Technische Universität München
TUM School of Engineering and Design

Autonomous Ride-Pooling-Services in Urban Environments: Operational Strategies and Simulation

Roman Engelhardt

Vollständiger Abdruck der von der TUM School of Engineering and Design der Technischen Universität München zur Erlangung des akademischen Grades eines *Doktors der Ingenieurwissenschaften (Dr.-Ing.)* genehmigten Dissertation

Vorsitz: Prof. Dr.-Ing. Rolf Möckel

Prüfer der Dissertation:

1. Prof. Dr.-Ing. Klaus Bogenberger
2. Prof. Dr. Hani S. Mahmassani
3. Prof. Dr. Maximilian Schiffer

Die Dissertation wurde am 02.10.2024 bei der Technischen Universität München eingereicht und durch die *TUM School of Engineering and Design* am 02.12.2024 angenommen.

Executive Summary

Increasing urbanization and the resulting demand for mobility in urban areas provide challenges for the transportation sector to cope with space scarcity, congestion, pollution, and greenhouse gas emissions. New modes of transportation are needed to reduce the dependency on private cars and bridge the gap to high-capacity public transportation. Technological advancements in connectivity, automation, and electrification have, and will continue to, enable new mobility services and business models. Especially shared mobility services have the potential to reduce the number of parking cars if private car ownership can be substituted, increase traffic efficiency if trips can be shared, and reduce emissions if electric vehicles are used. Without the cost of a driver, automated vehicles can provide these services at a low price, enabling frequent usage to achieve the desired benefits.

This thesis deals with autonomous ride-pooling, a shared mobility service that combines the described technologies. In autonomous ride-pooling, a service provider operates a fleet of autonomous vehicles that offers rides for customers on-demand. Customers share rides with other customers if they travel in a similar direction to increase the occupancy of the vehicles. In terms of service availability and travel time, autonomous ride-pooling can provide convenience similar to private cars. At the same time, a higher vehicle occupancy allows for a more efficient use of the road infrastructure.

This thesis focuses on the operations of autonomous ride-pooling services. To provide the service, potentially thousands of autonomous vehicles need to be dispatched to pick up and drop off customers. A central control system must compute vehicle routes and schedules in real-time to serve dynamically incoming ride requests while ensuring availability of vehicles in the operating area. Three key research questions are addressed in this thesis: 1) Assignment: How can the operator efficiently assign customers and schedules to fleet vehicles? 2) Repositioning: How can the operator reposition idle vehicles to ensure availability and efficiency? 3) Reservation: How can the operator offer customers the option to reserve a ride in advance?

Solving the assignment problem requires finding the solution to a large-scale Vehicle Routing Problem in real-time. As this problem is notoriously hard to solve, a tailored algorithm for the ride-pooling setting is developed. As customers of this service expect a convenient service, tight time windows for pick-up and drop-off can be exploited to reduce the search space for feasible vehicle schedules. An efficient search strategy exploits the problem's structure, while a dynamically updated database of computed vehicle schedules allows the reuse of solutions from previous assignments.

To maintain the availability of vehicles in the operating area, idle vehicles have to be repositioned dynamically to balance demand and supply. A key question is estimating the required supply in specific areas based on a prediction of future demand. In a ride-pooling service, this estimation must incorporate that future rides can be shared and accommodated by currently non-idle vehicles en-route. A repositioning algorithm is developed in this thesis that samples

future requests from a forecast and simulates future fleet states to detect supply shortages. An optimization problem is formulated to find repositioning actions to prevent these shortages.

When customers are allowed to reserve trips in advance, the operator can benefit, on the one hand, from additional information about future demand to plan vehicle schedules accordingly. On the other hand, the operator must commit to serving these reservations without knowing the future state of the system. A multi-rolling horizon approach is developed in this thesis that allows scheduling pre-booked rides while also serving dynamically incoming on-demand requests. The algorithm can guarantee the fulfillment of reservations while re-optimizing short-term schedules to provide an efficient mixed service for on-demand and pre-booking customers.

To evaluate the developed algorithms, an agent-based simulation framework tailored to assess ride-pooling services is developed. Next to the proposed algorithms for assignment, repositioning, and reservation, state-of-the-art benchmark algorithms are implemented to compare the performance. Case studies for the cities of Chicago, Munich, and Manhattan are conducted to evaluate the algorithms in different urban environments.

General results show a huge potential for the autonomous ride-pooling service: Approximately 1,000 fewer vehicles are needed to serve the ride-hailing demand in Chicago when trips are shared. In Munich, 1,250 vehicles can replace 10% of private vehicle trips in the city, while only 11% of the taxi fleet size is needed in Manhattan to serve the taxi rides. The comparison with benchmark algorithms shows that the developed assignment algorithm can reduce average termination times by 76% in the Chicago case study compared to a state-of-the-art benchmark algorithm with similar performance. The evaluation of repositioning shows the general importance of distributing vehicles in the operating area to ensure availability. In the Chicago and Manhattan case studies, the fraction of served requests could be increased by up to 40%. At the same time, vehicles generated revenue for the service for up to 6 additional hours per day if repositioning is applied. Compared to benchmark algorithms, the developed repositioning algorithm can increase the number of served requests by up to 3%. At least the same service rate can be achieved in all scenarios tested. Nevertheless, in scenarios where the proposed algorithm does not improve the service rate, the amount of empty repositioning vehicle kilometers can be reduced significantly, proving the efficiency of the developed algorithm. Concerning reservations, the developed multi-rolling horizon approach can successfully incorporate pre-booked rides into the service while guaranteeing the fulfillment of these reservations. Especially the ability to re-optimize short-term schedules to serve pre-booking and on-demand customers simultaneously can improve the service rate by up 7%. Nevertheless, the general impact of reservations on system performance is mixed: Slightly positive effects can be observed when the fraction of pre-booked rides is either low or very high, and the distribution of these rides is correlated with the distribution of on-demand requests. In the other scenarios, the performance of the service can slightly deteriorate due to the commitment to serve reservations.

Contents

Chapter 1

Introduction

1.1 Motivation

By 2050, the UN projects that 68% of the world's population will live in urban areas, up from 55% in 2018 [United Nations, 2018]. A rise in travel demand increasing traffic congestion, greenhouse gas emissions, air pollution and noise accompany this trend. In the United States (US), the times of congestion increased by 48% between 2007 and 2017, resulting in a total cost of 179$ billion annually [Schrank et al., 2019]. While during Covid-19 congestion reduced in 2019, pre-pandemic levels are reached again in many US and European cities [TomTom, 2022]. Besides the problem of congestion, the transportation sector is additionally responsible for a large share of greenhouse gas emissions, 28% in the US [US EPA, 2021] and 20% in Germany [Umweltbundesamt, 2022]. It is therefore imperative to enhance traffic efficiency: On the one hand, to combat climate change and comply with more and more stringed regulations, such as the "European Green Deal" [European Commission, 2023] seeking to reduce greenhouse gas emissions of cars by 55% by 2030 compared to 1990 levels, and on the other hand, to tackle resource scarcity induced by rising mobility demand. These problems have to be solved while still sufficient mobility solutions are provided to the people. Technology can have the potential to restructure the current status quo of mobility and meet these criteria. In fact Sperling [2018] describes three – currently happening – revolutions of the transportation sector: Automation, Electrification and Shared Mobility.

Driven by policies and subsidies aiming to reduce greenhouse gases, electrification in the transportation sector, i.e., the transition from fossil fuels to electric powertrains, is currently rapidly evolving. While the penetration of fully electric vehicles was below 1% in the European Union (EU) and the US before 2017, in 2022 it already reached 12% and 6.2% in EU and US, respectively [Joel Jaeger, 2023]. Also in China, electric vehicle sales have risen drastically in recent years. While in 2023, 95% of all electric vehicles were sold in China, Europe, and the US, just under 60% of all electric cars were sold in China in this time frame [IEA, 2024]. Especially considering the planned phase-out of combustion engines in many countries (e.g., effectively by 2035 in the EU [European Commission, 2023] and the United States [The White House, 2021]), the future domination of electric vehicles seems inevitable.

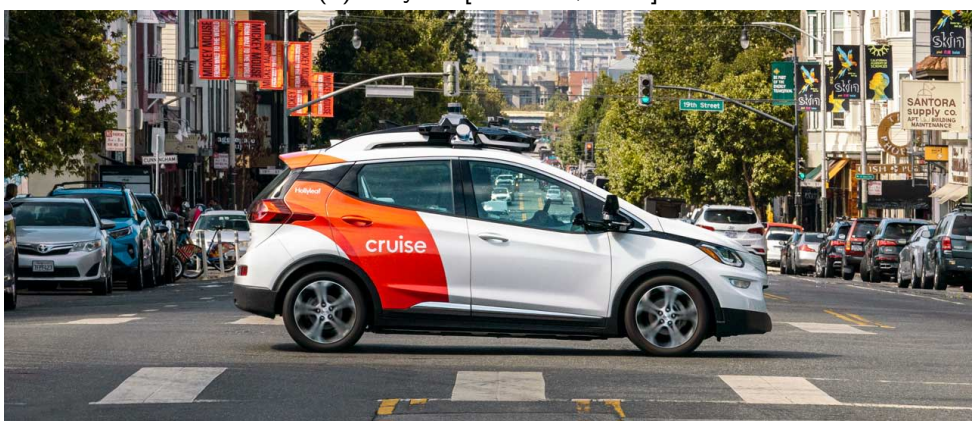
Nevertheless, just electrifying the current fleet of vehicles will not be sufficient to solve the problems of the transportation sector. Still, the same inefficient usage of private vehicles will lead to wasted space in cities and traffic congestion. If usage does not change, private vehicles remain idle for around 23 hours per day [Shoup and American Planning Association, 2005] blocking valuable space in cities, while a low average occupancy (e.g.,1.1 persons per ve-

hicle for commuting trips [Umweltbundesamt, 2024]) still results in inefficient road usage and congestion. While public transportation provides a sustainable alternative, a mode shift from private vehicles to public transit is often hindered by the lack of availability, flexibility, reliability, and comfort of public transportation services (e.g. [Göransson and Andersson, 2023]). Shared mobility can provide a solution to these problems. Instead of restricting vehicles to private use, the goal is to maximize the utilization of mobility resources by sharing them among multiple users while still providing a convenient and reliable service. Although the coordination of shared mobility (i.e., the matching of users and resources) provided technical challenges in the past, the rise of digitalization with GPS-enhanced smartphone applications enabled a rapid growth of shared mobility services in the last decade. For example in Germany, the number of registered users for carsharing services – in which customers can rent a vehicle for a short period of time – has increased from 0.45 million in 2013 to 4.47 million in 2023 [Bundesverband CarSharing e.V., 2023]. Alternatively, the goal of ride-sharing is to share a trip among multiple users. With the original intention to connect private drivers and passengers to share parts of their trips, Transportation Network Companies (TNCs) like Uber, Lyft, or Didi have emerged in the last decade. Uber, for example, was founded in 2009 and reported an increase in the number of trips from 3.0 billion in 2017 to 7.6 billion in 2022 [WallStreetZen, 2023].

Despite the increasing availability and popularity of shared mobility services, costs for using the service remain too high to replace most private vehicle trips. The largest cost component of these services is the driver, which is required to operate the vehicle. Vehicle automation has the potential to remove this cost component and therefore reduce the cost of shared mobility services significantly. Becker et al. [2020], for example, estimated the costs for these mobility services could be reduced by 29% to 84% mainly depending on the local level of labor costs. While the deployment of automated vehicles is still in early the stages of development, first test services are already in operation, waiting for large-scale rollouts. The best known test services are *Waymo* [Waymo, 2024], an automated taxi service currently operating in Phoenix, Arizona, and the automated services by *Cruise* [Cruise, 2024] and *Waymo* in San Francisco, California. But also in China, automated taxi services are in operation in multiple cities, for example by *Baidu* [Magramo et al., 2024] or *AutoX* [AutoX, 2024].

All these three revolutions combined – electrification, shared mobility, and automation – have the potential to disrupt the current status quo of mobility as we know it. If all revolutions converge into a socially beneficial system of Shared Autonomous Electric Vehicles (SAEVs), citizens may rely on low-cost, clean, convenient, and efficient mobility services for either door-to-door transport, or acting to bridge the gap to long-haul public transport [Narayanan et al., 2020].

(a) Waymo [WAYMO, 2024]



(b) Cruise [CRUISE, 2024]

Figure 1.1: Examples of automated vehicles of the test services by Waymo and Cruise in San Francisco.

1.2 Autonomous Ride-Pooling

This thesis deals with Autonomous Ride-Pooling (ARP) services, which can be interpreted as such a convergence. In an ARP service, customers can request a trip via a smartphone application from an origin location to a destination location. This trip can either be requested on-demand, which indicates a service as fast as possible, or it can be pre-booked to a specific time in advance. An operator (or service provider) offers the service and operates a fleet of Autonomous Vehicles (AVs). The operator's task is to coordinate its fleet and assign vehicles to serve incoming requests. The operator aims to pool multiple customers with similar origin-destination relations to share parts of their trip, thereby increasing vehicle utilization and reducing costs.

When designing such a service, the objectives of multiple stakeholders have to be considered to achieve a mobility service that is beneficial for all. From the operator's perspective, the

service has to be profitable. From the customer's perspective, the service must be attractive. From a regulatory perspective, the service has to benefit the society as a whole. The detailed specifications of these stakeholder objectives depend on the local context of the service.

This thesis focuses on the urban operation of an ARP service. Due to high population density, high demand for a city-scale service can be expected, requiring a large fleet (possibly hundreds to thousands or even tens of thousands) of AVs operating in the city to meet the demand. With good access to other mobility services, customers likely expect a high level of service, i.e., short waiting and travel times, and high service reliability. A service of this scale issues also challenges for municipal authorities, which have to ensure that the service provides social benefits. For approving the service, authorities may require the service to minimize traffic impacts or generally embed the service into the local public transportation system. The service provider, on the other hand, has to design and operate its service to meet the demands of customers and satisfy municipal requirements while still ensuring a profitable operation.

The focus of this thesis is on the operational perspective. Multiple questions have to be answered when designing and operating an ARP service, for example:

- How much demand can be expected?
- How many vehicles are required to meet the demand, and how many seats should they provide?
- How should vehicles be operated and controlled to serve the demand?
- In which operating area should the service be offered?
- Which services should be offered to customers (e.g., express vs. standard or on-demand vs. pre-booking)?
- What pricing scheme should be used?
- How should different stakeholder objectives be incorporated?

This list is not exhaustive, and additionally, answers to one question might influence the answer to another question (e.g., the pricing scheme will influence the demand that can be expected).

These questions can be summarized in the general question:

> **How should the ARP service provider operate and control its vehicles to serve the demand?**

Therefore, the thesis will focus on the backbone of the ARP service: The assignment of routing tasks to its fleet vehicles to serve customers of the service efficiently. The following section defines the problem statement and research questions in more detail.

1.3 Problem Statement and Research Questions

The goal of this thesis is to develop and evaluate methods to operate an ARP service in an urban environment. The underlying control problem can be formulated as follows: Given a

fleet of vehicles with certain attributes (e.g., number of seats), assign tasks to the vehicle fleet that serves incoming requests while optimizing a specific objective, such as the profit of the service. Tasks, thereby, involve directing a vehicle to a specific location at a designated time, following a particular route within a predefined operating area, to perform a specific action (such as picking up or dropping off a customer). An ordered set of tasks for a vehicle is called a schedule.

This problem, i.e., assigning the optimal set of schedules to vehicles, will be discussed in detail in this thesis. Unfortunately, it is notoriously hard to solve, resulting from the exploding number of possible schedules to choose from and assign to vehicles. Additionally, the problem is highly dynamic and stochastic. The primary source of dynamism and stochasticity stems from customers requesting trips on demand. These requests are usually not known in advance – at best, a distribution of requests from historical data is known – and the operator has to react to these requests in real-time.

This thesis aims to develop methods to solve this problem efficiently that could be applied in a real-world ARP service. Further, these methods are evaluated based on the impact on Key Performance Indicators (KPIs) of all stakeholders. The general control problem sketched above is split into three key subproblems, which are the focus of this thesis, namely **"Assignment"**, **"Repositioning"**, and **"Reservation"**. These subproblems and their associated research questions are defined in the following subsections.

Since a real-world ARP service is currently unavailable and testing these methods would not be economically viable, the evaluation is performed in a simulation environment. An agent-based simulation tool is developed that allows the evaluation of the proposed methods in a realistic setting. Case studies for Munich, Germany, Chicago, US and Manhattan, US are conducted to evaluate the methods in diverse settings and prove their general applicability.

1.3.1 Impacts of Ride-Pooling

The first set of research questions deals with the general impact of the Autonomous Ride-Pooling (ARP) and its choice of design parameters on the performance of the service. As sharing of rides is the key component of a ride-pooling service, on a higher level, the goal is to answer the research question

RQ I: What are the benefits of pooling rides?

Compared to a ride-hailing service, where each customer is served individually, a pooled service is expected to have a higher fleet utilization and, therefore, a higher efficiency. This higher fleet efficiency can lead to a reduction in the number of vehicles required to serve the same demand, resulting in the research questions *"How many vehicles are required to serve the demand when rides are pooled?"*, and *"How many seats should these vehicles provide to facilitate pooling?"*.

Demand for the service is a central parameter for its success. Therefore, the question arises *"How does the demand for the service impact the performance of an ARP service?"*. Especially in ride-pooling services, a sufficiently high demand seems critical to ensure that shareable trips can be found.

To answer these research questions, solving the control problem of the ARP service is necessary, which in turn involves additional research questions that are discussed in the following sections.

1.3.2 Assignment

Solving the assignment problem is the core of the ARP service. The solution to this problem defines which customer is served by which vehicle and creates the corresponding schedules and routes for fleet vehicles to serve these customers. The goal of the assignment problem is to accommodate new requests by finding vehicle schedules that optimize some objective while considering the current fleet state is considered and certain constraints are fulfilled.

Besides external factors, the algorithm for solving the assignment problem might be the most decisive factor for the performance of the ARP service and its impact on customers and traffic, motivating the main research question of this section:

> **RQ II:** How can the operator of an ARP service assign customers and schedules to fleet vehicles efficiently?

Mathematically, the underlying control problem is a Vehicle Routing Problem (VRP) defined by its objective and a set of constraints.

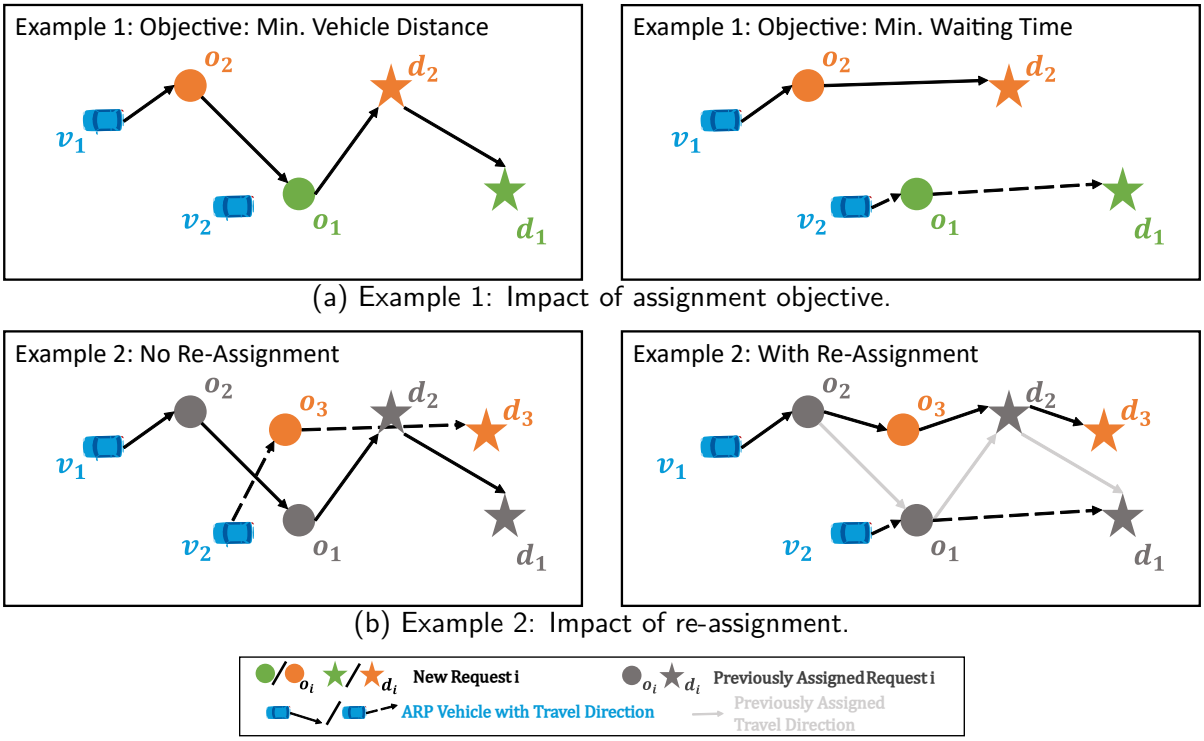
The objective is a control function that is used to rate possible vehicle schedules to enable a comparison between them and, therefore, a decision on which schedule to choose. The first example in Figure 1.2a shows the impact of two different (conflicting) objectives on the selection of schedules. To serve two requests with two vehicles, a schedule where customers share part of their trip would be the best option if the goal is to minimize the overall vehicle distance traveled. If, on the other hand, the objective is to offer a more convenient service for customers by, for example, minimizing their waiting time for pick-up, an assignment where the customers are served by different vehicles is preferred. Nevertheless, this solution will lead to additional vehicle kilometers traveled by the vehicle fleet.

Constraints, on the other hand, describe hard limits that have to be satisfied for schedules to be considered feasible. The vehicle capacity (i.e., number of passenger seats), for example, describes a hard limit on the number of customers allowed to be onboard the vehicle simultaneously. Time constraints can be applied to ensure a specific service level guarantee. For an ARP service, time constraints often limit the customer waiting time for pick-up or the maximum in-vehicle travel time which might get prolonged in case of pooling.

In a dynamic environment, an additional constraint is a fast computational time of the assignment algorithm to enable real-time decision-making and short response times to incoming requests. As the number of possible schedules grows exponentially with the number of vehicles and requests, and brute-force methods that search the whole solution space are computationally infeasible, the question arises *"How can the dynamic assignment problem be solved efficiently with short response times?"*. Typically, heuristic methods can be used to limit the search space and find good solutions in a reasonable time, resulting in a trade-off between solution quality and computational time that has to be quantified.

In the dynamic context, also the continuity of the assignment must be ensured. As new customers dynamically request new trips, decisions made in the past have to be taken into account when assigning new schedules to vehicles. It has to be ensured that customers who have already booked the trip are still scheduled to be served. Nevertheless, reconsideration of previous decisions should be taken into account after new customers have made their request. Depending on the incoming requests, it might be beneficial to re-assign already scheduled customers to different vehicles to accommodate new requests. The second example

in Figure 1.2b shows a possible impact of re-assignment. In a previous decision, the first vehicle was scheduled to serve request 1 and request 2 in a shared route. Once customer 3 requested a trip, it is beneficial (at least with the objective of minimizing vehicle distance) to re-assign request 2 to the other vehicle and serve requests 1 and 3 together. Nevertheless, from a customer standpoint, re-assignments also have disadvantages: Re-assignment will lead to varying scheduled pick-up times, reducing the predictability of the actual pick-up time and reliability of communicated scheduled pick-up times for the customer. It is therefore natural to ask *"Next operational benefits of re-assignments in an ARP service, what are the impacts on customers?"*, and consequently *"Which methods can be applied to reduce the drawbacks of re-assignments for customers while operational benefits are maintained?"*



(a) Example 1: Impact of assignment objective.

(b) Example 2: Impact of re-assignment.

Figure 1.2: Two examples sketching the possible impact of assignment objective and re-assignment.

