimization-simulation framework for vehicle and personnel relocations of electric carsharing systems with reservations. *Transportation Research Part B: Methodological, 95*, 214–237. https://doi.org/10.1016/j.trb.2016.10.007

Bruglieri, M., Pezzella, F., & Pisacane, O. (2019). An adaptive large neighborhood search for relocating vehicles in electric carsharing services. *Discrete Applied Mathematics, 253*, 185–200. https://doi.org/10.1016/j.dam.2018.03.067

Castiglione, J. & Cooper, D. (2018). TNCs and congestion. https://www.sfcta.org/projects/tncs-and-congestion. Accessed 11 Mar 2020.

Cordeau, J.-F., & Laporte, G. (2007). The dial-a-ride problem: Models and algorithms. *Annals of Operations Research, 153*(1), 29–46. https://doi.org/10.1007/s10479-007-0170-8

Dibbelt, J., Strasser, B., & Wagner, D. (2016). Customizable contraction hierarchies. *Journal of Experimental Algorithmics, 21*(1), 1–49. https://doi.org/10.1145/2886843

Doubek, J. (2018). New york city temporarily halts more uber and lyft cars on the road. https://www.npr.org/2018/08/09/637008474/new-york-city-temporarily-halts-more-uber-and-lyft-cars-on-the-road. Accessed 11 Mar 2020.

Ferrucci, F., Bock, S., & Gendreau, M. (2013). A pro-active real-time control approach for dynamic vehicle routing problems dealing with the delivery of urgent goods. *European Journal of Operational Research, 225*(1), 130–141. https://doi.org/10.1016/j.ejor.2012.09.016

Gambella, C., Malaguti, E., Masini, F., & Vigo, D. (2018). Optimizing relocation operations in electric car-sharing. *Omega, 81*, 234–245. https://doi.org/10.1016/j.omega.2017.11.007

Huang, H., Pouls, M., Meyer, A., & Pauly, M. (2020). Travel time prediction using tree-based ensembles. In E. Lalla-Ruiz, M. Mes, & S. Voß (Eds.), *Computational logistics*. Cham: Springer.

Huang, K., An, K., Rich, J., & Ma, W. (2020b). Vehicle relocation in one-way station-based electric carsharing systems: A comparative study of operator-based and user-based methods. *Transportation Research Part E: Logistics and Transportation Review, 142*, 102081. https://doi.org/10.1016/j.tre.2020.102081

Ichoua, S., Gendreau, M., & Potvin, J.-Y. (2006). Exploiting knowledge about future demands for real-time vehicle dispatching. *Transportation Science, 40*(2), 211–225. https://doi.org/10.1287/trsc.1050.0114

Jung, J. Y., & Chow, J. (2019). Large-scale simulation-based evaluation of fleet repositioning strategies for dynamic rideshare in New York city. In WCX SAE World Congress Experience: SAE International. https://doi.org/10.4271/2019-01-0924

Ke, J., Zheng, H., Yang, H., & Xiqun, & Chen,. (2017). Short-term forecasting of passenger demand under on-demand ride services: A spatio-temporal deep learning approach. *Transportation Research Part C: Emerging Technologies, 85*, 591–608. https://doi.org/10.1016/j.trc.2017.10.016

Li, B., Zhang, D., Sun, L., Chen, C., Li, S., Qi, G., & Yang, Q. (2011). Hunting or waiting? Discovering passenger-finding strategies from a large-scale real-world taxi dataset. In 2011 IEEE International Con-

ference on Pervasive Computing and Communications Workshops (PERCOM Workshops), (pp. 63–68). https://doi.org/10.1109/PERCOMW.2011.5766967.