1.3.3 Repositioning

Once vehicles finish their assignment, they might end up idle in the area of their last scheduled customer drop-off. These locations are usually different from areas where new customer requests emerge, driving the system into a spatio-temporal imbalance of demand and supply. As vehicles would tend to have a long approach trip to a new request, this imbalance would lead to high customer waiting times and/or customer cancellations. To reduce this imbalance, idle vehicles can be pro-actively redistributed to areas where future demand is expected. This

procedure is sketched in Figure 1.3 and referred to as "Repositioning" or "Rebalancing". This leads to the next research question:

RQ III: How does an ARP service benefit from repositioning?

Determining these trips usually requires three steps: 1) A prediction for future demand. 2) A methodology to determine the imbalance of demand and supply. 3) A formulation to define which vehicles should be rebalanced. This thesis focuses on the second and third steps. A special focus is put on the estimation of imbalance, when rides can be shared. In contrast to ride-hailing services that do not allow sharing, the imbalance is not only determined by the number of requests but also by the possibility of sharing rides of these requests, and the potential accommodation of en-route vehicles to serve these requests has to be considered. This observation leads to the questions *"How can the imbalance of demand and supply for an ARP service be determined?"*, and consequently *"Based on the imbalance of demand and supply, how can repositioning trips be assigned to vehicles?"*.

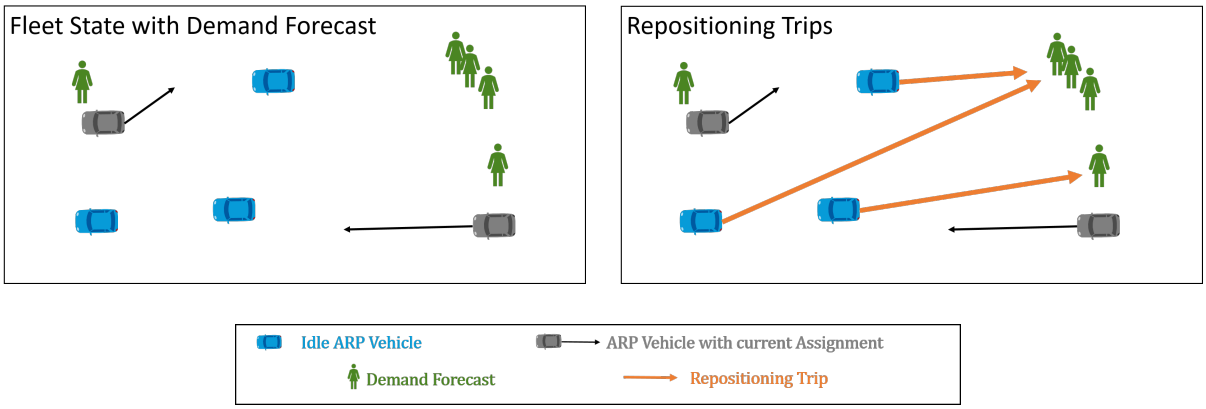


Figure 1.3: Sketch showing the repositioning problem. After serving requests, vehicles might end up in areas of low demand. Repositioning describes the pro-active dispatching of idle vehicles in areas with high expected demand.

1.3.4 Reservation

The final topic of this thesis deals with trip reservations (or pre-bookings) in the ARP service. Some trips – for example scheduled appointments – are known to travelers long time in advance. Instead of risking potential long waiting times or even no service at all, it can be beneficial for customers to book these trips in advance. Pre-bookings might also be beneficial for the operator: As pre-bookings are known in advance, the operator can plan its fleet accordingly, reducing the uncertainty of the system. On the contrary, pre-bookings might also deteriorate the performance of the service. As the operator commits to serve a pre-booked trip, once it was accepted, it might hinder the operator from serving more profitable on-demand requests.

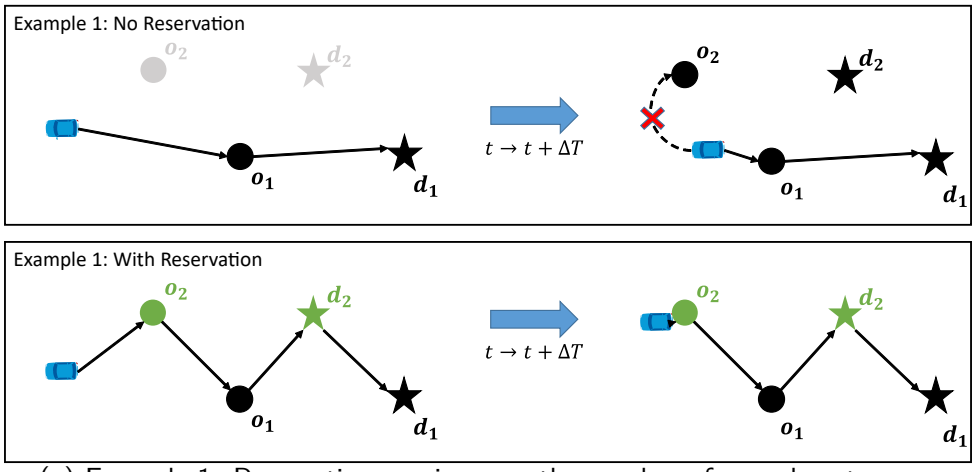
Figure 1.4 illustrates two examples where pre-bookings can improve or worsen the state of the ride-pooling service. The sketches show situations with a single vehicle in blue and multiple

requests i and their corresponding origin o_i and destination d_i indicated by a circle and a star, respectively. Different colors indicate different states of the request: Black corresponds to an on-demand customer that already requested a trip. Grey depicts a customer that is about to request a trip (still unknown to the operator at time t) and green corresponds to a pre-booked trip. In the first example (Figure 1.4a), only request 1 is known to the operator initially. The vehicle is assigned to pick up the customer. Later, customer 2 requests a trip, but the vehicle is no longer able to serve this customer without violating possible time constraints on pick-up and drop-off of customer 1. Therefore, customer 2 is rejected (indicated by the red cross). If customer 2 on the other hand booked the trip in advance, a feasible schedule for the vehicle could have been found to serve both customers. In contrast, example 2 (Figure 1.4b) shows a situation where booking in advance deteriorates the number of served customers. Without pre-booking a situation is sketched where three customers request a trip at the same time. A feasible schedule can be found that serves customers 1 and 3, but customer 2 has to be rejected. On the other hand, if customer 2 already booked the trip in advance, the operator guaranteed its service and can no longer serve the customers 1 and 3.

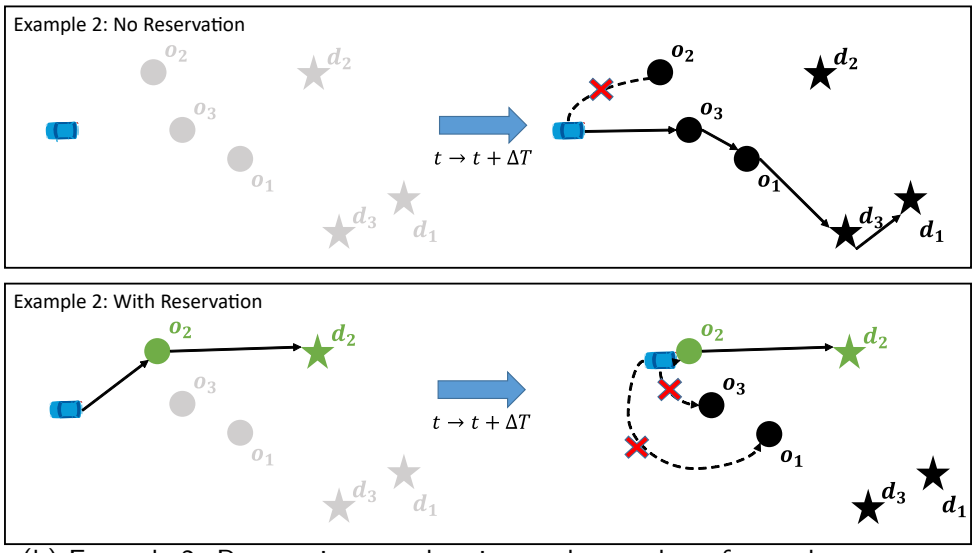
The underlying research question therefore is:

RQ IV: Does an operator of an ARP service benefit from offering pre-bookings?

Allowing pre-bookings to the service also raises operational questions, that need to be solved to answer **RQ IV**. The assignment algorithm has to deal with short term on-demand requests and long term pre-bookings at the same time and find feasible assignments for on-demand requests while ensuring the service of confirmed pre-bookings. From the assignment perspective, therefore the questions arise *"How to incorporate pre-bookings into the assignment problem of an ARP service?"*, and *"How can the service for pre-booked trips be guaranteed while still serving on-demand requests?"*. Additionally, the repositioning algorithm has to consider the long-term pre-bookings when determining available vehicles for repositioning, leading to the question *"How can long-term pre-bookings be incorporated into the repositioning algorithm?"*.



(a) Example 1: Reservation can improve the number of served customers.



(b) Example 2: Reservation can deteriorate the number of served customers.

Figure 1.4: Two Examples sketching the possible impact of reservations on the ARP-Systems.

1.4 Outline of the Thesis

The outline of the thesis is shown in Figure 1.5 highlighting the affiliation of the content to the research questions formulated in this chapter.

In the next chapter, a detailed literature review is provided to give an overview of the state of the art in the field of ARP services. The focus of the chapter lies on modeling ARP services and solution algorithms for solving the assignment and repositioning problems, and dealing with pre-bookings. Chapter 3 is the core of the thesis and describes the algorithms developed in this work. The chapter is split into four parts. First, the general problem formulation is provided, followed by the three subproblems assignment, repositioning, and pre-booking. Chapter 4 describes the simulation environment developed for this thesis, which is used to evaluate the developed algorithms. The chapter provides the description of the three case studies conducted for Chicago, Munich, and Manhattan, which are used to answer the research questions formulated in this chapter. Chapter 5 provides the results of the case studies. Finally, Chapter 6 concludes the thesis, answers the research questions, formulates limitations, and provides an outlook on future research directions.

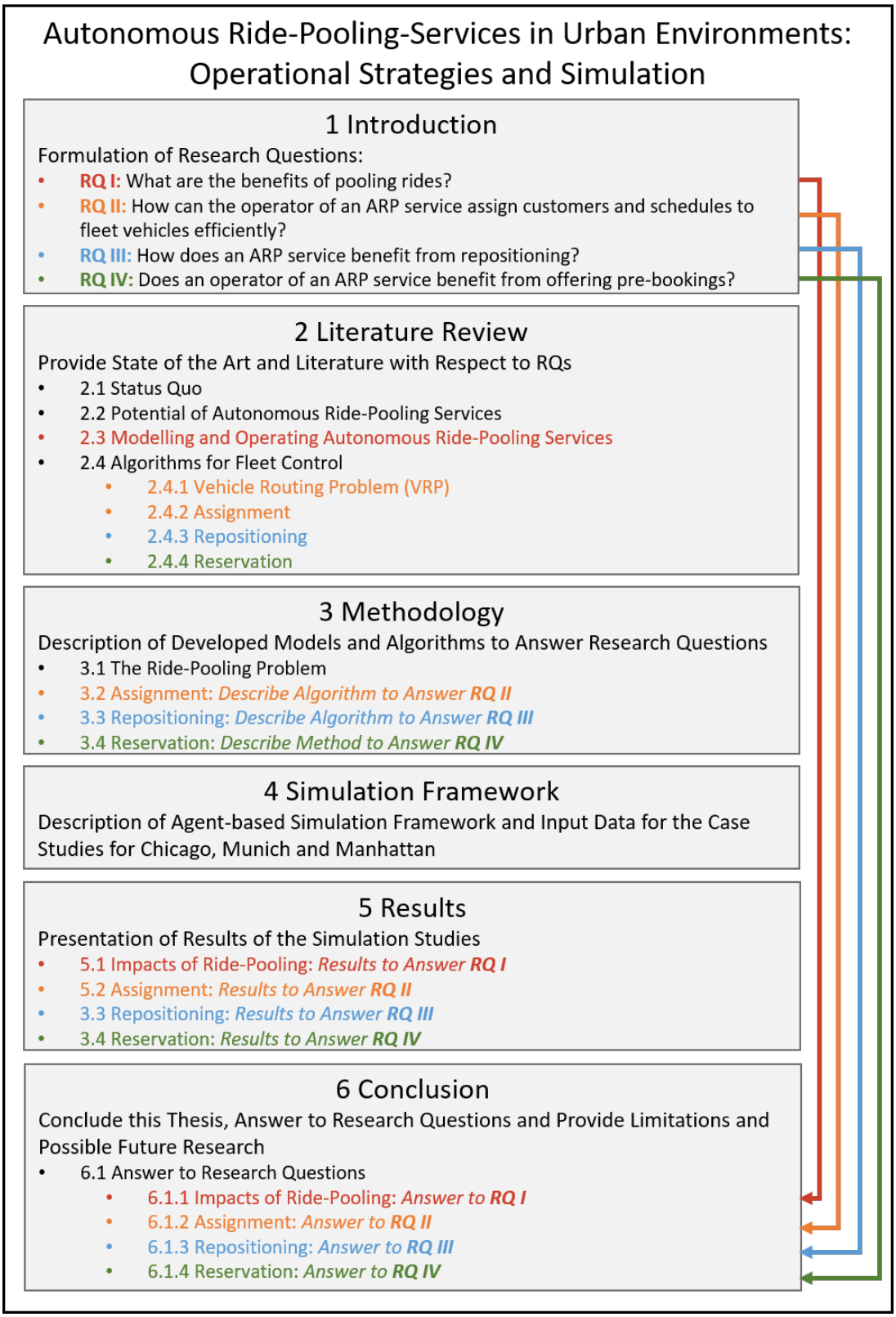


Figure 1.5: Structure of this thesis. Colors indicate affiliation of content to specific research questions.

Chapter 2

Literature Review

This chapter provides an overview of the current state of research in the field of autonomous ride-pooling services. It starts with a brief overview of the current state of shared mobility services and autonomous vehicles. Literature on the potential of autonomous ride-pooling services is then reviewed, focusing on simulation studies that evaluate the potential of these services and their implications for future transportation systems. It then discusses modeling approaches to evaluate these services. The final section reviews fleet control algorithms, focusing on the assignment, repositioning, and reservation problems in the context of autonomous ride-pooling services. Within these sections, the contribution of this thesis is also highlighted.

2.1 Status Quo

2.1.1 Ecosystem of Shared Mobility Services

Machado et al. [2018] define shared mobility as trip alternatives aiming to maximize the utilization of the mobility resources that society can pragmatically afford, disconnecting their usage from ownership. Shared mobility is an old concept that has been around for centuries, but only with the rise of digitalization, the concept has become popular as online platforms allow the connection of customers and assets, making it easier to use, more convenient, and more efficient [Shaheen, 2018]. While unorganized carpooling (i.e. private sharing of trips with, for example, family or friends) has been established for a long time, organized carpooling initially emerged in the 1940s after the Second World War in the US as the access to private vehicles was limited. Starting in the 1960s, carpooling was promoted by US and European governments, which was enhanced in the 1970s to reduce fuel consumption caused by the oil crisis [Shaheen, 2018; Lukasiewicz et al., 2022]. Nevertheless, with the ease of the oil market and growing accessibility to private vehicles, carpooling started to decline again in the 1980s [Shaheen, 2018].

With upcoming digitalization and social connectivity through the internet, the "sharing economy" has been on the rise, which is best known for facilitating peer-to-peer exchanges through digital platforms and mobile communication [Miguel et al., 2022]. Business models for the sharing economy include sharing of accommodation (e.g., Airbnb), reselling of goods (e.g., eBay) or open-source-software (e.g., Linux).

Shared mobility is one of the segments of the sharing economy with great disruptive potential, especially in urban transportation systems, due to the increased rates of motorization and the number of private vehicles. Many different shared mobility business models have emerged

in recent years with the intention of increasing the utilization of vehicles. Figure 2.1 shows an overview of the ecosystem of shared mobility services today, which can be classified into two main categories: Services to share a vehicle or services to share a trip. The closest category to the topic of this thesis is ride-pooling services, which are featured by sharing of rides by multiple users of an on-demand ride service. To put a focus on this category, the key categories (bold frames in Figure 2.1) of shared mobility services are discussed in more detail in the following.

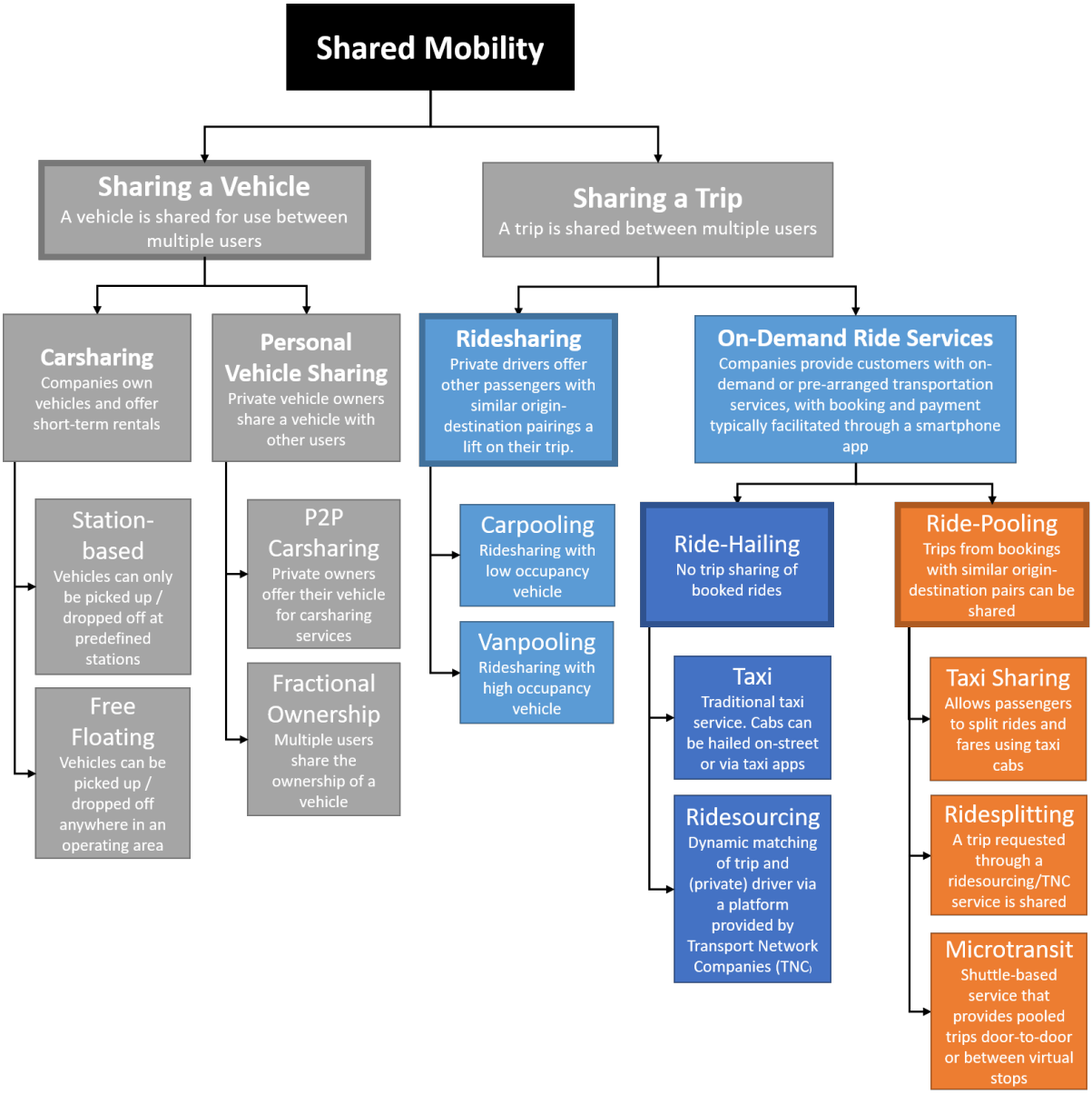


Figure 2.1: Ecosystem of car-based shared mobility services. Based on [Machado et al., 2018; Shaheen and Cohen, 2019]. Bold boxes indicate categories that are discussed in more detail in the text.

Sharing a Vehicle

The first category includes carsharing services, where vehicles are provided by companies for short-term rental. The vehicles can usually either be parked and picked up at predefined stations (station-based carsharing) or anywhere in a predefined operating area (free-floating carsharing). Well known examples for these services are *ShareNow*, *MILES*, or *Zipcar*, but distinct local players emerge in many cities. In Germany alone, the number of registered carsharing users increased from 200k in 2010 to 4.5 million in 2023, with an increase in fleet size from 5 thousand to 34 thousand vehicles, respectively [BCS, 2023]. Nevertheless, compared to an estimated market volume of 70 billion euros for all shared mobility services in Europe in 2022, carsharing only accounts for 3 billion euros [McKinsey, 2022]. Additionally, also shared mobility options emerged where private vehicles are shared. This includes platforms like *SnappCar* or *Getaround*, where private cars can be offered and used like a carsharing service. These services are also referred to as Peer-to-Peer (P2P) carsharing. Lastly, when multiple users share the ownership of a vehicle, this is referred to as fractional ownership.[1]

Ridesharing

Ridesharing is one category within the branch of sharing a trip. The critical feature of ridesharing is that not only passenger trips but also the trips of the driver are shared, i.e., the driver also wants to share costs or possibly drive duties for a personal trip. Historically, this includes unorganized ridesharing, where different parties are related through personal networks (e.g., family, friends, colleagues) and make ad hoc arrangements to share a trip. With smartphones and the internet, organized ridesharing services emerged, where trips between strangers can be matched through a digital platform. A well-known example of this is *BlaBlaCar*, which is mainly used for long-distance trips. Depending on the vehicle size, ridesharing can be further classified into carpooling and vanpooling, with the latter being more common for commuting trips that a company might organize.

Ride-Hailing

Ride-hailing is a subgroup within the on-demand ride services category. In contrast to ridesharing, a dedicated driver provides the service for a trip on-demand. This driver has no inherent interest in the trip but provides the service for a fee to transport passengers/customers. One usually distinguishes between services where trips between multiple customers can be shared or not, which is, in this thesis, referred to as ride-pooling and ride-hailing, respectively.

Taxi Classical taxi services are available in most cities and are an example of ride-hailing services. Traditionally, taxis can be hailed on the street or dispatched via phone call, but they are now increasingly accessible through mobile apps, too. The process of hailing a taxi through a mobile app is also referred to as e-hailing, which can also allow payment processing. Taxis, as in some countries viewed as part of public transport, are often subject to regulation by the government, including licensing requirements, fare regulation, or vehicle requirements.

[1] Also, classical car rental services are part of the sharing economy, but as they are typically not used for a single trip or short-term rentals, they have been excluded from this discussion of shared mobility.

Ridesourcing In contrast, Transportation Network Companies (TNCs) like Uber, Lyft, or Didi Chuxing emerged in the recent years. Originally, Lyft intended to provide a ridesharing app to connect drivers and passengers with an included payment option to share the costs of the driver. Nevertheless, the convenient payment method established a business model for drivers to provide passenger rides. Instead of only taking passengers for a lift on the route of a personal trip, platform drivers started to offer on-demand ride services for passengers in their private cars. While these companies are still often referred to as ridesharing companies, the term ridesourcing is more appropriate because most drivers no longer have an inherent travel demand. This platform-based matching of private drivers and passengers provided some advantages compared to traditional taxi services, leading to a rapid growth of these companies, particularly in relatively unregulated markets like US or China. Smartphone-based bookings allow a convenient process for drivers and passengers concerning route guidance and payment [Shaheen, 2018]. Additionally, the platform features dynamic pricing of the service to balance supply and demand. Due to operational efficiency and flexible driver arrangements, ridesourcing services can often offer cheaper fares than traditional taxi services, These advantages led to a vast growth in ridesourcing usage. Uber, for example, was founded in 2009 and reported an increase in trips from 3.0 billion in 2017 to 7.6 billion in 2022 [WallStreetZen, 2023]. Compared to estimated user penetrations of 34.8% and 26.2% in China and the US in 2023, respectively, a varying user penetration of, for example, 11.9% in Germany 26.7% in the United Kingdom was reported [Statista, 2024]. These differences in European countries are mainly due to regulatory restrictions aiming to protect the traditional taxi market as well as environmental concerns, and high quality public transport services [Fageda, 2021; Gomez et al., 2021].

Impacts of Ride-Hailing Besides its success and easy access to mobility, critique started to grow on ridesourcing services. On the one hand, the working status of drivers is debated, as they are often classified as independent contractors and therefore do not have the same rights as regular employees [Chen et al., 2017]. On the other hand, traffic efficiency is a major concern. As ride-hailing does not opt for sharing trips, a higher traffic efficiency than private vehicle trips is hardly achievable. On top, additional vehicle kilometers are produced by deadheading of idle drivers between trips and approaches to customer pick-up. In fact, Wenzel et al. [2019] evaluated ridesourcing trip data for Austin, Texas, and found 26% of Vehicle Kilometers Traveled (VKT) are associated with idling and pick-up trips, while another 19% of the overall VKT is caused by driver commuting trips. In a study for the Denver region, Henao and Marshall [2019] even estimated a deadheading ratio of 41%. Erhardt et al. [2019] found that ridesourcing services are the main contributors to traffic congestion in San Francisco by comparing traffic data of 2016 with a reference base case in 2010 without ridesourcing. The study by Hall et al. [2018] evaluated the impact of ridesourcing services on public transport ridership in the US. They found an increase in public transport ridership in larger, densely populated cities, while a decrease in ridership was observed in smaller cities.

Ride-Pooling

Ride-pooling services are a special kind of Mobility-on-Demand (MoD) services where multiple passengers are transported in one vehicle and share part of the ride. The goal of these services is to increase the occupancy and utilization of vehicles for a more sustainable way of transport compared to ride-hailing services, while passengers can split the cost of a ride. Usually, these services require app-based bookings, which allow an algorithm to compute and update routes for drivers and vehicles in real-time. Depending on the originating service platform, ride-pooling can be distinguished into three main categories: 1) Ridesplitting, where trips of ridesourcing services are shared. 2) Taxi sharing services, where trips of taxi rides are shared. 3) Microtransit, a distinct service often embedded into the public transport system.

Ridesplitting Ridesplitting is the shared ride option of ridesourcing services. *Uber*, for example, offered the variant *UberPool* while similarly *Lyft* offered *LyftLine*. Within the same app, customers could select this cheaper option for a trip, but users who opted for this option could be matched to share part of the ride. Launched in 2014, *UberPool* was available in 36 cities in 2017 (mainly US, Latin America, Toronto, London and Paris) and reported that twenty percent of the trips were pooled [Shaheen and Cohen, 2019]. For Toronto, Young et al. [2020] reported that only 15% of all trips had been made with *UberPool*. Abkarian et al. [2022a] evaluated TNC data for Chicago, which includes trips from Uber, Lyft, and Via between 2018 and 2019. They found that 20%-30% of all trips were selected under the shared ride option, but they also observed a drop in choosing the shared ride option from 30% to 18% between November 2018 and January 2020. Nevertheless, *UberPool* as well as *LyftLine* have been shut down during the pandemic in 2020, but Uber relaunched the service in 2022 under the name UberX Share [New York Post, 2022].

Microtransit Alternatively, microtransit services emerged, focusing solely on the sharing of trips. These services are sometimes referred to as Demand Responsive Transit (DRT) or ride-pooling services. Companies that offer these services usually operate their own fleet of vehicles and employ dedicated drivers. Some of these services operate on a stop-based level, where customers are picked up or dropped off at either physical or virtual predefined stops instead of door-to-door services. Often, these services are integrated into the public transport system and are either operated by the public transport operator itself or by a private company in cooperation with the public transport operator. In the latter case, usually, private companies provide the software for routing and dispatching of vehicles, while the public transport operator provides the vehicles and drivers. Well-known examples of these services are *ViaVan*, *MOIA*, and *IOKI*. Except for a slight decline during the pandemic, Foljanty [2022] reported a steady increase in new service launches from 2015 to 2021. Nevertheless, as most of these launches were government-funded projects, these services usually operate a quite small fleet with less than ten vehicles.

Impacts of Ride-Pooling A central requirement for the success of ride-pooling services is to find sharable trips, thereby increasing overall vehicle occupancy. Nevertheless, for the ridesplitting *UberPool* service for Toronto, Young et al. [2020] found that only 52% of the

trips were actually shared. For Denver, Henao and Marshall [2019] evaluated a distance-weighted average occupancy of 1.3 for *UberPool* and *LyftLine* trips. When they also accounted for deadheading and idling trips, the average occupancy decreased to 0.8. For the microtransit services *MOIA*, Kagerbauer et al. [2021] evaluated the impact of the service in Hamburg. Based on a simulation study calibrated on the status quo of the service, they found an average vehicle occupancy of 1.33, including idle vehicle trips. Nevertheless, with an estimated model split of 0.11% in Hamburg, the traffic impact is yet negligible.

2.1.2 Autonomous Vehicles

Seen as science fiction a few years ago, autonomous vehicles are on the verge of becoming reality. SAE [2021] defines six vehicle automation levels (from 0 to 5). The first three levels describe driver assistance systems, where the driver must constantly supervise the vehicle. Currently, most vehicles on the road are equipped with systems up to level 2. These driver assistance systems include lane-centering systems or adaptive cruise control. From level 3 on, the vehicle takes control over the driving tasks. As this represents a shift in liability from the driver to the vehicle, most Original Equipment Manufacturers (OEMs) yet hesitate to offer cars with these systems. Nevertheless, many OEMs claim that their vehicles are already capable of performing level 3 driving tasks, but instead of shifting liability to the vehicle, they refer to these systems as level 2+ systems. An example of a level 3 system is a traffic jam chauffeur, where the vehicle can handle the driving task in traffic jams on highways, but the driver still has to step in when requested by the car. In 2022, *Mercedes* became the first manufacturer to get approved by the German transport authorities to operate their level 3 system in Germany [Auto Motor und Sport, 2023] and is the first OEM selling level 3 vehicles in California and Nevada in 2024 [Fortune, 2024].

With level 4 and 5 systems, the vehicle is capable of driving without any intervention of the driver. While level 4 systems are limited to a specific operational design domain, level 5 systems are capable of driving in any situation. On the contrary to previously mentioned developments, some tech companies emerged that focus on the automated driving stack to directly develop level 4 and 5 systems. Well-known examples are *Waymo*, *Cruise*, or *Mobileye*. To test their technology in a confined environment, *Waymo* and *Cruise* started offering automated ride-hailing services (robotaxis). Waymo One, the robotaxi service of *Waymo*, has been available in the Phoenix metropolitan area since 2018 and expanded to San Francisco in 2021 [Waymo, 2024]. *Cruise Origin*, the robotaxi service of *Cruise*, is available in San Francisco since 2021 [Cruise, 2024]. After periods of testing with safety drivers and limited access, both services are now available to the public, while *Cruise* became the first service for commercial use [Kolodny, 2022]. A similar trend is observed in China, where services by *Baidu* or *AutoX* have begun offering robotaxi services in multiple cities for commercial use in 2022 [Magramo et al., 2024; AutoX, 2024]. In the meantime, also *Mobileye* received approval to test their robotaxis on German roads in 2023 [TÜV Süd, 2023].

2.2 Potential of Autonomous Ride-Pooling Services

In theory, autonomous vehicles have the potential to disrupt the transportation system. Mainly three reasons contribute to that: 1) Increased safety, 2) increased traffic efficiency, and 3) reduced costs.

Human error is the main cause of traffic accidents, which might be the cause of around 90% of all road accidents [Treat et al., 1979; Winkle, 2016]. Removing these human errors by automation can, therefore, greatly impact traffic safety.. Wang et al. [2020] studied traffic accidents of currently publicly operating AVs and found that only 6% of reported accidents were directly caused by the AV. In the long run, for example Shetty et al. [2021] and Ye and Yamamoto [2019] argue that not only human errors (i.e., impaired, reckless, or distracted driving) might be removed, but also errors caused, for example, by occluded vision or limited perception if vehicles are not only automated but also connected to other vehicles or infrastructure to enhance their field of view.

Next to increased safety, automation also has the potential to improve traffic efficiency, leading to higher throughput on roads and networks but also to a decrease in energy consumption. AVs can reduce the gap between the following vehicles drastically by minimizing reaction times to near zero. Even further gain can be expected if AVs are connected to each other and to local infrastructure. Coordinated and predictive driving can increase road capacity by stabilizing traffic, reducing the need for braking and accelerating and thereby reducing energy consumption (e.g., Kesting et al. [2010], Talebpour and Mahmassani [2016], and Motamedidehkordi et al. [2016]), and allow more efficient intersection control strategies (e.g., Friedrich [2016] and Niels et al. [2020]). Nevertheless, Talebpour and Mahmassani [2016] and Calvert et al. [2017], for example, showed that unfolding the full potential strongly depends on the penetration and connectivity rate of AVs. An automated vehicle is supposed to be more cautious and obedient to traffic rules compared to an average human driver, which might lead to a more defensive driving style and thereby to reduced traffic efficiency. Only a high penetration rate and connectivity of AVs can circumvent this issue.

2.2.1 Costs for Autonomous Ride-Pooling Services

From the viewpoint of ride-pooling operators, cost reduction by automation resulting from replacing human drivers is the most essential aspect. Human drivers are the main cost factor for on-demand ride services. Ridesourcing services usually withhold 15% to 30% of the fare as commission, resulting in a net driver cost of 70% to 85% of the fare [Wang and Yang, 2019]. If this cost factor can be removed, the service can be offered at a substantially lower fare, potentially skyrocketing their future use[2].

Nevertheless, when automation comes into play, not just the drivers' costs are removed, but other cost structures might change as well. Automated vehicles are expected to be more expensive than conventional vehicles, requiring additional sensors. Additionally, personnel are needed for maintenance, cleaning, and monitoring of vehicles during operation might be necessary. But automation might also allow a more efficient utilization of vehicles due to

[2]In ridesourcing services, drivers stem the operational costs (i.e., fuel and vehicle maintenance). Therefore, a direct transfer to fare reduction is not possible.

Study	Method	Cost Components	Pooling	Location	Estimated Passenger Cost
Friedrich and Hartl [2016]	S	vehicle purchase, operation cost, maintenance cost	✓	Stuttgart, Germany	0.15 €/km
Chen et al. [2016]	LR + S	vehicle purchase, operation cost, pricing methods, profit margin	✗	USA	75-100 ct/mile (0.44 - 0.58 €/km*)
Fagnant and Kockelman [2018]	S	vehicle purchase, operation cost, 19% profit margin	✓	Austin, Texas	100 ct/mile (0.58 €/km*)
Lim and Tawfik [2018]	LR	vehicle purchase, operation cost, finance, insurance, maintenance, license, advertisement revenue, short and long term horizon	✓	USA	≤20 ct/mile (≤0.12 €/km*)
Bösch et al. [2018]	LR	vehicle purchase, operation cost, finance, insurance, parking and toll, maintenance, cleaning	✗	Zurich, Switzerland	0.41 CHF/km (0.42 €/km*)
Loeb and Kockelman [2019]	LR + S	vehicle purchase, operation cost, maintenance, batteries, charger construction/maintenance, insurance and registration, administration	✓	Austin, Texas	29-89 ct/mile (0.17-0.52 €/km*)
Dandl and Bogenberger [2019]	S	same cost structure as car sharing service	✓	Munich, Germany	0.25-0.27€/km
Compostella et al. [2020]	LR	vehicle purchase, operation cost, insurance, charger, batteries	✓	USA	0.18 - 0.21 $/mile (0.10-0.30 €/km*)
Becker et al. [2020]	LR	vehicle purchase, operation cost, finance, insurance, parking and toll, maintenance, cleaning	✓	Multiple Cities Worldwide	0.08 - 0.41 $/km (0.05-0.24 €/km*)
Tirachini and Antoniou [2020]	LR + AM	vehicle purchase, operation cost, finance, insurance, parking and toll, maintenance, cleaning	✓	Munich, Germany; Santiago, Chile	0.22 - 0.67 €/km**
Negro et al. [2021]	LR + S	relative changes in fixed and marginal costs, cost for personnel	✓	Munich, Germany	0.42 €/km

Table 2.1: Comparison of studies determining potential cost per passenger distance. Abbreviations: LR: Literature Review; S: Simulation-based; AM: Analytical Model. *Based on a conversion factor of 1.07$/€ and 0.97CHF/€, **Values refer to operating costs.

central control of the fleet further reducing cost.

Table 2.1 gives an overview of studies that estimated the potential cost per passenger-distance traveled for automated ride services. The studies differ in their approach and the cost components they include. Generally, as many components are not yet known, these studies show a large range of possible costs. Nevertheless, all the studies agree that fares for these automated services can be drastically decreased compared to today's fares for comparable services. For comparison, the current km fare for a taxi trip in Munich, Germany is at 2.50 €/km, which is 3.7 times higher than the highest estimated value in table 2.1 [Taxi-München eG, 2024].

2.2.2 Simulation Studies for Autonomous Mobility-on-Demand

With this potential drastic reduction of fares, the question arises as to how these Autonomous Mobility-on-Demand (AMoD) services might impact future transportation systems. As these

services are not available yet, and therefore, no empirical data exists, simulation studies are a common approach to evaluate their potential to deal with complex interactions arising in transportation systems. Here, highlights of the results of these studies are presented. Technical details regarding modeling approaches will be discussed in the following sections.

These simulation studies are data intensive and, therefore, often focus on a single city or region with available data. Besides the choice of the study area, studies mainly differ in modeling the AMoD service and its assumed demand for the service. Table 2.2 gives an overview of studies that evaluated the potential of AMoD services. The main differences between are

1. whether they allow the pooling of trips,
2. the assumptions on the demand for the service. As this is the main source of uncertainty, many studies apply "what-if" scenarios where a specific set of current trips is replaced by the AMoD service. More sophisticated models include a mode choice model, i.e., travelers can choose between different modes of transport for their trips.
3. the number of trips that are finally assigned to the AMoD service.

The studies in table 2.2 that only consider ride-hailing services mainly evaluated the higher utilization of vehicles when operated in a fleet and dispatched by a central controller. When private vehicle trips are replaced, the studies agree that a single AV in the ride-hailing fleet can substitute up to 13 private vehicles while the service is available within a few minutes of customer waiting time. Inner-city operation, therefore, has a great potential to reduce space consumption for parking. While this potential mainly stems from higher temporal utilization of the vehicles, the aspect of central control of the fleet is also essential. Zhan et al. [2016] showed in their case study for New York City that by central coordination of a taxi fleet, around 2/3 of the taxis could be removed while still serving the same number of trips. This effect mainly stems from removing idling trips of taxis when looking for customers. They evaluated that 90% of empty pick-up trips could be avoided by central coordination. Nevertheless, as trips are not shared, a fraction of 10-20% of VKT is still attributed to deadheading. Stress to the road network can additionally be expected if the AMoD service replaces not only private vehicle trips but also trips that are currently performed by public transport or if the AMoD service even induces new trips due to its low fare and high availability.

Ride-pooling aims to reduce this additional induced VKT from empty vehicle trips. Studies that consider ride-pooling services, therefore, emphasize on the reduction of VKT. Some studies evaluate the replacement of a large proportion of private vehicles' trips by the ARP service and find a huge potential to reduce VKT. For example, Fiedler et al. [2018] and Zwick et al. [2021b] estimated a reduction of 60% and 54% in VKT if all private vehicle trips are replaced in their case study for Prague, Czech Republic, and Munich, Germany, respectively. Compared to Zhan et al. [2016], Alonso-Mora et al. [2017b] evaluated the substitution of taxi trips with a ride-pooling service. They found that only 2,000 vehicles are required to serve the same number of trips as 12,000 taxis in Manhattan, US. Nevertheless, the mentioned studies focused on scenarios with a very high demand for the service, facilitating many shared trips. Fagnant and Kockelman [2018] and Engelhardt et al. [2019] evaluated a lower penetration rate of private vehicle trips that are shifted to the ARP service. They found that the potential to

reduce VKT is much lower in this case, and the ARP service might even increase VKT. This effect occurs because not enough shareable trips can be found to overcome the additional VKT caused by empty pick-up trips.

As these results highly differ depending on which demand is assumed for the service, it is crucial to evaluate scenarios that include potential shifts from public transport users to the ARP service. The early studies ITF [2015] and Friedrich and Hartl [2016] evaluated the replacement of public transport trips by the ARP service in Lisbon and Stuttgart, respectively. Their results state clearly that it is not sustainable to substitute high-capacity public transport with the automated service even if the service is operated with a high volume of shared trips. In a study for Munich by Zwick et al. [2021b], travelers could decide on their mode of transport based on a mode choice model. After introducing the ARP, they evaluated a slight increase in system VKT because of a shift from public transport to the service. Kagerbauer et al. [2021] concluded in their study for Hamburg, Germany, that an improvement of the traffic system can only be achieved if regulatory measures are taken, like removing parking spots and reducing private vehicle ownership in the long run. The ARP service can then bridge the gap from the decreased availability of private vehicles in the city. Oke et al. [2020] simulated the introduction of an ARP service in artificial city archetypes. These city archetypes were based on typical population structure and public transport supply in cities worldwide. They evaluated "auto-sprawl" and "auto-innovative" cities, both relying heavily on private vehicles while the latter is characterized by a slightly higher mode share of public transport. These city types, therefore, differ from the previously mentioned studies, which are based on cities with a strong public transport system. They concluded that replacing public transport trips by the ARP service is only sustainable in the "auto-sprawl" archetype.

Study	Location	Demand	Number of AMoD Trips	Pooling	Main Findings
Fagnant and Kockelman [2014]	Austin, USA	RP of PV trips	60k	✗	▪ Each AV can replace conventional 12 PVs ▪ Increase in VKT by 10%
Bischoff and Maciejewski [2016]	Berlin, Germany	RP of PV trips	1.1 million per day (All PV trips)	✗	▪ 100k vehicles needed ▪ Each AV can replace up to 10 conventional PVs
Zhan et al. [2016]	New York City, USA	RP of Taxi Trips	400k to 500k per day	✗	▪ 2/3 of all taxis could be removed ▪ 90% of empty VKT can be avoided
Dandl and Bogenberger [2019]	Munich, Germany	RP of Carsharing Trips	N/A	✗	▪ AV can replace 2.8 to 3.7 Carsharing vehicles ▪ Fares can be reduced by 29-35%
Hörl et al. [2019]	Zurich, Switzerland	MC	Up to 360k per day	✗	▪ 7k to 14k vehicles needed ▪ Resulting fare of 0.5CHF/km ▪ Compared to 0.26 CHF/km variable PV cost and 0.7 CHF/km full costs
Liu et al. [2017]	County of Austin, USA	MC	Up to 4.5 million per day	✗	▪ Modal split for AMoD of 50.9% and 9.2% for fares of 0.5$ per mile and 1.25$ per mile, respectively
ITF [2015]	Lisbon, Portugal	MA	N/A	✓	▪ Each AV can replace around 10 PVs ▪ ARP service not sustainable to replace high capacity PT
Friedrich and Hartl [2016]	Region of Stuttgart, Germany	MA	Up to 5.1 million per day	✓	▪ ARP vehicle can replace 12-13 PVs ▪ Traffic only improves if most trips are shared ▪ Increase in traffic volume if PT is removed
Alonso-Mora et al. [2017a]	New York City, USA	RP of Taxi Trips	Up to 460k per day	✓	▪ 2,000 10-seater ARP vehicles can serve demand of 12,300 taxis with mean waiting time below 3 min.
Fiedler et al. [2018]	Prague, Czech Republic	RP of PV Trips	130k within 1.5h (All PV trips)	✓	▪ Average vehicle occupancy of 2.7 ▪ VKT decreased by 60% ▪ Number of heavily loaded road segments reduced from 208 to 35
Fagnant and Kockelman [2018]	Austin, USA	RP of PV Trips	56k per day	✓	▪ VKM increases if 10% of trips served with ARP service ▪ Without trip sharing, further increase by 8% of VKT
Engelhardt et al. [2019]	Munich, Germany	RP of PV trips	Up to 180k per day (15% of PV trips)	✓	▪ 3,000 vehicles can serve 15% PV demand within average waiting time of 4 min ▪ 3-5% PV trip penetration rate needed to achieve VKT savings ▪ VKT savings mainly on major roads
Vosooghi et al. [2019]	Rouen, France	MC	Up to 84k per day	✓	▪ Best result for ride-pooling fleet with 4 seats ▪ Modal split of ARP service below 7.6%, but modal split for PV remains at around 58% ▪ Overall VKT increases
Oke et al. [2020]	City Archetypes	MC	N/A	✓	▪ ARP RP of PT sustainable in cities with weak PT ▪ not sustainable in cities with modest PT ▪ PT integration beneficial for both city types
Zwick et al. [2021b]	Munich, Germany	RP of PV trips vs. MC	Up to 1.9 million per day	✓	▪ 18k 6-seater vehicles needed to replace all PV trips ▪ VKT reduced by up to 54% for RP ▪ Slight increase in VKT in MC ▪ Traffic noise can be reduced if public transport stops are used for boarding
Kagerbauer et al. [2021]	Hamburg, Germany	MC	Up to 1.2 million per week	✓	▪ ARP does not cannibalize PT ▪ ARP does not improve traffic state on its own ▪ push measures for PV needed

Table 2.2: Collection of simulation studies evaluating the potential impact of AMoD services. Abbreviations: RP: Replacement; MA: Mode Allocation (rule-based pre-assignment of trips to modes); MC: Mode Choice Model; PT: Public Transport; PV: Private Vehicle

2.2.3 Characteristics of Sharing Rides

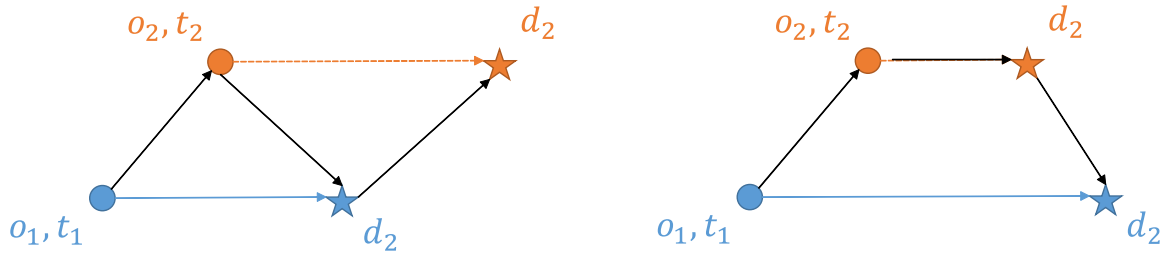


Figure 2.2: Two of four possible shared routes between two trips. The other two options are obtained by starting the route at the second (orange) trip.

As the case studies reveal, the potential to share rides is critical for the success of ARP services. Therefore, researchers tried to quantify characteristics that determine under which circumstances the sharing of rides can be successful. A quantity often used to describe the potential of sharing rides in a given system is referred to as "Shareability". To evaluate this quantity, it is necessary to first define under which circumstances trips are shareable. Given a trip i, specified by its origin o_i, destination d_i, and time of trip t_i, a common approach is to define that two trips are shareable if a route can be formed that serves both trips, while the total travel time and the pick-up time deviation from the original time of the trip is below a specific threshold. The pick-up time deviation is hereby defined as the difference between the trip's original time and when the passenger is picked up. Depending on the order of pick-ups and drop-offs, there are four possible options to share two trips as sketched in Figure 2.2. If one of these options fulfills the constraints, the trips are considered shareable.

Santi et al. [2014] defined shareability networks by connecting trips in a given dataset that are shareable. They calculated shareability by matching two trips in this network and computed the ratio of trips that were assigned to a partner trip. In their case study for taxi trips in New York City, they found that at approximately 100k daily trips (around 25% of the daily average) the shareability reaches close to 100% while an increase in travel time is limited to 5 minutes for these trips.