Li, X., Pan, G., Wu, Z., Qi, G., Li, S., Zhang, D., Zhang, W., & Wang, Z. (2012). Prediction of urban human mobility using large-scale taxi traces and its applications. *Frontiers of Computer Science, 6*(1), 111–121. https://doi.org/10.1007/s11704-011-1192-6

Liao, S., Zhou, L., Di, X., Yuan, B., & Xiong, J. (2018). Large-scale short-term urban taxi demand forecasting using deep learning. In 2018 23rd Asia and South Pacific Design Automation Conference (ASP-DAC), (pp. 428–433). https://doi.org/10.1109/ASPDAC.2018.8297361.

Lippi, M., Bertini, M., & Frasconi, P. (2013). Short-term traffic flow forecasting: An experimental comparison of time-series analysis and supervised learning. *IEEE Transactions on Intelligent Transportation Systems, 14*(2), 871–882. https://doi.org/10.1109/TITS.2013.2247040

Lowalekar, M., Varakantham, P., & Jaillet, P. (2021). Zone pAth construction (ZAC) based approaches for effective real-time ridesharing. *Journal of Artificial Intelligence Research, 70*, 119–167. https://doi.org/10.1613/jair.1.11998

Ma, S., Zheng, Y., & Wolfson, O. (2015). Real-time city-scale taxi ridesharing. *IEEE Transactions on Knowledge and Data Engineering, 27*(7), 1782–1795. https://doi.org/10.1109/TKDE.2014.2334313

Ma, T.-Y., Rasulkhani, S., Chow, J. Y., & Klein, S. (2019). A dynamic ridesharing dispatch and idle vehicle repositioning strategy with integrated transit transfers. *Transportation Research Part E: Logistics and Transportation Review, 128*, 417–442. https://doi.org/10.1016/j.tre.2019.07.002

Mes, M., van der Heijden, M., & Schuur, P. (2010). Look-ahead strategies for dynamic pickup and delivery problems. *OR Spectrum, 32*(2), 395–421. https://doi.org/10.1007/s00291-008-0146-3

Mitrović-Minić, S., & Laporte, G. (2004). Waiting strategies for the dynamic pickup and delivery problem with time windows. *Transportation Research Part B: Methodological, 38*(7), 635–655. https://doi.org/10.1016/j.trb.2003.09.002

Nourinejad, M., & Roorda, M. J. (2014). A dynamic carsharing decision support system. *Transportation Research Part E: Logistics and Transportation Review, 66*, 36–50. https://doi.org/10.1016/j.tre.2014.03.003

Pouls, M., Meyer, A., & Ahuja, N. (2020). Idle vehicle repositioning for dynamic ride-sharing. In E. Lalla-Ruiz, M. Mes, & S. Voß (Eds.), *Computational logistics.* Cham: Springer.

Powell, J. W., Huang, Y., Bastani, F., & Ji, M. (2011). Towards reducing taxicab cruising time using spatio-temporal profitability maps. In D. Pfoser, Y. Tao, K. Mouratidis, M. A. Nascimento, M. Mokbel, S. Shekhar, & Y. Huang (Eds.), *Advances in spatial and temporal databases.* Heidelberg: Springer.

Repoux, M., Boyacı, B., & Geroliminis, N. (2015). Simulation and optimization of one-way car-sharing systems with variant relocation policies. In TRB 94th Annual Meeting Compendium of Papers, (p. 19).

Repoux, M., Kaspi, M., Boyacı, B., & Geroliminis, N. (2019). Dynamic prediction-based relocation policies in one-way station-based carsharing systems with complete journey reservations. *Transportation Research Part B: Methodological, 130*, 82–104. https://doi.org/10.1016/j.trb.2019.10.004

Riley, C., van Hentenryck, P., & Yuan, E. (2020). Real-Time Dispatching of Large-Scale Ride-Sharing Systems: Integrating Optimization, Machine Learning, and Model Predictive Control. In Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, (pp. 4417–4423). Yokohama, Japan: International Joint Conferences on Artificial Intelligence Organization. https://doi.org/10.24963/ijcai.2020/609.