Tachet et al. [2017] evaluated the shareability [3] of taxi trips in four cities worldwide. Based on data from New York, San Francisco, Singapore, and Vienna they found that the Shareability curve (Shareability versus trips per hour) can be collapsed to a single curve by some rescaling factor, which they argue is a universal law. By proposing an analytical model, they could show that this scaling factor only depends on measurable quantities, like average velocity, allowed detour, and size of the operating area. In theory, this model could be used by planners to estimate the potential of a ARP service for a given city. Bilali et al. [2019b] further extended the model by Tachet et al. [2017] to include ride-pooling service parameters like maximum customer waiting time and boarding time, allowing a provider to estimate the design parameters of its service. Additionally, Bilali et al. [2019a] extended the model to include short-term reservations. Bilali et al. [2020] evaluated the possible real-world application

[3] Tachet et al. [2017] had a slightly different definition of shareability compared to Santi et al. [2014]: Instead of performing a trip matching step, all trips with a shareable partner are included in the calculation, resulting in a higher shareability.

of this model by comparing the model results to a simulation study for the city of Munich, Germany. They found that the simulation reproduces the analytical model well as long as the same assumptions are made in both models. Nevertheless, as modeling details increase, the simulation results deviate from the analytical model. Especially when evaluating the fraction of actual shared rides instead of the shareability, deviations are prominent because many trips are not beneficial to be shared (especially concerning required detours) even though sharing would be possible within given constraints.

To overcome this issue, Kucharski and Cats [2020] proposed a different approach to calculate shareability networks. Instead of relying on waiting and detour constraints, they propose trip matching based on traveler utilities, which they refer to as "attractive" shared rides. Their idea is to compare the utility of a shared ride, computed from waiting and travel time together with the fare of a ride, with the utility of a solo trip. If the utility of a shared ride is higher for all customers, the trip is considered attractive. In their case study for trips in Amsterdam, Netherlands, they found that from 3,000 customers, 1,900 form attractive rides if a 30% discount on the fare is offered for shared trips. In another study evaluating the impact of demand patterns, the authors found that more trips become attractive if trip origins and/or destinations are clustered in a few areas, for example, city centers, and trips tend to be longer [Soza-Parra et al., 2022]. The discount parameter as a control parameter in this formulation is critical. For a case study of New York City, they evaluated that this parameter has to be set to at least 10% to achieve attractive shared rides [Bujak and Kucharski, 2023]. Instead of matching actual trips, Sarma and Hyland [2024] proposed a method to calculate the shareability of traveler flows by defining the Maximum Network Flow Overlap Problem.

2.3 Modeling and Operating Autonomous Ride-Pooling Services

As described in the previous section, simulation studies have proven indispensable to studying AMoD services. These simulations can not only be used to evaluate large-scale impacts of the service, but also to evaluate fleet control algorithms and other strategies for operation. Agent-based simulations are a common approach to simulate AMoD services. In agent-based modeling, the goal is to model the behavior of individual agents and their interactions in a given environment to understand the system's behavior as a whole. In the case of AMoD services, typical agents are the travelers, the operators offering the service, and their fleet vehicles. As simulations can only provide a simplified representation of the real world, it is crucial to carefully select the level of detail of the models to manage the complexity of the simulation.

These aspects are not only relevant for modeling itself but are also central for the real-world design of AMoD services. For example, before deploying the service, it has to be defined within which area the service is offered, how many and which kind of vehicles are employed and how the service is operated. As pointed out by, for example, Dandl [2022], one can generally distinguish between strategic (long-term) and operational (short-term) decisions. Operating area and fleet composition are examples of strategic decisions, as changes in these decisions

are usually made on monthly to yearly time scales because high costs might be associated. On the other hand, the fleet control algorithm is an operational decision, as dispatching tasks are updated on a minute to second scale.

The different aspects that have to be considered when modeling AMoD services are depicted in a (not exhaustive) sketch in Figure 2.3. The following subsections give an overview of these aspects and the different approaches for modeling and determining that have been proposed in the literature. The core of the simulation – the fleet control algorithm – is discussed in detail in the next section, as this is a central aspect of this thesis.

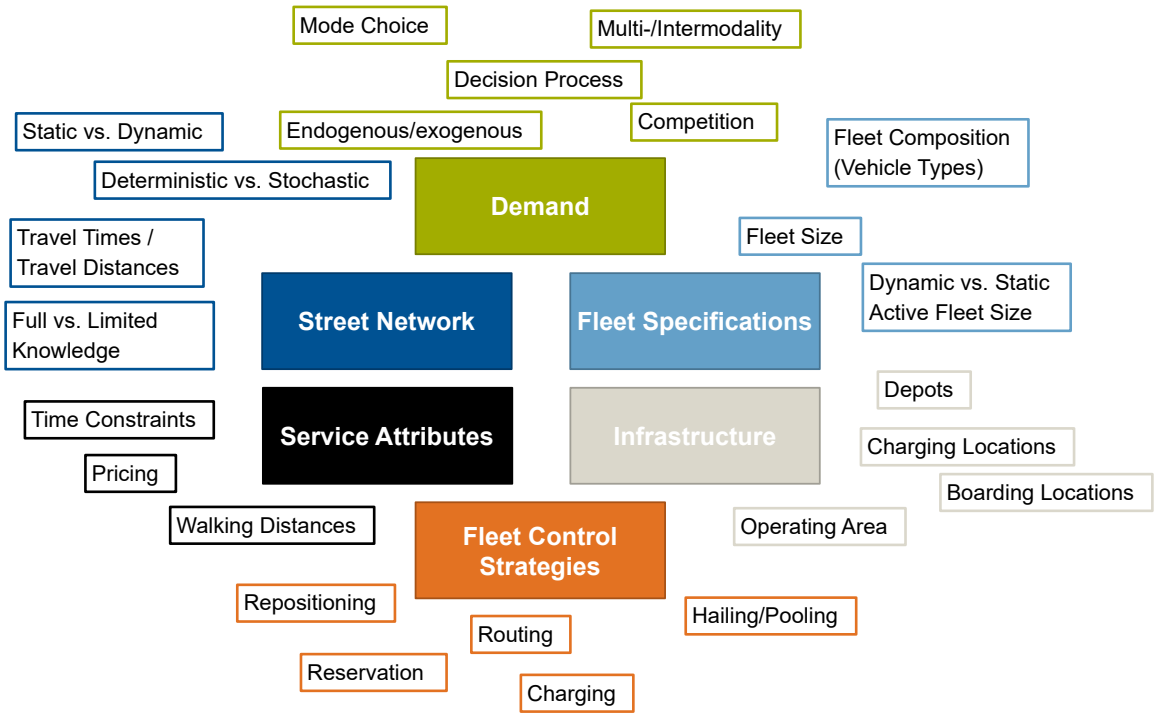


Figure 2.3: Essential categories for simulating ARP services.

2.3.1 Street Network and Traffic State

The representation of the street network is a central aspect of the simulation, but also for operation. The street network is the environment in which the service operates and, therefore, determines the possible vehicle routes. Additionally, travel time estimations between locations in the network are central to evaluating feasible matches between vehicles and customers. Due to the complex interactions between vehicles, the street network, and traffic control infrastructure, traffic state estimation, which estimates the utilization of the street network given a set of measurements is a research field on its own and can be highly complex. From a simulation standpoint, it is, therefore, essential to carefully choose the level of detail of the street network representation for the research question at hand. From an operation standpoint, choosing a feasible representation to be implemented in real-time operation is necessary. As will be discussed in detail later in this thesis, finding suitable vehicle routes involves many

travel time queries. Therefore, the street network's representation should be chosen so that travel time queries can be performed efficiently.

Early studies, therefore, represented vehicle movements in a purely geometrical plane. Fagnant and Kockelman [2014], Hyland and Mahmassani [2018], and Dandl et al. [2019] used a grid-based representation of the street network. As they investigated case studies for the US cities Austin and New York City, a regular grid with a Manhattan metric was used as an approximation of the real street network, allowing very fast computation of travel time. Alternatively, representing the street network as a graph allows a more realistic representation of the street network. In these graphs, nodes usually represent intersections, and edges represent street segments. Travel time attributes are assigned to the edges, which can be used to encode varying traffic states. The studies Alonso-Mora et al. [2017b] and Erdmann et al. [2021] used a graph representation of the street network for their case study of Manhattan, New York City. They estimated daily mean edge travel times based on historical taxi data. As the representation of the Manhattan street network is small enough and edge travel times are deterministic, they could precompute all travel times and, therefore, perform travel time queries very efficiently. To include also temporal varying traffic states, Dandl et al. [2020b] and Markov et al. [2021] used network-wide travel time scaling factors based on historical taxi data. As they employed the model for the larger case study of Chicago, they had to remove minor roads to still be able to preprocess all node-to-node travel times. Engelhardt et al. [2019] and Ghandeharioun and Kouvelas [2023] regularly update edge-specific travel times during the simulation. Syed et al. [2023] proposed a method to estimate edge travel times based on historical taxi data for AMoD simulations.

Nevertheless, the described studies assume that network travel times are deterministic, known to the operator, and not influenced by the service itself, which is not the case in reality. Dandl et al. [2017], Gueriau et al. [2020], and Wolf et al. [2023] used a microscopic traffic simulation model to overcome these limitations. While microscopic models can capture all the mentioned aspects, they are computationally costly and complex to calibrate for large-scale models. Therefore, Dandl et al. [2021b] used a dynamic model based on the Macroscopic Fundamental Diagram (MFD) to capture the traffic dynamics computationally efficiently.

To represent the influence of fleet vehicles on the traffic system, Zhang and Pavone [2016] used a queuing model to represent the network while Levin et al. [2017] proposed a link transmission model. Levin et al. [2017], Salazar et al. [2019], and Huang et al. [2022] thereby proposed optimization models for congestion-aware routing: Instead of routing vehicles on the shortest or fastest path, they aim to distribute vehicles on the network to reduce overall congestion. Nevertheless, as routing queries are a crucial component of the assignment process, these models are computationally expensive.

2.3.2 Demand and Travelers

Also, the representation of demand and travelers can be chosen from different levels of detail. The most straightforward representation assumes a fixed or static demand, i.e., a fixed set of customers for the service. Alternatively, demand can be created endogenously. In this case, travelers are generated by a demand model that often represents general trips across multiple modes of transport. Trips for the AMoD service are calculated by determining a user

equilibrium, given the interaction of demand and supply. Further differentiation can be made by the time customers wait for the operator to confirm a trip. Customers are often referred to as "patient" or "impatient" depending on their modeled willingness to wait. Additional categorization can be made whether customers and/or operators can decline or reject a trip offer, respectively.

Hyland and Mahmassani [2020] and Riley et al. [2019], for example, used a static demand with patient customers in their model. In case of a supply shortage at the time of a request, i.e., no vehicle is available to serve the request, customers would wait indefinitely for a vehicle to become available. While this might not be a reasonable assumption for a real-world service, this model allows to study the relation between the level of service (e.g., by measuring mean waiting times) and the deployed fleet size. On the contrary, all studies marked as "Replacement" or "Mode Allocation" in the Demand column of Table 2.2 assumed impatient customers. In this case, customers would cancel their request if no vehicle is available within a specific time after the request is made. All travelers are assumed to use the service if a vehicle is available within this time. This suggests that the operator is aware of this behavior and consequently dispatches vehicles in a manner that ensures no customer waits longer than the maximum allowable waiting time. Engelhardt et al. [2019] refined this model by introducing a multistep approach: Within the first dispatching step, the goal was to serve customers within a short waiting time, which they always accepted. If this was not possible, customers were served within a longer waiting time, which was assumed they would accept only with a specific waiting-time-dependent probability.

If the goal of the study is to evaluate the potential demand attraction of the service in a given study area, a more detailed demand model is required. A common approach is using agent-based demand models like MATSim [Horni et al., 2016], SimMobility [Yang et al., 2015], POLARIS [Auld et al., 2016] or mobiTopp [Mallig et al., 2013]. To include the ARP service into these models, two steps are necessary: 1) A mode choice model is required that includes the ARP service as a mode of transport. As the ARP service is not available yet, the mode choice model has to be based on stated preference surveys, which have been conducted, for example, by Frei et al. [2017] in the Chicago region, Morsche et al. [2019] and Alonso-González et al. [2020] in the Netherlands, König and Grippenkoven [2020] in Germany or Kagerbauer et al. [2021] in Hamburg, Germany. 2) A model of the ARP service has to be integrated into the agent-based demand model. This has been done by Ruch et al. [2018], Marczuk et al. [2015], Gurumurthy et al. [2020] and Wilkes et al. [2021] for MATSim, SimMobility, POLARIS, and mobiTopp, respectively. Nevertheless, the big disadvantages of these models are that they are data-intensive, difficult to calibrate, and computationally expensive.

2.3.3 Fleet Size and Composition

The fleet size and composition is a strategic decision for the operator. The fleet size determines the number of vehicles available for the service and, therefore, the number of customers that can be served simultaneously. The fleet composition determines the types of vehicles deployed in the service, e.g., the number of available seats.

A typical approach to determine the fleet size is to assume a fixed number of vehicles and

evaluate the performance of the service for this fleet size. Given static demand, the fleet size can be determined by the number of vehicles needed to serve a specific percentage of demand in case of impatient customers (e.g., Boesch et al. [2016], Dandl and Bogenberger [2019], and Engelhardt et al. [2022c]) or a specific mean waiting time in case of patient customers (e.g., Hyland and Mahmassani [2020] and Riley et al. [2019]). The same approach can also be used to determine vehicle capacity. For example, Alonso-Mora et al. [2017b] evaluated that a fleet of 4-seater vehicles showed the best performance for their case study of Manhattan, New York City. An increase in vehicle capacity would not lead to a significant increase in served customers.

Nevertheless, determining fleet size and composition by simulation requires numerous simulation runs, which can be computationally expensive. Vazifeh et al. [2018] provides a method based on shareability networks to estimate the fleet size for a given demand. Wallar et al. [2021] used a similar approach to estimate the required number of vehicles for an inhomogeneous fleet composition (a mix of 2- and 4-seater in their case study). Narayan et al. [2021] developed a method to estimate the required fleet size of a service offering both ride-hailing and ride-pooling. Dandl et al. [2021b] proposed Bayesian optimization to reduce the search space for simulation-based estimations.

2.3.4 Infrastructure

AMoD services require infrastructure, reflecting an additional strategic decision for the operator. AMoD vehicles are expected to be electric, likely because of financial motives [Arbib and Seba, 2017] or cities might even enforce the use of Electric Vehicles (EVs) to reduce emissions. Therefore, the operator has to provide charging infrastructure for the vehicles. Different studies developed methods to determine charging stations' locations and required capacities. Methods differ between simulation-based approaches (e.g. Chen et al. [2016]), clustering algorithms (e.g. Zalesak and Samaranayake [2021]), or optimization-based models (e.g. Vosooghi et al. [2020]). Zhang et al. [2022] evaluated the feasibility of using public charging infrastructure in the operation of an AMoD service. On an operational level, charging tasks must be integrated into the fleet control algorithm. Many studies (that consider the charging problem in their formulation) apply threshold-based charging strategies, i.e., vehicles are charged if their battery level falls below a specific threshold (e.g. Chen et al. [2016], Dandl and Bogenberger [2019], and Zhang et al. [2022]). Others propose mid-term planning strategies to schedule charging processes during low demand periods (e.g. Zalesak and Samaranayake [2021] and Dandl et al. [2020a]), integrate charging into repositioning decisions (e.g. Dean et al. [2022]), evaluate dynamic electricity prices (e.g. Estandia et al. [2021]), or use reinforcement learning to learn optimal charging strategies (e.g. Ahadi et al. [2022]).

Stops, i.e., locations where boarding processes occur, are another infrastructure element that must be considered. While many studies assume door-to-door services or even curbside pick-up, studies suggest that this operation can increase congestion, VKT, and customer in-vehicle travel times [Atasoy et al., 2015; Engelhardt et al., 2019; Zwick et al., 2021a]. Zwick et al. [2021a] suggested using existing bus stops as stops for the AMoD service to reduce these negative effects, while Goel et al. [2017] developed a maximum coverage model

to select optimal stop locations. Harmann et al. [2022] evaluated possible virtual stops based on the surrounding built environment, while Stueger et al. [2023] simulated different boarding strategies at urban intersections. A set of further studies evaluated the impact of the density of stop locations on the performance of the service. They developed algorithms for online selection of the optimal boarding location serving a set of customers [Engelhardt and Bogenberger, 2021; Fielbaum et al., 2021; Zuo et al., 2021].

2.3.5 Pricing

Pricing is a central aspect of the real-world operation of AMoD services. From a modeling perspective, the choice of the pricing model depends on the demand model applied. Many studies that use a static demand model do not consider pricing as they implicitly assume a fare cheap enough to attract the given set of customers (e.g., studies marked with MA or RP in Table 2.2). Other studies that use a mode choice model employed static pricing models based on cost estimations for the operation of autonomous vehicles (e.g. Chen et al. [2016], Liu et al. [2017], Hörl et al. [2019], and Vosooghi et al. [2019]). More refined models include surge pricing mechanisms (e.g. Zhang and Nie [2021] and Dandl et al. [2021b]), a dynamic pricing model based on the current demand and supply situation (e.g. Castillo et al. [2017]). When rides are shared, the advantage from a customer viewpoint is that the fare can also be split. Ruijter et al. [2023] evaluated different pricing strategies under the aspect that the reduced fare has to compensate for disutilities like additional detours from sharing the trip. Karaenke et al. [2023] suggested an ex-post pricing model, that charges customers based on the actual service provided, i.e., a cheaper fare is charged if a trip has been shared, which they argue can increase the willingness of customers to share trips.

2.3.6 Integration into the Mobility System

Due to the potentially disruptive nature of AMoD services, it is essential to evaluate the impact of these services on the existing mobility system and develop operational constraints to achieve societal benefits. The studies in Table 2.2 revealed that a potential displacement of high-capacity public transport should be avoided.

If the AMoD service is operated by a private company, regulation by the public authority might be necessary to ensure socially beneficial operation. Typical measures evaluated in the literature are fleet size constraints or congestion pricing to improve social welfare [Li et al., 2019b; Zhang and Nie, 2021; Dandl et al., 2021b]. Conversely, if the public authority operates the AMoD service, the service can be integrated into the public transport system. Therefore, studies developed methods to integrate the AMoD service into the existing public transport system either by complementing the existing public transport system (e.g. Sieber et al. [2020], Mo et al. [2021], and Cortina et al. [2023]) or developing frameworks to re-designing public transport networks with complementary AMoD services as feeders to high capacity public transport or replacement of currently underutilized and not economically viable bus lines (e.g. Pinto et al. [2020], Salazar et al. [2020], Auad-Perez and van Hentenryck [2022], Kumar and Khani [2022], Fielbaum and Alonso-Mora [2024], and Ng et al. [2024]).

Besides competition and cooperation with public transport, multiple operators could offer their services. As demand penetration is a critical factor for success, especially in ride-pooling services, competition between operators can lead to a decrease in service quality. Séjourné et al. [2018] and Kondor et al. [2022] studied this market fragmentation with theoretical and simulation-based models, while Pandey et al. [2019] and Engelhardt et al. [2022c] developed models for regulated platforms to foster cooperation and mitigate the effects of market fragmentation.

2.4 Algorithms for Fleet Control

Controlling the fleet of vehicles is the core of the AMoD and ARP service. The primary purpose of the fleet control algorithm is to assign vehicles to customers and determine the routes of the vehicles. As new customers request trips dynamically, the fleet control algorithm has to adopt and update its decisions on short time scales. The goal of these decisions is to optimize specific objectives in the long term, e.g., maximizing the service profit. These decisions are subject to constraints that might include, for example, vehicle capacities, time constraints for customer pick-up and drop-offs, or consistency constraints with previous decisions, e.g., a promised customer pick-up has to be fulfilled.

This section reviews the approaches proposed in the literature to solve the fleet control problem. First, classical approaches to solving the vehicle routing problem are discussed, followed by approaches to tackle large-scale assignment problems as required for ARP services. Finally, methods to incorporate information about future demand and the integration of reservations are discussed.

2.4.1 Vehicle Routing Problem

The VRP is a combinatorial optimization problem which asks "What is the optimal set of routes for fleet vehicles to serve a given set of customers?". It is a generalization of the traveling salesman problem and was first formulated by Dantzig and Ramser [1959]. VRPs are known to be Non-deterministic Polynomial-Time (NP) hard, which means that no efficient algorithm is known to solve the problem in polynomial time and, therefore, problems of arbitrary size cannot be solved optimally in a reasonable timeframe. For VRPs, this stems from the number of possible solutions (permutations of pick-ups and drop-offs) growing exponentially with the number of customers. Thus, exact solution algorithms are only feasible for small problem sizes. For larger problems, heuristics and meta-heuristics are applied to find good solutions in a reasonable time.

As solution algorithms are usually designed for specific settings, VRPs are often classified by their characteristics (e.g., the formulation of constraints). The closest variant to the VRP for ARP services is the Dial-a-Ride Problem (DARP), which is defined by a set of customers with pick-up and drop-off locations and time windows. Another variant is the Pickup and Delivery Problem with Time Windows (PDPTW) that specifies a similar setting but usually defines depots where vehicles have to start and end their route.

The DARP has been studied for decades [Psaraftis, 1980; Psaraftis et al., 2016], and a variety of solution algorithms have been proposed. Molenbruch et al. [2017] and

Hyland and Mahmassani [2017] provide classifications and taxonomy of these algorithms and their underlying problem settings. The most common classification, which is followed here, is based on the assumed dynamism and stochasticity of the problem.

The static and deterministic DARP is the simplest variant of the problem. It assumes that all information (i.e., customer requests) is known in advance and no randomness is involved. Cordeau and Laporte [2003] formulated this problem as a mixed integer linear program and proposed a tabu search heuristic to solve the problem. They later proposed a branch-and-cut algorithm to solve the problem optimally [Cordeau, 2006]. Nevertheless, due to the NP-hardness of the problem, the size was limited to four vehicles, with up to 30 customers to be transported in the latter. Following studies focused either on (meta-)heuristics to small scale benchmark instances as close to optimality as possible (e.g., [Parragh et al., 2010; Parragh and Schmid, 2013; Massobrio et al., 2016]), or develop heuristics to solve large-scale instances (e.g., [Muelas et al., 2013; Muelas et al., 2015]). As the use case of the static variant is to plan vehicle routes in advance, computational time is not a strongly limiting factor. For example, the variable neighborhood search algorithm by Muelas et al. [2015] could solve instances of up to 16k requests a day. While an initial solution could be found within a few minutes, the search algorithm did not converge in a (local) optimum within 3 hours for these instances.

In the dynamic and deterministic DARP, customer requests are not known in advance but are revealed over time. Therefore, solutions to the DARP have to be updated over time. A common approach is to solve the problem in a rolling horizon fashion, i.e., the static variant of the problem is solved for currently revealed information and the solution is updated. The computational time is, therefore, limited to the update period. A fast heuristic for this problem was proposed by Jaw et al. [1986]: Once new requests are revealed, they are inserted into the current solution, resulting in a fast (but not necessarily optimal) solution to the DARP. This approach is often referred to as "Insertion Heuristic". Other approaches utilized the developed meta-heuristics for the static DARP and terminate the search after a specific time limit (e.g. Parragh and Schmid [2013], Massobrio et al. [2016], and Jain and van Hentenryck [2011]). Nevertheless, the scale remained relatively small as, for example, an instance with a maximum of 45 passengers was solved by Massobrio et al. [2016].

In the dynamic and stochastic DARP, customer requests are revealed over time. Additionally, stochastic information about the future is available. This can include information about the probability of future requests, the probability of a customer accepting a trip offer, or stochastic travel times. Hyytiä et al. [2012] proposed a method based on queuing theory to include information about future requests in the assignment process. Bent and van Hentenryck [2004] proposed a stochastic programming method by including stochastic requests drawn from a probability distribution in the assignment algorithm and solved it on benchmark scenarios with 100 requests and up to 17 vehicles. van Engelen et al. [2018] adopted the insertion heuristic by Jaw et al. [1986] for demand-anticipatory assignments. Due to the efficient insertion heuristic, the authors could solve instances with up to 2000 requests and 100 vehicles in a few seconds.

For a more detailed overview of the DARP and its solution algorithms, the reader is referred to Molenbruch et al. [2017]. Most of the studies presented so far focused on solving rather small-scale instances of the DARP. The fleet control problem of the ARP can be interpreted

as a dynamic and stochastic DARP, but in contrast to the studies presented here, the scale of the problem is much larger. As these large-scale problems are the core of this thesis, the following section focuses on studies that have been proposed to solve these large-scale assignment problems for ARP.

2.4.2 Assignment

The development of large-scale assignment algorithms for ARP services is a rather recent research field and evolved in several directions. Firstly, the already mentioned DARP studies focused on small to mid-scale instances of the problem, with an emphasis on achieving optimality. Secondly, large-scale assignment algorithms for ride-hailing services, where trips are not shared, received significant attention from the scientific community. This progress was driven by the recent rise of ride-hailing services like Uber and Lyft. Finally, large-scale assignment algorithms for ridesharing services have been developed, driven by the recent interest in shared mobility and the need for applications with real-time matching of drivers and riders.

Assignment for Ride-Hailing

The computational advantage of the ride-hailing assignment problem is that the number of permutations to create vehicle schedules is much smaller compared to a ride-pooling service. This led to the application of simple rules to assign vehicles to customers, like First-Come-First-Served (FCFS) assignment strategies to assign incoming requests to the nearest idle vehicle [Zhang et al., 2015] or identifying the nearest idle vehicles in a given subregion of the network [Fagnant and Kockelman, 2014]. Bischoff and Maciejewski [2016] changed the assignment strategy based on the current utilization of the fleet: If there is an oversupply of idle vehicles, incoming requests are assigned to the nearest vehicles, while in case of an undersupply, vehicles are assigned to the nearest unserved customers. Hyland and Mahmassani [2018] compared multiple heuristic strategies to assign customers to vehicles immediately. If multiple requests are assigned at once in a batch, the assignment for ride-hailing can be formulated as a bipartite matching problem, which can be solved efficiently [Ruch et al., 2018]. The efficiency of assigning requests in batches was shown by Hörl et al. [2019] for a case study of Zurich, Switzerland. Syed et al. [2019] further introduced a neural network based meta-heuristic to solve the assignment problem in batches. Erdmann et al. [2021] proposed a mixed framework that allows assigning requests immediately while assignments are re-optimized in batches. Tobias Enders et al. [2023] developed a deep reinforcement learning algorithm to make informed acceptance and rejection decisions for incoming requests.

Matching for Coordinated Ride-Sharing

The matching problem for coordinated ridesharing services is similar to the ride-pooling problem, as trips are shared between multiple customers. Nevertheless, the difference is that drivers of the ridesharing services are private drivers with a specific planned trip they are willing to share. Therefore, driver detours and trip delays also have to be considered in the matching process. Agatz et al. [2011] proposed a greedy heuristic to match driver-rider pairs for a large-scale case study of Atlanta, US. Other approaches included kinetic tree search Huang

et al. [2013] and non-myopic graph search heuristics [Guo et al., 2021]. Stiglic et al. [2016] showed the importance of driver and rider flexibility (i.e., the willingness to shift trips) for the success of a match. Huang et al. [2022] additionally proposed a congestion-aware routing model for the ridesharing service. For a more detailed overview of the matching problem for ridesharing services, the reader is referred to Agatz et al. [2012] and Furuhata et al. [2013].

Assignment for Ride-Pooling

Computational time is the main limiting factor for the assignment problem of ARP services. A potential huge solution space has to be searched efficiently to find a good solution within a short time frame to allow fast reaction times to new requests. Therefore, early studies focused on the development of efficient search heuristics. Ma et al. [2013] were among the first to develop a large-scale solution algorithm for a ride-pooling service. Multiple heuristics are applied to deal with the complexity of the problem. The insertion heuristic is used to create the schedules, while a spatial local search is applied to decide on a candidate vehicle in the vicinity of a request. Additionally, a grid-based travel time matrix is used to decrease the need for computationally expensive travel time calculations.

Because of its efficiency, the insertion heuristic has been used in many studies that evaluated the impact of city-wide ARP services. Fagnant and Kockelman [2018] sequentially checked the nearest vehicles for a new request and assigned it to the first vehicle that could fulfill a set of constraints (e.g., waiting and detour constraints). Other studies (e.g., Bischoff et al. [2017], Vosooghi et al. [2019], Fiedler et al. [2018], and Dandl [2022]) inserted new requests into a preselected set of candidate vehicles and assigned the schedule with the best objective value. This preselection is usually based on all vehicles that can reach the request within a specific maximum waiting time constraint [Bischoff et al., 2017; Vosooghi et al., 2019; Fiedler et al., 2018] or further heuristics that decrease the search space [Dandl, 2022].

Even though the insertion heuristic is computationally efficient and, therefore, practical for many applications, it is not guaranteed to find the optimal solution. On the one hand, new requests are treated sequentially, resulting in suboptimal solutions compared to requests treated in batches. On the other hand, the insertion heuristic does not allow for adapting previously made decisions based on new information as assignments remain fixed. It might be beneficial to update the stop order of already assigned schedules to accommodate new requests or even shift the assignment of customers who have not been picked up yet to another vehicle. This process is usually referred to as "re-assignment".

To allow re-assignments and improve the initial solution often created by an insertion heuristic, a series of other studies focused on the development of meta-heuristics. Santos and Xavier [2013] proposed an adaptive search procedure which they improved later on [Santos and Xavier, 2015]. Jung et al. [2016] applied a hybrid-simulated annealing meta-heuristic, while Zhan et al. [2021] developed a modified bee colony algorithm.

One of the most influential algorithms for large-scale assignment problems for ARP services has been proposed by Alonso-Mora et al. [2017a]. This algorithm elaborates on the concept of shareability graphs introduced by Santi et al. [2014] and extends it by introducing request-vehicles graphs to not only find possible request-request but also request-vehicle matches. If pick-up and drop-off time constraints (e.g., maximum waiting time or maximum trip detour

time) are tight enough, an efficient search algorithm can be formulated that allows finding a large subset (under some circumstances, even all) feasible vehicle schedules. Even if the algorithm would not terminate in time for real-time operation, time-outs can be used to return solutions anytime. As re-assignments are possible, the authors argue that the solution can be improved over time if a suboptimal solution is returned in a previous optimization epoch due to time-outs. The possibility for large-scale application is demonstrated in a case study for Manhattan, NYC, with up to 2,000 simultaneously active requests and vehicles with a capacity of up to ten passengers.

Multiple studies elaborated on this approach to improve computational efficiency. Liu and Samaranayake [2022] developed speed-up techniques for the algorithm by Alonso-Mora et al. [2017a]. Firstly, only schedules for the closest vehicle to a request are created, while it is checked later whether other vehicles can fulfill this schedule in time, too. Secondly, an efficient distribution of computational load to different processors is proposed. They report a speed-up of up to 98% compared to the original algorithm. Engelhardt et al. [2020] and Li et al. [2021] suggested further improvements by keeping already computed schedules and graphs in memory. As the time between consecutive optimization epochs is short (usually in the range of seconds to a minute), the authors argue that most of the previously computed schedules remain feasible. Engelhardt et al. [2020] additionally proposed vehicle search heuristics that decrease the connectivity in the shareability graph and thereby decrease the computational load.

Riley et al. [2019] proposed a column generation approach to iteratively improve the current assignment until the best solution is found. Within a rolling horizon, the goal is to minimize waiting time, while a maximum trip time constraint guarantees short travel times. No maximum waiting time constraint is considered. Instead, customers queue up until a feasible solution is found. They report that instances of up to 30,000 requests per hour could be solved for a Manhattan case study. Simonetto et al. [2019] further proposed a method to deal with large-scale ride-pooling assignment problems. If no re-assignments are allowed and each new request is assigned to a different vehicle, the schedule to vehicle assignment can be cast into a linear assignment problem, providing computational efficiency. By comparing the algorithm with the one proposed by Alonso-Mora et al. [2017a], the authors found that a speed-up of 4x could be achieved while they argue that the loss in solution quality is minor. Wang et al. [2023] allows re-assignment by solving multiple linear assignment problems iteratively until the solution converges.

Notable further extensions of the assignment problem for ARP services include the dynamic selection of boarding locations [Engelhardt and Bogenberger, 2021; Fielbaum et al., 2021; Zuo et al., 2021], the integration of transfers between vehicles [Masoud and Jayakrishnan, 2017; Namdarpour et al., 2024], or spatially dependent rejection penalties to foster service fairness in the operating area [Schuller et al., 2021].

Study	Objective	Time Constraints	Re-Assignment	Demand	Fleet Size (Vehicle Capacity)	Method
Ma et al. [2013]	1. Served Requests 2. VKT	Pick-up TW (5min)	✗	Up to 60k/hour	3k(?)	Multiple Search Heuristics
Santos and Xavier [2013] Santos and Xavier [2015]	1. Served Requests 2. Traveler Fares	LAT (30min) MTF (private ride)	(✓)	78k/day	1333(4)	Greedy Randomized Adaptive Search Procedure
Hosni et al. [2014]	Profit	MDT (20min)	✗	200 per instance	50(4)	Lagrangian Decomposition
Jung et al. [2016]	1. Served Requests 2. Ride Time & Wait Time Compare to Profit	MWT (15min) MDT (2.0 rel)	✓	Up to 18k/4h	600(4)	Hybrid Simulated Annealing
Alonso-Mora et al. [2017a]	1. Served Request 2. Travel Delay	MWT (7min) MDT (twice MWT)	✓	Up to 460k/day	3k(10)	Graph-based Trip Search + Assignment
Riley et al. [2019]	1. Served Request 2. Wait Time	MDT(max(1.5 rel, 2min max))	✓	Up to 33k/hour	2k(5)	Column Generation
Simonetto et al. [2019]	1. Served Requests 2. System Time	MWT (7 min) MDT (7 min)	✗	Up to 460k/day	3k(10)	Linear Assignment
Engelhardt et al. [2020]	1. Served Requests 2. VKT	MWT(8 min) MDT (1.4 rel)	✓	180k/day	3k (4)	Trip Search + Assignment vs Insertion Heuristic
Hyland and Mahmassani [2020]	Served Requests + Delay + VKT	MDT (Up to 1.8 rel)	✗	18k/7h	Up to 1,000(2)	Bi-partite Matching
Li et al. [2021]	1. Served Requests 2. Travel Delay	MWT(5 min) MTD (twice MWT)	✓	Up to 800k	3200(10)	Graph-based Trip Search + Assignment
Zhan et al. [2021]	Served Requests + Travel Cost Ratio + Travel Time Ratio	MWT (5min) MDT (1.3 rel)	✓	3,661/h	2400(4)	Modified Bee Colony Algorithm
Liu and Samaranayake [2022]	1. Served Requests 2. Travel Delay	MWT(5min) MTD (twice MWT)	✓	116k/6h	3k(10)	Graph-based Trip Search + Assignment
Fiedler et al. [2022]	1. Served Requests 2. Travel Delay	MTD (3 to 7 min)	✓/ ✗	Up to 120k/h	Up to 16k (5)	Trip Search + Assignment vs Insertion Heuristic
Wang et al. [2023]	1. Served Requests 2. System Time	MWT(10min) MDT(1.5 rel)	✓	316/20min	394(4)	Iterative Graph-based Matching
Ruijter et al. [2023]	Net Sharing Benefit	Sharing Benefit Improvement	✓	1210/h	150(3)	Graph-based Trip Search + Assignment

Table 2.3: Collection of studies presenting solutions to the ride-pooling assignment problem. Abbreviations: MWT: Maximum Wait Time, MDT: Maximum Delay Time, TW: Time Window, LAT: Latest Arrival Time, MTF: Maximum Travel Fare, VKT: Vehicle Kilometers Traveled.

Refined Contribution Statement

Table 2.3 provides an overview of the studies discussed in this section. Most studies utilize hierarchical objectives with the primary goal of maximizing the number of served requests, while the secondary goal includes minimizing traveler delay, VKT or costs. The primary objective is needed for those algorithms that employ hard time constraints, especially on customer waiting times, as generally a service for all customers cannot be guaranteed. Most studies apply maximum waiting time constraints for five to ten minutes, with a maximum travel delay of a similar order of magnitude, either in absolute measures or relative to the customer direct trip travel time. The trend is that these time constraints are tighter for large-scale instances. From a computational standpoint, this reduces the solution space and, therefore, computational time. From the service standpoint, it is assumed that the larger fleet size can provide this customer service level.

This thesis builds upon the work of Engelhardt et al. [2020] and contributes with the following aspects to the literature:

Contributions

- Improving the computational efficiency of the algorithm by Alonso-Mora et al. [2017a], especially by keeping computed schedules in memory. (Section 3.2.4)
- Development and Evaluation of re-assignment strategies from operator and customer perspective. (Section 3.2.5)
- A benchmark comparison with the variants proposed by Alonso-Mora et al. [2017a], Simonetto et al. [2019] and the Insertion Heuristic by Jaw et al. [1986]. (Section 3.2.6)
- A detailed evaluation of three different case studies for Chicago, Munich, and Manhattan. (Section 4.2)

2.4.3 Repositioning

The repositioning (or rebalancing) problem arises predominantly in systems characterized by high dynamism and stochasticity. Just reacting myopically to incoming demand will lead to an imbalanced system when the spatio-temporal demand patterns are not symmetric. These imbalances can lead to long user waiting times or even unfulfilled service requests if supply is locally not available when requested. These features are particularly common in MoD services but also have applications in other domains like disaster response [Gao, 2022] or repositioning of ambulances [Brotcorne et al., 2003].

Within MoD services, the repositioning problem originates from vehicle sharing services, like carsharing services (e.g. Weikl and Bogenberger [2013] and Illgen and Höck [2019]), bikesharing services (e.g. Dell'Amico et al. [2014] and Reiss and Bogenberger [2017]) or, after their advent, scootersharing services (e.g. Osorio et al. [2021] and Lee et al. [2024]). In these services, vehicles have to be repositioned to ensure they are available where needed. As staff is needed to relocate the vehicles manually, repositioning is costly and has to be scheduled carefully, limiting the frequency of repositioning to a maximum of a few times a day. In ridesourcing services, drivers are available for each vehicle who can frequently reposition their vehicle when they are not serving a customer. Nevertheless, as drivers are paid by the number of served customers, they tend to reposition their vehicles greedily to maximize their own revenue resulting in the high fraction of empty VKT discussed earlier [Castillo et al., 2017].

When services become autonomous[4], the constraints for repositioning problem changes: In contrast to Car- and Bike-Sharing services, the cost of repositioning decreases drastically as no service provider employee has to be transported to the vehicles and/or move the vehicles manually. This allows for the repositioning of idle vehicles on a much higher frequency. In contrast to TNC services, where drivers tend to rebalance themselves greedily to maximize their revenue, the operator of an AMoD service can centrally plan the repositioning of vehicles to optimize a global objective.

Ride-Hailing Algorithms

For the ride-hailing use case, which does not allow for shared trips, a common approach is to aggregate the expected future demand into zones. Since each anticipated future trip requires exactly one vehicle as supply to be available, analytical formulations for zonal demand-supply imbalances are derived that can be used to solve a matching problem to rebalance idle vehicles. For example, Zhang and Pavone [2016] used a queuing theoretical approach to formulate imbalances of the AMoD service as a Jackson Network and solve the repositioning problem to stabilize it. This approach has been further extended by Iglesias et al. [2018] and Tavor and Raviv [2023]. Valadkhani and Ramezani [2023] proposed a macroscopic model to predict future fleet states and rebalance vehicles accordingly to optimize profit.

The aforementioned models rely on the spatial and temporal aggregation of anticipated future demand. Therefore, Dandl et al. [2019] evaluated the impact of spatio-temporal demand forecast aggregation and found that less aggregated demand profits the ride-hailing

[4] Or at least vehicles are operated by drivers that receive centrally planned routes and are paid by working hours and not by the number of served customers.

service. However, it is crucial to find an appropriate balance. Zones that are too small may cause the approximated spatial coverage of vehicles to extend beyond the zone boundaries. To reduce the impact of the spatial aggregation method for repositioning, Syed et al. [2021], therefore, introduced spatial correlations based on Gaussian Kernels between zones. Zhu et al. [2022] approximates the spatial supply density by dynamically adjusting Voronoi cells originating from each vehicle. Ackermann and Rieck [2023], too, developed a method to reposition vehicles based on dynamically created zones that are partially overlapping. Besides interzonal rebalancing, Yeo et al. [2023] also integrated intrazonal repositioning into their model. Hörl et al. [2019] evaluated the impact of the rebalancing algorithm in a Multi-Agent Transport Simulation (MATSim) simulation. They compared a service without rebalancing to a service with repositioning using the algorithm by Zhang and Pavone [2016] with a perfect forecast and a forecast extrapolating the current demand (myopic forecast). They found that the choice of the repositioning algorithm has more impact on the overall system performance than the choice of the assignment algorithm. Additionally, applying myopic forecast drastically increases empty VKT and customer waiting times. Brar and Su [2021] proposed a learning-based method to dynamically adjust hyperparameters like repositioning frequency and temporal forecast horizon to optimize the service performance. Their evaluation showed a trade-off between served requests and VKT, especially in repositioning frequency: The higher the frequency, the more requests can be served, but the higher the VKT due to a higher number of unnecessary repositioning trips. Finally, Guo et al. [2022] used a Long Short-Term Memory (LSTM) based learning approach to directly learn valuable repositioning actions from historical data.

Most of these studies focused on maximizing served requests or minimizing customer waiting times while minimizing repositioning costs. Winter et al. [2021] and Schuller et al. [2021] also considered fairness aspects (e.g., equal service availability in the whole operating area) in their repositioning algorithm. Additionally, Dean et al. [2022] considered the combination of charging and repositioning, while Winter et al. [2021] also considered parking constraints in their approach.

Ride-Pooling Algorithms

The repositioning problem is more complex for the ride-pooling use case, as multiple customers can share a trip. As a result, the relationship between expected demand and the necessary supply becomes intricate. Idle vehicles can serve multiple future requests. Additionally, en-route vehicles can accommodate future requests, too. Some studies have suggested methods to address this challenge: Wallar et al. [2018] as well as Bischoff and Maciejewski [2020] introduced a linear scaling factor of predicted demand to convert expected demand to supply, allowing the use of a computationally efficient macroscopic model. Alternatively, Schlenther et al. [2023] extended the approach by Bischoff and Maciejewski [2020] and proposed aligning relative demand and supply distributions instead of rebalancing vehicles to absolute measures of demand. Ma and Koutsopoulos [2022] formulated the problem by repositioning vehicles to zones that maximize the likelihood of finding at least one matching request within the forecast horizon but did not include sharing of trips explicitly in their formulation. In contrast, Wen et al. [2017] introduced load factors for currently en-route vehicles based on their current occupancy for non-idle vehicles to contribute to fractional

zonal supply. Alonso-Mora et al. [2017a], with additional refinements proposed by Liu and Samaranayake [2022], suggested a purely reactive approach to rebalance idle vehicles to locations with unserved demand. In a follow-up paper, Alonso-Mora et al. [2017b] additionally developed a large-scale predictive repositioning method: Samples from future requests are directly included in the assignment algorithm. While this method showed promise in large-scale simulations for Manhattan, the inclusion of future request samples drastically increased computational time, necessitating the addition of multiple time-outs in the assignment process to manage computational demands effectively. Lowalekar et al. [2018], too, developed a sampling-based demand anticipatory matching algorithm based on a multi-stage stochastic optimization model, which they solved by applying Bender's decomposition. Nevertheless, only request trips starting and ending in the same zone can be shared in their model. Tuncel et al. [2023] proposed an integrated matching and rebalancing problem. Within the predictive rebalancing problem, sharing of trips is considered weighting available vehicle seats within their supply estimation.

While these approaches are computationally efficient and allow application in large-scale instances, other studies have proposed more sophisticated methods. Sayarshad and Chow [2017] formulated a rebalancing problem based on Markov Decision Processes, but the problem size is restricted to six zones in their case study. Li et al. [2019a] proposed a solution method for the stochastic DARP using sampling of predicted future requests, but the problem size was restricted to four vehicles. Tsao et al. [2019] proposed a Model Predictive Control (MPC) approach to steer vehicles towards future expected demand, but this method is limited to a maximum of two requests sharing a trip. Adopting the insertion heuristic, van Engelen et al. [2018] proposed a demand-anticipatory assignment algorithm that steers en-route vehicles towards expected demand. While the algorithm can reduce customer waiting time, they evaluated that a simple idle vehicle rebalancing strategy outperforms the demand-anticipatory assignment in terms of customer rejections. Tafreshian et al. [2021] formulated the ride-pooling control problem as a time-expanded network. They suggested a two-stage model to anticipate future demand: In the first (offline) stage, a set of candidate vehicle routes is generated to serve future demand. In the second (online) stage, on-demand requests are assigned to these routes.

As analytical formulations are hard to find, multiple studies proposed deep learning approaches that show promising results. Wen et al. [2017] and Chouaki et al. [2022] applied a Q-learning approach that shows similar performance to their optimization-based approach but with a much lower computational burden. Gueriau et al. [2020] showed that deep learning can also be applied to incorporate stochastic travel times in the repositioning problem. Cheng Li et al. [2022] proposed a value-based learning approach with an offline policy evaluation and an online update procedure and showed that their approach outperforms a myopic strategy.

Refined Contribution Statement

Table 2.4 provides an overview of the studies developing repositioning strategies for ride-pooling services that have been discussed in this section.

Checkmarks in the table indicate whether pooling is explicitly considered in the formulation. A pure checkmark refers to studies that explicitly consider pooling in their supply estimation.

This is especially true for sampling-based methods that create possible future vehicle routes from sampled future requests. Brackets indicate that a ride-hailing-based algorithm has been adapted slightly to the ride-pooling use case. These methods usually rely on the calibration of parameters to account for the pooling service. Due to the computational burden, it can be observed that only rather small-scale studies fully consider pooling in their formulation. Forecast of demand is usually based on a simple method of averaging historical demand. Some studies not only forecast demand on a trip-origin level but also the destination of trips, which is especially relevant for algorithms that explicitly consider ride-pooling in their formulation.

Three further features are considered in the table: Spatial and temporal aggregation of demand, and whether repositioning trips have to be completed once assigned (the trip is "locked"), or whether a repositioning trip can be aborted to serve incoming demand. Spatially, demand is usually aggregated on a zonal or station-based level, while temporally, demand is aggregated within a single horizon. In contrast to single-horizon approaches, multi-horizon approaches, like the MPC approach by Tsao et al. [2019], allow the incorporation of multiple repositioning epochs leading to a better estimation of the available supply. The locking of repositioning trips varies among the studies, giving no clear indication about the best strategy. The study by Tafreshian et al. [2021] is marked in brackets, as their repositioning approach by following candidate routes deviates from other approaches of assigning single repositioning trips.