Ritzinger, U., Puchinger, J., & Hartl, R. F. (2016). A survey on dynamic and stochastic vehicle routing problems. *International Journal of Production Research, 54*(1), 215–231. https://doi.org/10.1080/00207543.2015.1043403

Schmid, V. (2012). Solving the dynamic ambulance relocation and dispatching problem using approximate dynamic programming. *European Journal of Operational Research, 219*(3), 611–621. https://doi.org/10.1016/j.ejor.2011.10.043

Shah, S., Lowalekar, M., & Varakantham, P. (2020). Neural Approximate Dynamic Programming for On-Demand Ride-Pooling. Proceedings of the AAAI Conference on Artificial Intelligence, 34(01), 507–515. https://doi.org/10.1609/aaai.v34i01.5388

Thomas, B. W. (2007). Waiting strategies for anticipating service requests from known customer locations. *Transportation Science, 41*(3), 319–331. http://www.jstor.org/stable/25769357.

Uber (2020). Surge pricing. https://marketplace.uber.com/pricing/surge-pricing. Accessed 07 Apr 2020.

Uber (2021). Uberpool. https://www.uber.com/us/en/ride/uberpool. Accessed 08 Jan 2021.

Ulmer, M. W., Goodson, J. C., Mattfeld, D. C., & Hennig, M. (2019). Offline-online approximate dynamic programming for dynamic vehicle routing with stochastic requests. *Transportation Science, 53*(1), 185–202. https://doi.org/10.1287/trsc.2017.0767

Ulmer, M. W., Goodson, J. C., Mattfeld, D. C., & Thomas, B. W. (2020). On modeling stochastic dynamic vehicle routing problems. *EURO Journal on Transportation and Logistics, 9*(2), 100008. https://doi.org/10.1016/j.ejtl.2020.100008

Ulmer, M. W., Soeffker, N., & Mattfeld, D. C. (2018). Value function approximation for dynamic multi-period vehicle routing. *European Journal of Operational Research, 269*(3), 883–899. https://doi.org/10.1016/j.ejor.2018.02.038

Vlahogianni, E. I., Golias, J. C., & Karlaftis, M. G. (2004). Short-term traffic forecasting: Overview of objectives and methods. *Transport Reviews, 24*(5), 533–557. https://doi.org/10.1080/0144164042000195072

Vlahogianni, E. I., Karlaftis, M. G., & Golias, J. C. (2014). Short-term traffic forecasting: Where we are and where were going. *Transportation Research Part C: Emerging Technologies, 43*, 3–19. https://doi.org/10.1016/j.trc.2014.01.005

Voccia, S. A., Campbell, A. M., & Thomas, B. W. (2019). The same-day delivery problem for online purchases. *Transportation Science, 53*(1), 167–184. https://doi.org/10.1287/trsc.2016.0732

Xu, M., & Meng, Q. (2019). Fleet sizing for one-way electric carsharing services considering dynamic vehicle relocation and nonlinear charging profile. *Transportation Research Part B: Methodological, 128*, 23–49. https://doi.org/10.1016/j.trb.2019.07.016

Yao, H., Tang, X., Wei, H., Zheng, G., & Li, Z. (2019). Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction. Proceedings of the AAAI Conference on Artificial Intelligence, 33, 5668–5675. https://doi.org/10.1609/aaai.v33i01.33015668

Yao, H., Wu, F., Ke, J., Tang, X., Jia, Y., Lu, S., Gong, P., Ye, J., & Zhenhui, L. (2018). Deep multi-view spatial-temporal network for taxi demand prediction. In The Thirty-Second AAAI Conference on Artificial Intelligence.

Zhang, J., Zheng, Y., Qi, D., Li, R., Yi, X., & Li, T. (2018). Predicting citywide crowd flows using deep spatio-temporal residual networks. *Artificial Intelligence, 259*, 147–166. https://doi.org/10.1016/j.artint.2018.03.002