With that in mind, this thesis builds upon the work of Engelhardt et al. [2023] and contributes with the following aspects to the literature:

Contributions

- Development of an efficient multi-horizon repositioning algorithm that explicitly considers ride-pooling in the supply estimation. (Section 3.3)
- Evaluation of forecast accuracy and aggregation on the repositioning performance. (Section 5.3.3)
- Assessment of allowing re-assignments of repositioning trips. (Section 5.3.4)
- Benchmark to other state-of-the-art repositioning algorithms. (Section 3.3.5)
- Evaluation in the three large-scale case studies Chicago, Munich, and Manhattan. (Section 5.3)

Study	Spatial Aggregation	Temporal Aggregation	Supply Estimation	Lock Repo.	Demand Forecast	Explicit Pooling Formulation	Tested System Size
Lowalekar et al. [2018]	Zonal	Single Horizon	Schedules by Request Sampling (Only requests with same OD relation can share trips)	✓	Historic Data-based (O+D)	(✓)	8000 Vehicles 53k Rq. per 2.5h
Alonso-Mora et al. [2017a] Liu and Samaranayake [2022]	None	None	Autocorrelation with Unserved Demand	✗	None	✗	3000 Vehicles 460k Rq. per Day
Wallar et al. [2018]	Zonal	Single Horizon	Linear Factor of Demand Estimation	✗	Adaptive real-time estimation (O)	(✓)	3000 Vehicles 460k Rq. per Day
Alonso-Mora et al. [2017b]	Zonal	Single Horizon	Schedules by Request Sampling	✗	Historic Data-based (O+D)	✓	3000 Vehicles 460k Rq. per Day
Ma and Koutsopoulos [2022]	Zonal	Single Horizon	Equal Prob. of Unmatched Demand	✗	Historic Data-based (O)	(✗)	3000 Vehicles 125k Rq. per Day
Schlenther et al. [2023]	Zonal	Single Horizon	Equal Vehicle-To-Population-Ratio	✓	Simulation-based (O)	(✓)	2000 Vehicles 27.5k Rq. per Day
Tuncel et al. [2023]	Zonal	Single Horizon	Load Factors based on Vehicle Occupancy	✗	Historic Data-based (O)	(✓)	1500k Vehicles 120k Rq. per 16h
Tsao et al. [2019]	Station-based	Multi Horizon	MPC with Max. 2 Customers Sharing a Trip	✓	Historic Data-based (O+D)	(✓)	400 Vehicles 465k Rq. per Month
Sayarshad and Chow [2017]	Station-based	Single Horizon	Queue Length Estimation	✓	Historic Data-based (O)	✓	150 vehicles 8640 Rq. per Day
Bischoff and Maciejewski [2020]	Zonal	Single Horizon	Linear Factor of Demand Estimation	✓	Simulation-based (O)	(✓)	100 Vehicles 11.1k Rq. per Day
Tafreshian et al. [2021]	Station-based	Multi Horizon	Preprocess Daily Routes Based on Network Flow Problem	(✓)	Historic Data-based (O+D)	✓	100 Vehicles 250k Rq. Per Day
Wen et al. [2017]	Zonal	Single Horizon	Load Factors based on Vehicle Occupancy	N/A	Perfect (O)	(✓)	20 Vehicles 100 Rqs. per Hour
Li et al. [2019b]	None	Single Horizon	Schedules by Request Sampling	✗	Historic Data-based (O+D)	✓	4 Vehicles 164 Rq. per Day

Table 2.4: Collection of studies proposing repositioning strategies of ride-pooling services sorted by reported system size of the case study. Abbreviations: (O): Only Origin; (O+D): Origin and Destination

2.4.4 Reservation

The incorporation of reservations into the ARP service promises benefits on the operational and customer side. On the operational side, reservations could allow for better fleet planning, leading to a more efficient service. On the customer side, pre-booking rides can increase service reliability, which might be a convenient option for trip purposes that are time-critical and involve a fixed appointment.

Full Reservation-Based Services

From the operational side, Santi et al. [2014] evaluated the matching efficiency on the shareability graph. When all requests are known ahead, saved travel time can be improved by up to 8% compared to when revealed online. Bilali et al. [2019a] developed an analytical model to compare the shareability of trips when all requests are known a short time ahead compared to purely on-demand requests. They evaluated that a reservation time of only 2 minutes can increase the chances of finding sharable trips by up to 30%. Ouyang et al. [2021] developed an analytical model to calculate the optimal reservation time for a many-to-many ride-sharing service.

The advantage of pre-booking rides originates from the absence of stochasticity in the assignment problem[5]. If all requests are known ahead, the assignment problem can be solved once by solving the static DARP. Agatz et al. [2011] proposed a rolling horizon approach to solve the many-to-many ride-sharing problem for different reservation horizons. Within the rolling horizon approach, the static DARP is solved only for customers requesting a ride within a specific time horizon to reduce the overall problem size. Lu et al. [2023] tested the impact of different reservation horizons for a purely reservation-based ride-pooling service. Using a heuristic VRP solver, they found a maximum in service efficiency for a reservation horizon of around 30 minutes. Yang et al. [2022] evaluated a purely reservation-based ride-pooling service, where customers continuously request rides for at least two hours in advance. They proposed a network flow optimization approach with a rolling horizon and found a steady increase in system performance with increased horizon length.

Other studies focused on finding good solutions for the large-scale static DARP directly using heuristics and meta-heuristics. Su et al. [2022] proposed a clustered tabu search algorithm to solve instances with up to 4,000 requests within three hours based on the NYC taxi data set. Solutions could be found within 300s, but this study did not consider ride-pooling. Muelas et al. [2015] developed a variable neighborhood search algorithm to solve the problem with ride-pooling. Solutions to the problem of up to 16k requests and 1,700 vehicles for case studies of San Francisco could be found within an execution time of three hours. Wallar et al. [2021] formulated a heuristic based on a batch scheduling algorithm initially intended to estimate fleet sizes for an on-demand ride-pooling service. Nevertheless, the approach solves a DARP with shared rides for the NCY taxi data of up to 450k requests a day within a computational time of up to 7.6 hours. While previous studies mainly focused on minimizing operational costs, Kucharski and Cats [2020] developed a model to solve the static ride-pooling problem by assigning customers to schedules based on their utility of a shared ride. Only routes that

[5] Ignoring other sources of stochasticity like network travel times.

improve customer utility are assigned compared to a private ride, which they refer to as an attractive ride.

Mixed - On-Demand and Pre-Booking - AMoD Services

When mixed services are considered, where trips can be requested both on-demand and pre-booked, the scheduling problem becomes more complex as the assignment algorithm has to consider requirements for both types of requests. This complexity arises from the unknown state of the fleet at the time of the pre-booking request, which is mainly influenced by on-demand requests that are revealed over time. When pre-booking is considered in an AMoD service, the main question is: "How can a service be guaranteed for a pre-booked trip once it is confirmed?" According to Yu et al. [2023], real-world operators rely on simplified methods yet: They report that Lyft, for example, treats pre-booked requests similar to on-demand requests when the pre-booked time approaches, while Didi treats pre-booked rides separately with increased spatial search radius for drivers.

Different studies proposed more sophisticated methods to evaluate the impact of reservations on service performance. The first block of studies focused on integrating short- to mid-term reservations in the order of 30 minutes to one hour pre-booking times [Wen et al., 2019; Ma and Koutsopoulos, 2022; Dandl, 2022]. In this case, long-term fleet planning is not necessary. Therefore, pre-booked requests can be treated similarly to on-demand requests and directly assigned to vehicle schedules, ensuring available capacity after booking confirmation. To cope with the increased solution space when incorporating pre-booked requests, all three studies rely on insertion heuristics to assign pre-booked rides. While Wen et al. [2019] and Ma and Koutsopoulos [2022] also applied insertion heuristics for on-demand requests, Dandl [2022] considered a rolling horizon-based re-optimization approach for on-demand requests and pre-booked requests that fall within the time horizon of the current optimization epoch. Concerning the impact of reservation requests, Wen et al. [2019] evaluated an increase in service performance when 5% of customers pre-booked their rides but a decrease in service performance when 10% of the rides have been pre-booked. They argued that the reduction in service performance results from the additional commitment to serve pre-booked rides, resulting in additional rejections for on-demand customers. A similar observation is made by Dandl [2022]. In this study, pre-booking consistently reduced service performance. The service performance could be improved only if all requests reserved their rides. On the contrary, Ma and Koutsopoulos [2022] found a steady performance increase with increasing pre-booking fractions and pre-booking time.

The second block of studies focused on integrating long-term reservations, where customers can pre-book rides for more than two hours in advance. Yu et al. [2023] developed a rolling horizon approach to solve the mixed service problem with long-term reservations. Nevertheless, the confirmation of a pre-booked ride is only guaranteed when the pick-up time falls within the rolling horizon. Duan et al. [2020] developed a two-stage model to confirm long-term reservations upon request for a ride-hailing service. They split vehicle schedules into short-term and long-term assignments, where long-term assignments treated pre-booked rides. The short-term assignment included on-demand and upcoming pre-booked rides while maintaining the feasibility of long-term assignments. For a ride-pooling service, the studies by Cui et al. [2023] and Duan et al. [2023] developed similar approaches to the one presented in this

thesis, which is based on Engelhardt et al. [2022a]: Similar to Duan et al. [2020], short and long-term assignments are considered to guarantee service for long-term reservations. Yu et al. [2023] developed an ant colony meta-heuristic to solve the mixed service problem on a small network of Delft with up to 1,112 requests in 15h. Duan et al. [2023] deploys several heuristics to cope with the large-scale problem, which they test for Manhattan with up to 365k requests per day. The long-term assignment is based on the insertion heuristic to create a long-term solution for pre-booked rides, while the short-term assignment includes a re-assignment procedure to improve the long-term solution within a rolling horizon when on-demand customers are revealed.

For long-term reservations, too, the results of the studies are diverging: Duan et al. [2020] observed an increase in operator profit until a pre-booking fraction of around 80% followed by a slight decrease. Nevertheless, the service always profits from pre-booked rides when trips are not shared. Similarly, Cui et al. [2023] found that the operator profit increases with increasing pre-booking fractions and book-ahead time. Duan et al. [2023] found that the operational efficiency maintains a relatively constant level with varying pre-booking fractions and average pre-booking time.

Finally, the study by Abkarian et al. [2022b] should also be mentioned here. Contrary to the other studies, they proposed a service with two options for customers: They could reserve a vehicle for a specific period of private use or book a ride on demand. A simulation study for taxi trips in Chicago showed that a mixed-service operation can provide a better service than two separate operations.

Study	Ride-Pooling	Mixed Service	Long-Term Reservations	Reservation Acceptance	Method	Main Result Efficiency is:	Case Study
Su et al. [2022]	✗	✗	✓	After solving routing problem	Solve Offline Problem for Reservation Requests	N/A (Algorithm Comparison)	Manhattan 3968 Rqs/3h
Lu et al. [2023]	✓	✗	✗	At Request Time	Heuristic VRP Solver	▪ optimal at 30 min *RH*	Town in Poland 1637 Rqs/day, 20 vehs
Yang et al. [2022]	✓	✗	✓	At Short Term Horizon (2h)	Network Flow Model Within Rolling Horizon	▪ increasing with *RH*	Delft 1112 Rqs/15h
Wen et al. [2019]	✓	✓	✗	At Request Time	Insertion Heuristic for both request types	▪ increased at 5% *RR* ▪ decreased at 10% *RR*	Major European City 700 Rqs/h
Ma and Koutsopoulos [2022]	✓	✓	✗	At Request Time	Insertion Heuristic for both request types	▪ increasing with *RR* ▪ increasing with *RH*	Chengdu 125k Rqs, 3k vehs
Dandl [2022]	✓	✓	✗	At Request Time	Insertion for Long-Term, Re-Opt in Short Term	▪ decreased with *RR* ▪ improved only at 100% *RR*	Munich
Duan et al. [2020]	✗	✓	✓	At Request Time	Short term and long term planning with offline solution	▪ increasing with *RR* ▪ decreasing from 80% *RR* ▪ decreasing with *RH*	Manhattan 700 vehs
Yu et al. [2023]	✗	✓	✓	At Short Term Horizon (30 min)	Network Flow Model Within Rolling Horizon	▪ increasing with *RR*	Manhattan 200 vehs, 250 Rqs/h
Cui et al. [2023]	✓	✓	✓	At Request Time	Short term and long term planning with offline solution	▪ increasing with *RR* ▪ increasing with *RH*	Delft 1112 Rqs/15h
Duan et al. [2023]	✓	✓	✓	Immediate	Short term and long term planning with offline solution	▪ rather constant with *RR* ▪ rather constant with *RH*	Manhattan 364k Rqs/day

Table 2.5: Collection of studies proposing reservation strategies of ride-pooling services. Abbreviations: (*RR*): Reservation Rate; (*RH*): Reservation Horizon

Refined Contribution Statement

Table 2.5 provides an overview of the studies that have been discussed in this section and deal with the mixed service for AMoD services with on-demand and pre-booking requests. This thesis deals with the mixed service problem for ride-pooling services, focusing on long-term reservations and service guarantees for pre-booked rides. As the table shows, only the studies by Cui et al. [2023] and Duan et al. [2023] dealt with this problem. Both studies have been published slightly after the publication referring to this chapter in this thesis [Engelhardt et al., 2022a] and show some similar approaches to this work. Similarly to Cui et al. [2023] and Duan et al. [2023], a two-stage approach is developed that first creates long-term vehicle schedules for pre-booked requests. In the second stage, the schedules are re-optimized within two rolling horizons to schedule on-demand requests together with upcoming reservation requests while maintaining the feasibility of long-term vehicle schedules. In contrast to the method by Cui et al. [2023], the method in this thesis allows the evaluation of large-scale instances. This is also the case for the approach proposed by Duan et al. [2023]. Nevertheless, a disadvantage of the method is that the algorithms are based on many heuristics, and the treatment of assignment, rebalancing, and reservation tasks is strongly interrelated. The approach in this thesis is highly modular and allows for a more detailed analysis of the impact of different components on the overall service performance. This is especially relevant considering the different observed impacts of reservation as shown in Table 2.5.

Building upon the work of Engelhardt et al. [2022a], the contributions of this thesis are therefore summarized as follows:

Contributions

- Developing a method to solve the mixed service problem for ride-pooling services with long-term reservations with service guarantees. (Section 3.4)
- Modular integration of reservations into assignment and repositioning algorithms. (Section 3.4.2 + 3.4.4)
- Assessment of reservations for different spatio-temporal demand patterns. (Section 5.4.1)
- Evaluation for the three case studies Chicago, Munich, and Manhattan. (Section 5.4)

Chapter 3

Methodology

This chapter presents the proposed methodology to solve the ARP problem. First, a general description of the ARP problem is given in section 3.1. The discussion then shifts to the assignment problem in section 3.2, where the solution approach for assigning schedules to vehicles in the ARP service, addressing dynamically incoming on-demand requests is presented. A sampling-based approach for repositioning idle vehicles to meet future demand is presented in section 3.3. Finally, the treatment of pre-booking customers is discussed in section 3.4.

3.1 The Ride-Pooling Problem

3.1.1 General Problem Description

In the ARP service, a service provider operates a fleet of autonomous vehicles. During operation, customers continuously request trips from the fleet operator. The operator centrally controls its vehicles, i.e., the operator performs actions A_t at specific time steps t. These actions consist of assigning tasks to its vehicles, for example, assigning schedules to serve customer requests in a particular order, repositioning vehicles to a different location or sending vehicles to a charging station. In addition to these operational actions, the action space also includes strategical actions, which refer to operator decisions made on longer time scales. The operator can, for example, decide to change its fleet by acquiring or selling vehicles, or it can adapt the service area. Overall, the goal is to perform actions that optimize the operator's long-term objective O. Usually, this long-term objective corresponds to the operator's profit if a purely profit-oriented company conducts the operation. Nevertheless, also other objectives can be considered, e.g., societal aspects or environmental impacts, which may be particularly relevant for public transport operators. Actions are generally performed in a dynamic environment, which can be described by a system state S_t at time t. The system state consists of endogenous variables that are directly influenced by the operator's actions, like the vehicle positions, previously assigned tasks, or strategic decisions that have been made in the past. In contrast, exogenous variables are not directly influenced by the operator's actions, e.g., new customer requests or the traffic state.

The control problem to find suitable actions A_t can be mathematically formulated as

$$\max_{A} O \qquad O = \sum_{t} O_t(A_t, S_t) \tag{3.1a}$$

$$\text{s.t.:} \qquad S_{t+1} = \Omega(S_t, A_t, s_{t+1}) \qquad \forall t \tag{3.1b}$$

In Equation 3.1a the goal is to maximize the overall objective O, while O_t evaluates the incremental objective generated at time-step t. Equation 3.1b defines the state transition function Ω, which describes the system evolution based on the performed action A_t, the current system state S_t and the exogenous variables s_{t+1}, that refer to state changes independent of operator actions, e.g., new customers requesting trips[1].

If an operator could formulate the system dynamics Ω sufficiently accurately, finding the optimal actions A_t^* would solve the ARP problem. Nevertheless, this formulation requires evaluating future system states to determine the optimal action A_t^* at the current time t. Bellman provided a theoretical solution to this problem by introducing the Bellman equation [Bellman, 1957], which can be applied to evaluate optimal actions A_t^* if stochastic information about future exogenous state changes is available:

$$A_t^* = \arg\max_{A_t}(O_t(A_t, S_t) + \mathbb{E}[\sum_t \gamma^t O_{t+1}(A_{t+1}, S_{t+1})]) \tag{3.2}$$

$$\text{s.t.:} \quad S_{t+1} = \Omega(S_t, A_t, \tilde{s}_{t+1}) \qquad \forall t \tag{3.3}$$

The second term in Equation 3.2 evaluates the expected incremental objective generated in future time steps. The operator needs to predict future exogenous state changes $\tilde{s}_{t+1}$ to evaluate future states. As the prediction becomes less reliable the further it is in the future, weights on future rewards are dampened exponentially, described by the parameter $\gamma \in [0, 1]$.

In theory, approaches like Monte Carlo simulations can be used to sample future system states and thereby estimate future rewards, while dynamic programming approaches can be applied to determine the optimal actions A_t^*. Nevertheless, the number of possible decisions to make, i.e., variables to determine increases exponentially with the number of look-ahead time steps, which is generally known as "Course of Dimensionality" [Bellman, 1957]. For an ARP system, this approach would require solving a wide variety of DARPs, which is computationally intractable for large-scale ARP systems.

A general approach to overcome this problem is to decompose the decision-making process into smaller sub-problems, which can be solved sequentially and is often referred to as "Hierarchical Decision-Making". One possibility for this decomposition is to split the problem based on typical timescales of the system. One clear distinction between strategic (long-term) and operational (short-term) decisions can be made: For an ARP service, strategic decisions are usually made on a timescale of multiple weeks, months, or even years. Decisions include the operated fleet size, the fleet composition, the service area, service designs, or the general pricing strategy. They are characterized by high investment costs, e.g., adaptions to the infrastructure, or negotiating contracts with municipalities or other stakeholders. Too frequent changes in other strategic decisions like pricing schemes might also lead to negative customer perception. Typically, strategic decisions are grounded in a long-term projection of future demand, frequently relying on historical data and expert insights. Using this forecast, the operator can, for instance, employ simulations to ascertain the optimal fleet size, it's composition and service area to optimize the long-term objective O.

This thesis focuses on the operational decisions, which are made on a timescale of seconds, minutes, or hours. Operational decisions include the assignment of driving tasks to vehicles to serve incoming demand and provide service offers to these customers, the repositioning of idle

[1] Changes in endogenous variables are covered by the performed actions A_t based on the current state S_t.

vehicles to undersupplied areas, dispatching of vehicles to charging stations, and determining surge pricing factors to balance demand and supply.

As the literature review revealed, solving all these problems simultaneously is computationally intractable for large-scale ARP systems. A common approach is, therefore, to split operational decisions into further sub-problems. These sub-problems can be divided according to the timescale of the decisions, too. This thesis focuses on the following three sub-problems:

1. **Assignment**: The fleet operator reacts to the new customer requests in this sub-problem. The operator decides whether to accept or reject a customer request, communicates the decision to the customer and assigns schedules to its vehicles to serve these requests. As on-demand customers expect fast feedback regarding service availability, this decision-making process typically operates on a timescale of seconds or, at most, a few minutes

2. **Repositioning**: Repositioning aims to redistribute idle vehicles to undersupplied areas to increase service availability and vehicle utilization. This problem is usually solved on a timescale of multiple minutes to a few hours, reflecting timescales in which supply-demand imbalances occur typically.

3. **Reservation**: When customers can pre-book trips, multiple timescales come into play. On the one hand, the fleet operator needs to decide whether to accept a pre-booking request. On the other hand, the operator needs to assign schedules to vehicles to serve these pre-booked requests. While the former decision should be made on a timescale of seconds or minutes to provide fast feedback, the latter decision can be made on a timescale of hours or even days, depending on the pre-booking horizon. Even though time scales between assignment and reservation overlap, a separation in the decision-making process is reasonable as the acceptance decision can be decoupled from the current fleet state if the pre-booking horizon is sufficiently long.

These three sub-problems will be discussed in detail in the sections 3.2, 3.3, and 3.4.

There are further operational sub-problems that can be considered in the context of ARP services, but are not in the focus of this thesis. Firstly, the charging and fuelling of vehicles is an important operational task as it limits the range of vehicles and thus their potential availability. The decision-making process to determine when and where to charge or fuel vehicles can be considered as another sub-problem. However, it is argued that with technological advancement, the range of electric vehicles will further increase allowing a full day of operation without the explicit need for frequent charging. In such cases, charging processes could be scheduled during periods of low demand. Therefore, conflicts between charging and serving customers can be minimized, allowing for the separate handling of these tasks. Secondly, maintenance and cleaning of vehicles are other operational tasks that can be considered. Like charging and fuelling, this task can be shifted to off-peak times to avoid interfering with the assignment process. Thirdly, dynamic pricing, i.e., an increase in fare if most vehicles are utilized, can be considered as an operational task to balance supply and demand. Nevertheless, the development of dynamic pricing strategies is out of scope of this thesis and is therefore not considered in the following.

3.1.2 General Terminology

Before going into detail of the applied methodology, some general definitions and assumptions are introduced in this subsection that are valid for the remainder of this thesis.

Street Network The fleet operator provides a service in a specific operating area. Within this area, the operator needs a representation of the street network to determine routes, travel times, and distances between locations. This representation is provided by a directed graph $G = (N, E)$, where N is the set of nodes and E is the set of edges connecting these nodes. Each node $n \in N$ represents a location in the street network, which can be either a street intersection or a location on a street segment. Each edge $e \in E$ represents a street segment defined by its start and end node $n_s, n_e \in N$. An edge e is associated with a length l_e as a physical attribute. Additionally, each edge e is associated with a travel time $\tau_e(t)$ as a dynamic attribute, which reflects an estimate of the average vehicle travel time to pass this edge. This travel time depends on the time t to represent varying traffic states and congestion. The operator can use this graph to determine routes (a sequence of connected nodes), travel times $\tau(a, b)$, and distances $d(a, b)$ between two nodes $a, b \in N$. In this thesis, these attributes are determined by the fastest path (i.e., the path with minimal travel time) between two nodes. Well-known algorithms like Dijkstra's algorithm can be used to calculate these fastest paths.

Customers and Requests If customers intend to travel with the ARP service, they need to request a trip from the fleet operator, e.g., via a smartphone application. In case a customer requests a trip on-demand (i.e., expects a service as fast as possible), a request r is defined by its origin $o_r \in N$, destination $d_r \in N$, and a request time t_r. If a customer pre-books a trip, the request is additionally associated with an earliest pick-up time t_r^p. After a customer requests a trip, the operator provides feedback on whether it is possible to serve the customer. If a vehicle is available for service, the customer receives a service offer from the operator. Otherwise, a rejection is communicated to the customer. This service offer consists of attributes describing the expected characteristics of the trip. In this thesis, the offer consists of the expected pick-up time t_r^{ept} and the expected drop-off time t_r^{edt}.[2] The customer can then accept or decline this service offer and communicate the decision to the operator.

Vehicles and Schedules The operator controls a fleet of vehicles V, where $|V|$ is the overall fleet size available to the operator. Each vehicle $v \in V$ is associated with a capacity c_v representing the number of available passenger seats. The current state of a vehicle can be described by its current location, which can be either a node $n \in N$ or a general position on an edge $e \in E$. Additionally, each vehicle is associated with a set of customers currently on board the vehicle R_v^{ob} and a currently assigned schedule ψ_v. A schedule ψ_v is an ordered sequence of stops with tasks to be performed at these stops by the vehicle $v \in V$ in a given

[2] Generally, a charged fare will also be communicated to the customer. Nevertheless, as the impact of pricing is out of the scope of this thesis, it is dropped here for the sake of clarity. Depending on the service design, additional attributes might be suitable for communicating with customers, too. If the service performs pick-ups and drop-offs only at virtual stops, for example, the corresponding pick-up and drop-off location should also be communicated

order. A stop is defined by a location in the network (usually a specific node $n \in N$) and the task to be performed there. A task can, for example, refer to a boarding process, which is characterized by a set of customers boarding and/or alighting the vehicle and a duration t_B for the boarding process. If a repositioning trip is scheduled, the corresponding task only refers to driving to a particular location in the network. Between stops, vehicles move along a network route as specified above. The operator communicates new schedules, i.e., assignments, to the vehicles, while vehicles regularly report their current state to the operator.

Figure 3.1 illustrates the information flow between customers, the fleet operator, and the vehicles as considered in this thesis. The fleet operator acts as an intermediary between customers and vehicles, providing feedback to customers and assigning schedules to vehicles to serve these customers. To provide the best possible service, the operator tracts the current system state to make informed decisions to centrally control its fleet. The methodology for the assignment, repositioning, and reservation problem used in this control problem is discussed in the following sections.

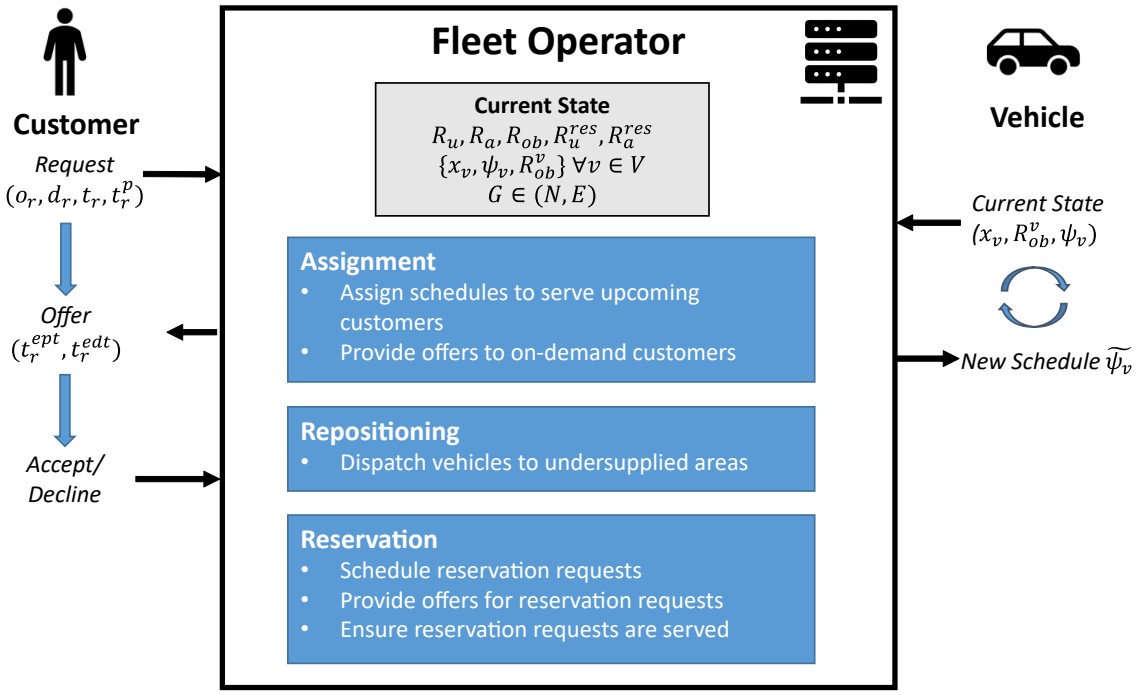


Figure 3.1: Information flow between customers, fleet operator and vehicles.

3.2 Assignment

This section deals with the assignment problem. For now, only the assignment of on-demand requests is considered. The incorporation of pre-booked requests is discussed in a following section. First, the general problem is described, and the corresponding optimization problem is formulated and discussed. Then, the solution approach of this thesis is presented, followed by a formulation of strategies to increase assignment reliability. Finally, benchmark algorithms are provided.

3.2.1 Operator Policy and Service Design

During operation, customers request trips continuously. The operator needs to react to these requests by providing feedback to the customer and assigning schedules to its vehicles. Typically, there are two different approaches to react to customer requests:

- **Immediate Response**: The operator provides offers immediately (or at least before the next customer is treated) after a customer requests a trip.
- **Batch Response**: The operator collects customer requests for a certain time period Δ_E and provides offers in batches.

There are advantages and disadvantages to both approaches. Immediate assignment allows the operator to react to new requests as fast as possible and provide quick feedback. Nevertheless, this approach requires the operator to solve the assignment problem for each new request, which can be computationally expensive. Alternatively, applying simple heuristics may be necessary. But this can lead to suboptimal fleet states.

Therefore, the batch response approach is applied in this thesis. Batch response allows the operator to solve the assignment problem only once for a batch of requests. The batch period (usually ranging from multiple seconds to a minute) defines the maximum run-time of the assignment algorithm, which can be used to apply more sophisticated but computationally heavier algorithms compared to the immediate response procedure. Additionally, responding in batches allows for making more informed decisions, as the operator can consider all requests in the batch to find the best possible assignment. Nevertheless, this approach leads to a delay in responding to the customer, which the customer might perceive negatively if the batch period is chosen too high.

Assignment Objective

Once the flow of information is established, the operator must define a policy to assign schedules to its vehicles, i.e., define criteria to evaluate the quality of a schedule and determine the best schedule. This policy is usually based on the operator's long-term objective, as stated in Equation 3.1a. Nevertheless, this long-term objective is not directly applicable to the assignment problem, as the operator can hardly estimate the long-term effects of current decisions. For example, choosing to assign a schedule to serve a customer purely based on generated profit might lead to high customer waiting and travel times if the schedule with the lowest cost is to pool as many customers as possible in one vehicle. Customers might even get rejected

if no profitable schedule can be found. This deterioration in service quality might lead to negative customer perception and, therefore, decrease demand and, consequently, profit in the future.

Therefore, the usual approach is to define a short-term objective ρ that can be evaluated based on the current system state S_t with the goal to approximately optimize the long-term objective O. Let $\rho(\psi)$ be a function that evaluates the short-term objective of a schedule ψ that is to be minimized (i.e., a measure of cost). Some typical components are (for reference see Table 2.3):

1. **Maximize Served Customers:**

$$\rho_s(\psi) = -\sum_{r \in R_\psi} p_r \ , \tag{3.4}$$

   with the goal to serve as many customers as possible in the current time step. Thereby, R_ψ refers to the set of customers served by the schedule and $p_r > 0$ refers a reward for serving customer r. The minus sign converts the maximization of served customers to a minimization to align with the definition of ρ.

2. **Minimize Customer Delay:**

$$\rho_{del}(\psi) = \sum_{r \in R_\psi} t_r^{do,\psi} - t_r \ , \tag{3.5}$$

   with the scheduled drop-off time $t_r^{do,\psi}$ of customer r based on this schedule.

3. **Minimize Vehicle Driving Time:**

$$\rho_t(\psi) = t_{end}^{\psi} - t \ , \tag{3.6}$$

   with the scheduled time t_{end}^{ψ} this schedule is completed and t being the current time.

4. **Minimize Vehicle Driving Distance:**

$$\rho_{dis}(\psi) = d^{\psi} \ , \tag{3.7}$$

   with d^{ψ} the distance the vehicle needs to drive to complete the schedule.

The idea of $\rho_s(\psi)$ and $\rho_{del}(\psi)$ is to assign schedules that ensure high service quality for customers, aiming to provide a high service rate and rapid service. On the contrary, $\rho_t(\psi)$ and $\rho_{dis}(\psi)$ are used to assign schedules that are efficient for the operator. Minimizing $\rho_{dis}(\psi)$ for example reduces fuel costs and vehicle wear, while minimizing $\rho_t(\psi)$ allows vehicles to be available to serve future customers as fast as possible, reducing the required fleet size. These different objectives are partially conflicting, e.g., minimizing $\rho_{dis}(\psi)$ might lead to high customer waiting times and thereby high $\rho_{del}(\psi)$. Therefore the operator needs to find a trade-off between these objectives. Additionally, solutions that are optimal for one objective might not be unique. For example, there might be multiple sets of schedules that serve the

same number of customers, Therefore, a weighted sum of these objectives as the short-term objective $\rho(\psi)$ can be used to define trade-offs and distinguishable solutions:

$$\rho(\psi) = \rho_s(\psi) + \omega_{del}\rho_{del}(\psi) + \omega_t\rho_t(\psi) + \omega_{dis}\rho_{dis}(\psi) \ , \tag{3.8}$$

with $\omega_{del}, \omega_t, \omega_{dis} \geq 0$ being the weights of the different objectives. Note that the weight for serving customers is implicitly set to 1 as it is absorbed by the reward p_r.

In fact, HYYTIÄ et al. [2012] showed that a minimization of vehicle travel time and delay time is the optimal choice from a queuing theoretical perspective[3]. Nevertheless, as empirical estimates for the values of ω_{del} and ω_{dis} are available, and there is a strong correlation between $\rho_{dis}(\psi)$ and $\rho_t(\psi)$ (via the average network speed), ω_t is set to 0 in this thesis, resulting in the assignment objective:

$$\rho(\psi) = \rho_s(\psi) + \omega_{del}\rho_{del}(\psi) + \omega_{dis}\rho_{dis}(\psi) \ , \tag{3.9}$$

Time Constraints and Feasible Schedules

It is not reasonable to assume that customers would wait for a very long time to be served by the service or that they accept very long travel times inside the vehicle. Therefore, hard time constraints are applied to guarantee a particular service quality. The time constraints (for on-demand customers) considered in this thesis are:

1. **Maximum Waiting Time:** The customer is only willing to use the service if the pick-up is scheduled no longer than t_{max}^{wait} after the request time t_r.

2. **Maximum Travel Time:** The customer is only willing to use the service if the in-vehicle travel time does not exceed $t_{r,max}^{travel} = (1 + \Delta_{det})\tau_r^{direct}$. Δ_{det} is a factor to account for the additional travel time compared to the direct travel time τ_r^{direct}.

The operator could set these time constraints within the real service, defining the service quality it wants to provide. Alternatively, the customer could also set these constraints to define the maximum waiting and travel time they are willing to accept. Independent of the interpretation, this thesis assumes that the same time constraints are applied to all customers.

By combining physical constraints and time constraints, a schedule is called *feasible* if

1. all on-board customers are scheduled to be dropped off,

2. customer pick-ups are scheduled before their corresponding drop-offs,

3. the vehicle capacity is never exceeded when en-route,

4. the maximum waiting time of no customers is exceeded,

5. the maximum travel time of no customers is exceeded.

[3] They also added a quadratic term for vehicle travel time but evaluated only a minor impact of this term.

The first two constraints must be ensured for any routing problem to create meaningful schedules. The third constraint limits the feasible schedules based on deployed vehicles, while the last two constraints are mainly based on customer preferences.

From an operational standpoint, time constraints have the advantage that the solution space for feasible schedules is reduced. Nevertheless, it is possible that no feasible schedule can be found for a customer request, e.g., if the customer requests a trip to a location that is not reachable within the maximum travel time. In this case, the operator needs to reject the customer request or retry the assignment with relaxed constraints[4].

3.2.2 Standard Formulation of the Assignment Problem

In each batch epoch, the goal of the operator is to assign new schedules to vehicles to incorporate yet unassigned requests and possibly re-consider already assigned schedules to improve the overall objective based on new information. The question the operator seeks to answer is: "Which are the optimal schedules to be assigned to fleet vehicles to serve yet unassigned and already assigned customer requests given the current system state and previously defined objective and constraints?" Finding an answer to this question is classically referred to as Dial-a-Ride Problem (DARP). In the following, the traditional DARP formulation based on Riley et al. [2019] is presented, which is used as a basis for further discussion and the presentation of solution approaches.

For clearer notation, the current vehicle state (x_v, R^v_{ob}, ψ_v) and its capacity c_v can be extended by the parameters t^v_0,q^v_0 and u^P_i. t^v_0 refers to the time, the vehicle can start its next trip. This is either the current time or the expected end time of an ongoing boarding process. $q^v_0 = |R^v_{ob}|$ is the number of customers currently on board. u^P_i describes the recorded pick-up time of customers who are already on board.

The goal is to decide on a (new) schedule for each vehicle. Stops in a newly assigned schedule are encoded by the decision variables x^v_{ij}, which are 1 if the vehicle v travels from node $i \in N$ to node $j \in N$ and 0 otherwise. Let the set of all nodes N be defined as $N = V \cup P \cup D \cup D_{ob} \cup \{s\}$. $V = 1, ..., |V|$ refers to the set of vehicle nodes. $P = \{|V|+1, ..., |V|+|R_u|+|R_a|\}$ is the set of pick-up nodes for yet unassigned requests R_u and already assigned but not yet picked-up requests R_a. Consequently, $D = \{|V| + |R_u| + |R_a| + 1, ..., 2(|V| + |R_u| + |R_a|)\}$ is the set of drop-off nodes. The set of drop-off nodes for on-board requests is defined by $D_{ob} = \cup_{v \in V} R^v_{ob}$. Finally, the sink node is defined by s. This artificial node is reachable at zero cost and is used to formulate the flow continuity constraints.

Each pick-up node $i \in P$ is associated with a request $r_i \in R_u \cup R_a$ which also refers to corresponding request attributes $(o_i, d_i, t^r_i, q_i, t^{ept}_i, t^{lpt}_i, \tau^{max}_i)$, the pick-up location o_i and the drop-off location d_i, the request time t^r_i, the group size q_i, the earliest pick-up time t^{ept}_i, the latest pick-up time t^{lpt}_i, and the maximum travel time τ^{max}_i. Each node is associated with the number of people to pick up ($\tilde{q}_i = q_i$) or to drop off ($\tilde{q}_i = -q_i$) at this node. t^B_i refers to the boarding time at node i.

[4] Dandl et al. [2021a] showed that a retry with the same constraints will never result in a later assignment if the assignment problem is solved optimally and network travel times do not change.

The corresponding Mixed-Integer Programming (MIP) can be formulated as follows:

$$\text{min} \quad \sum_{i \in P} p_i z_i + \omega_{del} \sum_{v \in V} \sum_{i \in P} (u_i^v - t_i^{ept}) + \omega_{dis} \sum_{v \in V} \sum_{i \in P} d_i^v \tag{3.10a}$$

$$\text{s.t.} \quad (\sum_{v \in V} \sum_{j \in N} x_{ij}^v) + z_i = 1 \qquad \forall i \in P \tag{3.10b}$$

$$\sum_{j \in N} x_{ij}^v - \sum_{j \in N} x_{ji}^v = 0 \qquad \forall i \in N \backslash V \cup \{s\}, \forall v \in V \tag{3.10c}$$

$$\sum_{j \in N} x_{vj}^v = 1 \qquad \forall v \in V \tag{3.10d}$$

$$\sum_{j \in N} x_{js}^v = 1 \qquad \forall v \in V \tag{3.10e}$$

$$\sum_{j \in N} x_{ij}^v - \sum_{j \in N} x_{n+i,j}^v = 0 \qquad \forall i \in P, \forall v \in V \tag{3.10f}$$

$$\sum_{v \in V} \sum_{i \in N} x_{ij}^v = 1 \qquad \forall j \in R_a \tag{3.10g}$$

$$\sum_{i \in N} x_{ij}^v = 1 \qquad \forall v \in V, \forall j \in R_{ob}^v \tag{3.10h}$$

$$u_j^v \geq (u_i^v + t_i^B + \tau_{ij}) x_{ij}^v \qquad \forall i, j \in N, \forall v \in V \tag{3.10i}$$

$$d_j^v \geq (d_i^v + d_{ij}) x_{ij}^v \qquad \forall i, j \in N, \forall v \in V \tag{3.10j}$$

$$u_0^v \geq t_0^v \qquad \forall v \in V \tag{3.10k}$$

$$d_0^v \geq 0 \qquad \forall v \in V \tag{3.10l}$$

$$u_i^v \geq t_i^p \qquad \forall i \in P, \forall v \in V \tag{3.10m}$$

$$u_i^v \leq t_i + t_{max}^{wait} \qquad \forall i \in P, \forall v \in V \tag{3.10n}$$

$$\tau_i^{direct} \leq u_{n+i}^v - (u_i^v + t_i^B) \leq (1 + \Delta_{det}) \tau_i^{direct} \qquad \forall i \in P, \forall v \in V \tag{3.10o}$$

$$\tau_i^{direct} \leq u_i^v - (u_i^P + t_i^B) \leq (1 + \Delta_{det}) \tau_i^{direct} \qquad \forall i \in R_{ob}^v, \forall v \in V \tag{3.10p}$$

$$w_j^v \geq (w_i^v + \tilde{q}_j) x_{ij}^v \qquad \forall i, j \in N, \forall v \in V \tag{3.10q}$$

$$0 \leq w_j^v \leq c_v \qquad \forall j \in N, \forall v \in V \tag{3.10r}$$

$$x_{ij}^v, z_i \in \{0, 1\} \qquad \forall i, j \in N, \forall v \in V \tag{3.10s}$$

Objective 3.10a is a reformulation of Equation 3.9 with the goal to minimize unserved requests, traveled distance, and customer delay for all vehicle schedules. The real-valued decision variables u_i^v and d_s^v refer to the arrival time and the driven distance at node i for vehicle v, respectively. The first term penalizes unassigned requests by a factor p_i indicated by the integer decision variable z_i[5]. Constraints 3.10b ensure that customers are registered as unassigned if they are not part of a schedule. Constraints 3.10c is the flow conservation, while constraints 3.10d and 3.10e ensure that the vehicle starts at their corresponding source node and ends at the sink node. With $n = |R_u| + |R_a|$, constraints 3.10f ensure that each

[5] In this formulation, unserved customers are penalized instead of giving a reward for served customers in Equation 3.4, which is equivalent in terms of the resulting optimal solution.

request that is picked up is also dropped off. Constraints 3.10g and 3.10h ensure that each assigned and on-board request is assigned again. Constraints 3.10i to 3.10l set the decision variables u_i^v and d_s^v, while τ_{ij} and d_{ij} refer to the travel time and distance between node i and j. Constraints 3.10m and 3.10n ensure that no customer is picked up before the earliest and latest pick-up times, respectively. Constraints 3.10o and 3.10p ensure that the vehicle arrives at the drop-off location of a request before the maximum travel time elapsed. Constraints 3.10q and 3.10r ensure that the vehicle capacity is not exceeded. Finally, constraints 3.10s define the binary decision variables.

In theory, standard solvers like Gurobi[6] or CPLEX[7] can be used to solve this problem and obtain new assignments for the given batch epoch. Given the linear objective function ρ, this MIP can be reformulated as a Mixed-Integer Linear Programming (MILP) by using big-M formulation for the non-linear constraints 3.10i, 3.10j and 3.10q. Nevertheless, as MILPs are still NP-hard, standard methods fail to find optimal solutions for large-scale problems in a reasonable time. Therefore, in section 3.2.4, methods are presented that can be used to find good solutions for large-scale problems and even optimal solutions for medium-scale problems.

3.2.3 Dynamism and Re-Assignment

Solving the assignment problem as outlined in the previous section would yield the optimal solution within the current epoch. Nevertheless, it is essential to discuss dynamic aspects that occur from the operator's and customer's point of view when solving the assignment problem epoch by epoch. In the current formulation, the following changes for assigned customers can occur between epochs:

1. If a previously assigned request is not added to the set of assigned requests R_a, a consecutive assignment is not guaranteed.

2. A different vehicle might be assigned to a customer request, leading to a re-assignment.

3. The stop sequence of a vehicle might change due to new requests, resulting in a change in scheduled pick-up and drop-off times for other assigned customers.

From an operator's point of view (or rather from the perspective of the assignment optimality) keeping as much flexibility for optimization as possible is desirable. Nevertheless, computational time constraints might become a limiting factor as the explorable search space becomes larger with increased flexibility. From the customer's point of view, however, it is rather instead that the operator provides a reliable service. This reliability can be categorized into the following aspects:

1. **Service Reliability:** Once the accepted trip is communicated to the customer, the customer expects to be served. This rather obvious aspect can be integrated into the mathematical model by shifting a new request $r \in R_u$ to R_a once assigned to a vehicle. As this is a central aspect of providing a reliable service, it will be implemented in all scenarios tested and is not further discussed in this thesis.

[6] https://www.gurobi.com/
[7] https://www.ibm.com/de-de/products/ilog-cplex-optimization-studio

2. **Service Vehicle Reliability:** This aspect, often referred to as re-assignment, describes the possibility that the vehicle scheduled to pick up the customer may not be the same vehicle that ultimately carries out the pickup. Re-assignment occurs if the vehicle is re-scheduled to serve another request if the overall assignment objective can be improved. From a customer's point of view, this might be unfavorable as the customer cannot track the vehicle on the smartphone application, making the service less comprehensible.

3. **Pick-up Time Reliability:** This aspect describes the reliability and stability of the communicated expected pick-up time to the customer. While the pick-up time variation is constraint by the maximum waiting time constraint t_{max}^{wait}, the expected pick-up might be updated within each optimization epoch if vehicles are rescheduled to accommodate new requests. This can happen if a new customer is scheduled to be picked up before another customer is assigned to another vehicle, or the customer itself is assigned to another vehicle. Other sources of variations that are not considered in detail in this thesis are delays in traffic, late cancellations or no-shows of customers.

4. **Travel Time Reliability:** Lastly, this aspect describes the reliability and stability of the communicated expected travel time to the customer. Similar to the pick-up time reliability, the overall travel time of the request r is constrained by the maximum travel time $t_{r,max}^{travel}$. As new customers are assigned to the vehicle, the expected travel time may increase due to additional pickups along the route.

As a ride-pooling service is, per design, inherently dynamic and stochastic by offering service on-demand, complete reliability concerning the aspects mentioned above is not possible. Nevertheless, it is important to quantify the trade-off between the operator's and customer's point of view and to develop a solution approach that can balance these aspects.

First, the following section deals with a solution approach for the assignment problem defined in the previous section 3.2.2. Then, in section 3.2.5, the discussed aspects of re-assignment and reliability are reiterated and strategies for re-assignment are discussed.

3.2.4 Solving the Assignment Problem

The algorithm to solve the assignment problem extends the method proposed by Alonso-Mora et al. [2017b]. The idea is to decompose the DARP problem (Equations 3.10a to 3.10s) into two main steps. First, feasible candidate schedules for each vehicle are created that could serve a specific set of requests. Second, the best candidate schedules are assigned to vehicles by solving an assignment problem.

In the following, the assignment problem is reformulated first before the creation of candidate schedules is discussed.

Reformulation of the Assignment Problem

Let $\psi_i(R_\gamma, v)$ be a feasible schedule for vehicle v that serves precisely the set of requests R_γ. As there are generally multiple feasible schedules (permutations of stops) for a vehicle and a set of requests, the index j refers to the j-th feasible schedule. The candidate schedule

$\psi^*(R_\gamma, v)$ for vehicle v to serve the set of requests R_γ can then be defined as that schedule that minimizes the assignment objective:

$$\psi^*(R_\gamma, v) = \arg\min_j \rho(\psi_j(R_\gamma, v)) \ . \tag{3.11}$$

For shorter writing, let further $\rho(v, R_\gamma) = \rho(\psi^*(R_\gamma, v))$ be the objective value of the candidate schedule.

The assignment problem can then be formulated as an Integer Linear Programming (ILP) as follows:

$$\text{minimize} \quad \sum_{v \in V} \sum_{\gamma \in \Omega_v} \rho(v, R_\gamma) \cdot z_{v,\gamma} \tag{3.12a}$$

$$\text{s.t.} \quad \sum_{\gamma \in \Omega_v} z_{v,\gamma} \leq 1 \qquad \forall v \in V \tag{3.12b}$$

$$\sum_{v \in V} \sum_{\gamma \in \Omega_{i,v}} z_{v,\gamma} = 1 \qquad \forall i \in R_a \tag{3.12c}$$

$$\sum_{v \in V} \sum_{\gamma \in \Omega_{i,v}} z_{v,\gamma} \leq 1 \qquad \forall i \in R_u \tag{3.12d}$$

$$z_{v,\gamma} \in \{0, 1\} \qquad \forall v \in V, \gamma \in \Omega_v \tag{3.12e}$$

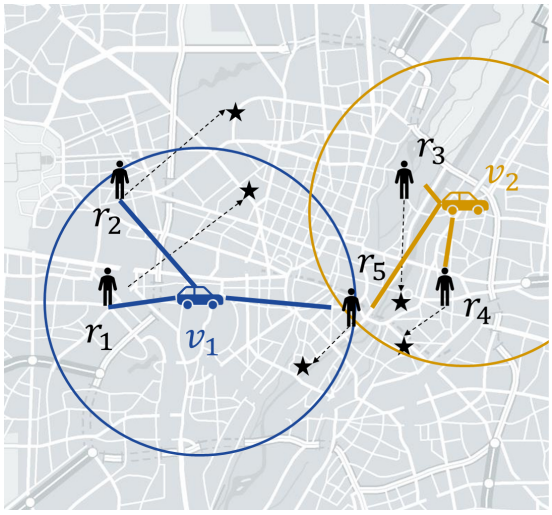
The objective of the assignment problem is to minimize the sum of the objective values of the candidate schedules for each vehicle. $z_{v,\gamma}$ is a binary decision variable that is 1 if the candidate schedule $\psi^*(R_\gamma, v)$ is assigned to vehicle v and 0 otherwise. Ω_v refers to the set of all feasible schedules for vehicle v, while $\Omega_{i,v}$ refers to the set of all feasible schedules for vehicle v that serve request i. Constraints 3.12b ensure that each vehicle is assigned at most one candidate schedule. Constraints 3.12c ensure that each assigned request is served by exactly one vehicle. The equality ensures that customers assigned in a previous optimization epoch (R_a) will remain to be served. Constraints 3.12d ensure that at most one vehicle serves each unassigned request (R_u). These requests might, therefore, remain unassigned if no feasible schedule can be found that improves the objective The binary decision variables $z_{v,\gamma}$ are defined in constraints 3.12e.

The advantage of this formulation is that it effectively decouples the assignment problem from the creation of candidate schedules. It turns out that the major computational effort is required to create the candidate schedules while solving the problem above can be done efficiently by standard solvers. Additionally, a subset of all feasible schedules can be considered in this assignment problem while still feasible assignments can be obtained[8]. This enables heuristics or timeouts in the search procedure to compute only a subset of candidate schedules to ensure real-time performance.

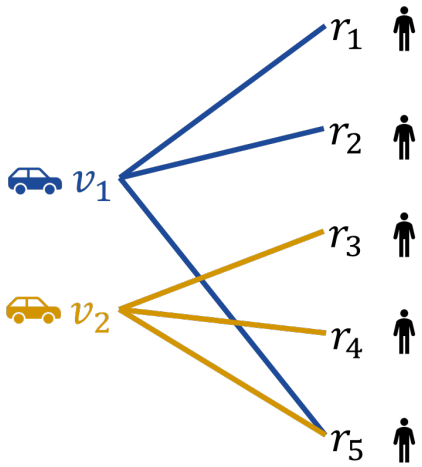
Candidate Schedule Creation

The creation of candidate schedules is the central component of the presented algorithm. The goal is to create a set of (if possible all) feasible schedules for each vehicle that could

[8] To guarantee a feasible solution of the assignment problem a minimum set of feasible schedules has to be provided. However, a trivial feasible solution to the problem can always be obtained by using the currently assigned schedules as input to the problem.



(a) Constructing the RV Graph (Circle indicates travel distance within maximum waiting time constraint).



(b) Resulting RV Graph.

Figure 3.2: Sketch for the RV Graph.

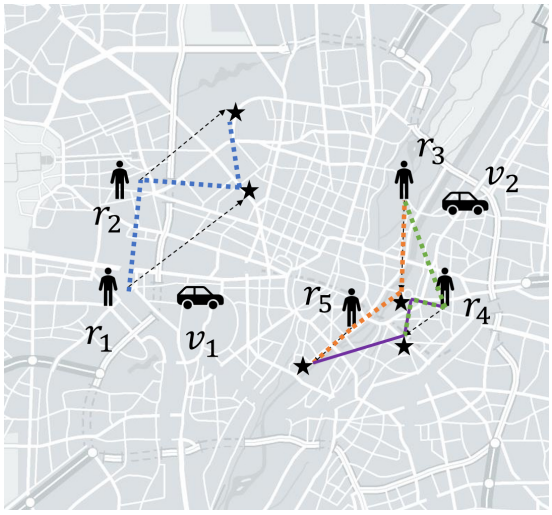
serve a given set of requests. A straightforward approach would be to exhaustively search for all vehicles and all combinations of requests, and solve a single vehicle DARP to obtain all feasible candidate schedules. Nevertheless, the number of possible combinations to check increases exponentially with the number of requests[9]. Consequently, this approach is not computationally tractable for large-scale problems.

The main idea first proposed by Alonso-Mora et al. [2017b] is to exploit the fact that in a typical setting for a ARP service, the majority of all possible combinations to check will not yield a feasible schedule for assignment. This is either due to time constraint violations or because the vehicle capacity is exceeded. The idea of the algorithm therefore is to develop an efficient search strategy to only compute vehicle schedules for a subset of all possible request combinations that fulfill some necessary requirements that need to hold for a feasible schedule to exist.

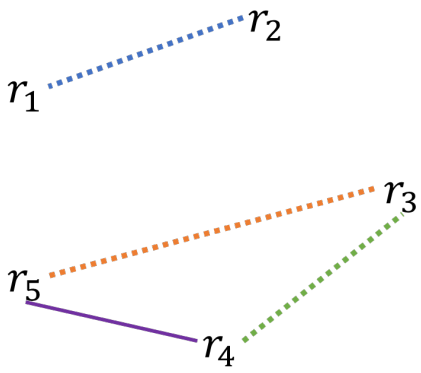
The search strategy consists of a graph-based representation of mutual feasibility of request-vehicle and request-request combinations. An iterative schedule-building procedure exploits these graphs to prune the search space for feasible schedules. In the following, first, the request-vehicle and request-request graphs are described, followed by the iterative schedule-building procedure.

Request-Vehicle (RV) Graph The RV graph (Figure 3.2) encodes the subset of all available vehicles that are able to serve a given request, because the maximum waiting time constraint t_{max}^{wait} limits the eligible vehicles to those in the vicinity of the request's origin position. Consequently, vehicles and requests are represented as nodes in the graph. An edge between a

[9] On the one hand, the number of request combinations increases exponentially ($\mathcal{O}(|V|2^{|R|})$ with the number of requests $|R|$). On the other hand, the number of possible stop permutations to solve the single vehicle DARP increases exponentially with the number of requests, too.



(a) Constructing RR Graph (feasible shared routes are shown in color).



(b) Resulting RR Graph.

Figure 3.3: Sketch for the RR Graph.

vehicle and a request is created if the vehicle can reach the origin position of the request before the maximum waiting time constraint is violated. As a direct approach of a given vehicle v to the request is the fastest possible way to serve the request r, any other schedule of this vehicle to serve a subset of requests R_γ with $r \in R_\gamma$ cannot be feasible if there is no edge between the vehicle v and the request r in the RV graph. These schedules can, therefore, be excluded from the search procedure. From an implementation standpoint, the RV graph can be computed efficiently by, for example, using Dijkstra's algorithm backward search for each request to be computed with the request origin node as the source node and the current vehicle locations as the target nodes. The algorithm can be stopped as soon as the travel time of the explored nodes exceeds the maximum waiting time.

Request-Request (RR) Graph The RR graph (Figure 3.3) encodes a necessary condition for the existence of a feasible schedule that serves at least two requests at once. Based on the idea of shareability networks proposed by Santi et al. [2014], graph is created with requests represented as nodes. An edge between two requests is added if any feasible schedule exists that serves both requests at once (either by a shared or scheduled trip) by any hypothetical vehicle available at the origin of one of both requests. This requires checking six different combinations of schedules for each request pair (for each request being picked up first, there is one sequential schedule and two shared schedules, which have been indicated in Fig. 2.2 in the previous chapter). If no edge between two requests exists, there cannot be any feasible schedule by any vehicle that serves both requests simultaneously, as the vehicle's approach will further delay a schedule. Additionally, suppose no edge between two requests exists. In that case, there also cannot be any feasible schedule by any vehicle that simultaneously serves a subset of requests that includes both requests.

Iterative Schedule Building To create the candidate schedules for each vehicle, a guided search procedure can be applied that utilizes the RV and RR graph and terminates once the reduced search space for feasible schedules is exhausted. Let $|R_\gamma|$ be the grade, i.e. the number of requests served by the candidate schedule $\psi^*(R_\gamma, v)$. The following conditions need to hold for the feasible schedule to exist:

1. There has to be an edge in the RV graph between the vehicle v and each request in R_γ.
2. There has to be an edge in the RR graph between each pair of requests in R_γ.
3. For any feasible schedule with grade $|R_\gamma| > 1$ to exist, the corresponding schedules with grade $|R_\gamma| - 1$ resulting from removing one of its request $(\{\psi^*(R_\gamma \setminus \{r_k\}, v) \forall r_k \in \gamma\})$ have to exist, too. For example, for the existence of a feasible schedule $\psi^*(\{r_1, r_2, r_3\}, v)$ it is necessary (but not sufficient) that the feasible schedules $\psi^*(\{r_1, r_2\}, v)$, $\psi^*(\{r_1, r_3\}, v)$ and $\psi^*(\{r_2, r_3\}, v)$ exist, too.

The resulting tree-based requirements are sketched in Fig. 3.4. These conditions can guide the search procedure by creating feasible schedules with grade $|R_\gamma| = 1$ and iteratively building schedules with higher grades. For schedules of grade 1, the existence of an edge in the RV graph between the vehicle and the request is sufficient to create a feasible schedule. Because of the third condition, only combinations of R_γ that add upon already computed and existing schedules of grade $|R_\gamma| - 1$ need to be considered. Before a new schedule is computed, it is checked whether all necessary conditions above are fulfilled.

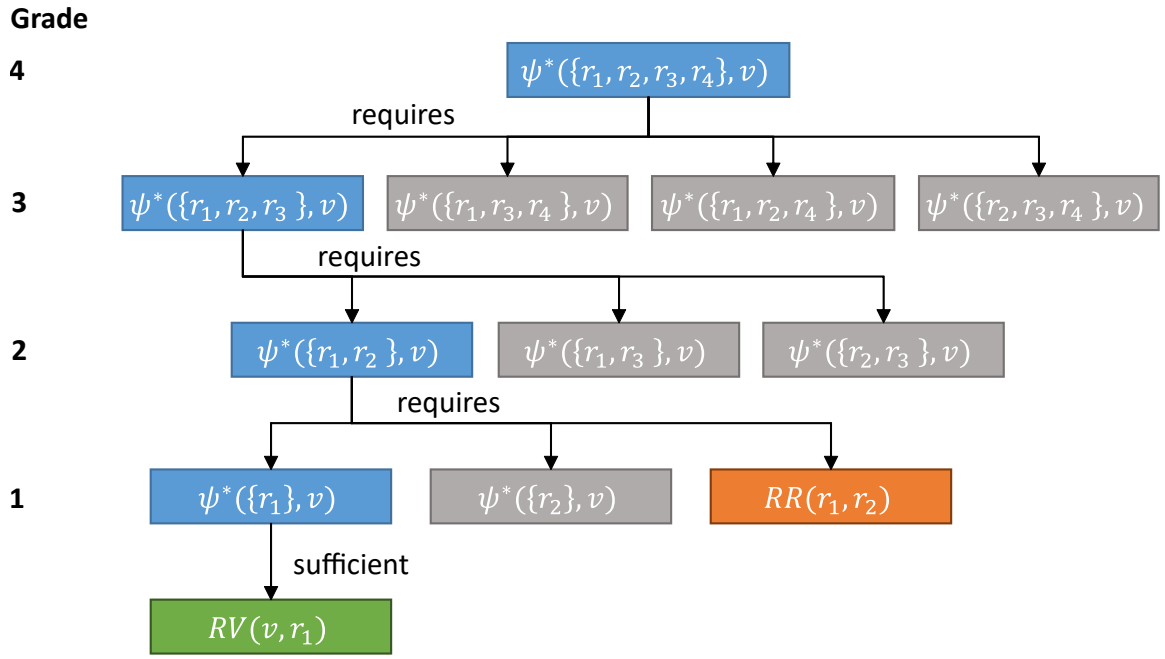


Figure 3.4: Tree-based sketch of requirements for the existence of $\psi^*(\{r_1, r_2, r_3, r_4\}, v)$. The lower branches of the tree are only shown for colored boxes.

Running this algorithm to termination and feeding the candidate schedules into the assignment problem (Eq. 3.12a to 3.12e) will yield the optimal assignment of schedules to vehicles

for the given batch epoch and therefore corresponds to the exact solution of the DARP from Eq. 3.10a to 3.10s. In theory, the NP-hardness of the DARP is not reduced by this approach, and termination of large-scale problems in reasonable computational time is not guaranteed. Nevertheless, as long as time constraints are tight (which is a reasonable assumption for most ARP services) and therefore the RR and RV graphs are sparse, this algorithm can solve large-scale problems in reasonable time.

In the following, details of the original implementation from ALONSO-MORA et al. [2017b] are presented before the adaption of the implementation of the algorithm in this thesis is discussed.

Original Implementation In the original implementation of ALONSO-MORA et al. [2017b], RV and RR graphs are computed from scratch in each optimization epoch. When constructing the actual schedules (i.e., once all three above-mentioned conditions hold for a vehicle with a set of requests) in the iterative schedule-building process, the authors use two options to create the schedules:

1. If the grade of the schedule is smaller than 5, an exhaustive search is applied to solve the single vehicle DARP for the given set of requests.

2. If the grade of the schedule is at least 5, a schedule is constructed by inserting the new request into the corresponding lower-grade candidate schedule to reduce the computational time (see Algorithm 1).

Algorithm 1 Insertion Heuristic

ψ_{best} = None
v_{best} = None
for all $v \in V^{ca}$ **do**
  $\psi_{\tilde{k}}(v; R_\psi, P_\psi \cup \{r_i^p\})$ = insert$(\psi_k(v; R_\psi, P_\psi), r_i^p)$
  if $d(\psi_{\tilde{k}}(v; R_\psi, P_\psi \cup \{r_i^p\})) - d(\psi_k(v; R_\psi, P_\psi)) < (1 - \tau_{th}) d(o_i^p, d_i^p)$ **then**
    if $\phi(\psi_{\tilde{k}}(v; R_\psi, P_\psi \cup \{r_i^p\})) < \phi(\psi_{best})$ **then**
      $\psi_{best} \leftarrow \psi_{\tilde{k}}(v; R_\psi, P_\psi \cup \{r_i^p\})$
      $v_{best} \leftarrow v$
    end if
  end if
end for
if $v_{best} \neq None$ **then**
  assignSchedule(v_{best}, ψ_{best})
  $P_u \leftarrow P_u \setminus \{r_i^p\}$
end if

Improving the Algorithm: Database for Feasible Schedules

The idea of this thesis, first presented in ENGELHARDT et al. [2019], is that most computations made in a given optimization epoch can be reused in the next optimization epoch. As the

time between two consecutive optimization epochs is usually short, this approach conjectures that many computed schedules maintain their feasibility. Consequently, instead of rebuilding feasible candidate schedules from scratch again, the idea is to store the computed schedules in a database and only check if the schedules are still feasible in the following optimization epoch.

Instead of computing only one candidate schedule for each vehicle and set of requests, (possibly) all feasible schedules (i.e., stop permutations) are stored in an object referred to as Vehicle-to-Request-Bundle (V2RB). The V2RB $\Psi(v, R_\gamma) = \{\psi_i(R_\gamma, v) \forall i\}$ collects all feasible permutations i of stops for vehicle v to serve the set of requests R_γ. The V2RB $\Psi(v, R_\gamma)$ can be created by inserting the request $r \in R_\gamma$ into all feasible schedules collected in the V2RB $\Psi(v, R_\gamma \setminus \{r\})$ of the corresponding lower grade. The objective value of the V2RB is the objective value of the representing schedule $\psi^*(R_\gamma, v)$ that minimizes the assignment objective.

In between optimization epochs, two things can change that influence the feasibility of schedules:

1. Customers can board or alight a vehicle, s a previously feasible schedule might not feasible anymore if this boarding process was not scheduled.

2. Vehicles can move. If a vehicle does not move toward the next planned stop in a given schedule, it might no longer arrive at any stop in the schedule in time, and time constraints may be violated.

3. Network travel times can change. If travel times change, previously feasible schedules might become feasible, and new schedules that were tested unfeasible before might become feasible.

Therefore, given the V2RBs from the previous assignment epoch, an iterative search procedure can be applied to remove all schedules from the V2RBs that are not feasible anymore and the V2RB itself if no feasible schedule remains. First, boarding processes are considered: Once a customer r_x enters vehicle v_y, all schedules of all V2RBs of vehicle v_y are deleted that did not schedule the boarding process of r_x. Similarly, all V2RBs of all other vehicles that do contain the customer r_x can be removed, too. In the second step, the movement of vehicles is considered: Starting V2RBs of grade 1, the feasibility with updated vehicle locations is checked, and the objective values (and therefore the representing candidate schedule) are recomputed. If a schedule is not feasible anymore, it is removed from the V2RB, and similarly, the V2RB is removed from the database if no feasible schedule remains. For higher grade V2RBs, it is first checked if corresponding lower grade V2RBs are still feasible, and only if this is the case the feasibility of the V2RB is checked. Otherwise, the requirements for the existence of a feasible schedule are not fulfilled, and the V2RB can be removed directly from the database.

Once the updated V2RBs and the corresponding database is computed, the above-mentioned algorithm is applied to build new V2RBs for new requests only.

If the whole tree is entirely built in each time step and network travel times remain the same, the solution space is completely explored, and the optimal solution for the DARP can

be found by solving the assignment problem[10].

If travel times change, new schedules might become feasible that were tested not feasible before and are therefore not part of the V2RB database. In this case, the V2RB database has to be rebuilt from scratch. Additionally, it can occur that even currently assigned schedules are not feasible anymore. In this case it has to be ensured that a feasible assignment is still available in order to find a solution to the assignment problem. Therefore, currently assigned schedules are included in the V2RB database even if they became infeasible due to travel time updates. Nevertheless, building new schedules based on these infeasible schedules is prohibited.

Heuristics

Although the described algorithm is a very efficient procedure to compute feasible candidate schedules, the curse of dimensionality still limits the computational tractability of large problems. In this case, heuristics can be applied to prune the search space, reducing the computational effort in the schedule-building process. In this thesis, the following heuristics are applied and tested for their impact on the solution quality and computational time:

Limited Number of Feasible Schedules Per V2RB *(LS)* Because the number of feasible schedules to serve a given set of requests can increase exponentially with the number of requests, it can be computationally infeasible to compute all feasible schedules of an V2RB of high grade. Nevertheless, as only the best schedule is assigned to a vehicle, keeping all feasible schedules stored in the V2RB might not be necessary. The idea is therefore to limit the number of feasible schedules stored in the V2RB to a certain number N_{V2RB}^{max}. After a new V2RB is created, only the N_{V2RB}^{max} best schedules are stored. The best possible schedule is still found for the first V2RB that triggers this heuristic. This is not the case if further higher grade V2RBs are created based on the corresponding V2RB. The optimal solution of the corresponding single vehicle DARP might not be accessible anymore by simply inserting a new request into the reduced set of schedules. A similar heuristic is applied in the algorithm of Alonso-Mora et al. [2017a]. As the suggested algorithm does not store all feasible schedules, they limit the search procedure by only solving the insertion of a new request into the best lower-grade schedule instead of solving the single vehicle DARP to optimality once the number of associated requests exceeds a specific number. Therefore, the proposed method can be seen as a generalization of this heuristic.

Candidate Vehicle Reduction Per Request *(RV)* Depending on the number of vehicles available, the number of candidate vehicles to serve a request in the RV graph can be high, resulting in numerous insertions that need to be checked. Nevertheless, only one vehicle can serve the request. The idea is, therefore, to limit the number of candidate vehicles that are used for building V2RBs for a request to a certain number N_{RV}^{max} and therefore prune the RV graph. Alonso-Mora et al. [2017a], for example, chose the N_{RV}^{max} vehicles that are closest to the request's origin position. More refined methods are tested by Engelhardt et al. [2020] and Dandl [2022], where vehicle selection heuristics were based on currently assigned

[10] see Dandl et al. [2021a] for a proof of this statement.

schedules and an equal distribution of requests across vehicles. The similarity of these methods is that the set of candidate vehicles is first determined, and afterward, the V2RBs are created. Nevertheless, this approach ignores possible re-assignments of requests to other vehicles that might be necessary to improve the overall assignment objective. Therefore, the idea described here is to prune the RV graph after making the first assignment. For an incoming request, all vehicles within the complete RV graph are first considered, and corresponding V2RBs are created. Once the assignment problem is solved, candidate vehicles are sorted based on the objective values of the best V2RBs that includes the new request. The N_{RV}^{max} vehicles with the best objective values are kept in the RV graph, and the remaining vehicles are removed. To ensure feasibility, it is enforced that the assigned vehicle is added to this set if it is not already included. Vehicles are removed by deleting V2RBs from the database that includes the vehicle and the corresponding request. The reduced computational effort comes from fewer V2RBs eligible for insertion in upcoming batch epochs.

Search Timeout per Vehicle *(TO)* For real-time applications, it might still be necessary to limit the computational time of the schedule-building process even further. A brute-force approach to limit the computational time is to set a timeout $\nu_{TO,v}$ for the schedule-building process for each vehicle. If the timeout is reached, the schedule-building process is stopped, and the V2RBs found so far are eligible for assignment for each vehicle. The assumption is that this time-out is mainly triggered for vehicles with many requests in the vicinity. Consequently, a high number of feasible schedules need to be computed. Nevertheless, as a request can only be assigned to one vehicle and the number of requests to be served by a single vehicle is limited, many of the computed schedules might not be needed for the assignment problem. When a timeout is applied, the order of inserting new requests into the V2RBs becomes relevant. Therefore, new requests are first inserted into the V2RBs that include only currently assigned requests to the vehicle. These insertions likely lead to good and feasible assignments if an insertion is possible. Afterward, the new requests are inserted in random order into the remaining V2RBs. While this heuristic can ensure in-time termination of the algorithm, the significant disadvantage in simulation studies is the dependence of the solution quality on available computational resources and efficiency in the implementation, reducing comparability of the results.

3.2.5 Strategies for Increased Reliability

As discussed in section 3.2.3, solving the formulated assignment problem epoch by epoch might lead to reliability issues from a customer's point of view because re-assignments of vehicles to serve requests can result in deviations from the initially communicated trip characteristics. Nevertheless, from an operator point of view, re-assignments can lead to a more efficient assignment of requests to vehicles and therefore an overall more efficient operation.

To explore the trade-off between the operator's and customer's point of view, the following scenarios are tested in this thesis:

1. **Full Re-Assignment:** In this scenario, the assignment problem is solved epoch by epoch, and the optimal solution is assigned. Therefore, this scenario allows unconstrained customer-vehicle re-assignment in each epoch until the customer is picked up.

2. **No Re-Assignment:** After the initial assignment of a customer to a vehicle, further re-assignments are not allowed. Technically, this can be achieved by deleting all feasible V2RBs that contain the customer but do not belong to the vehicle the customer is currently assigned to.

3. **Temporally Limited Re-Assignment:** In this scenario, full re-assignment is allowed until a threshold time $t_{th}^{reassign}$ before the scheduled customer pick-up. After this threshold time, the assignment is fixed, and no re-assignment is allowed. This scenario converges to the *no re-assignment scenario* if $t_{th}^{reassign} = 0$ and to the *full re-assignment* scenario if $t_{th}^{reassign} = t_{max}^{wait}$. From a customer point of view, it is possible in this scenario to display the assigned vehicle on the smartphone application once the threshold time is reached, leading to an increased *service vehicle reliability*. At the same time, the operator can still exploit optimization potential.

4. **Re-Assignment Penalty:** In this scenario, a penalty $p_{reassign} > 0$ is introduced directly in the objective function for re-assigning a customer to another vehicle by setting

$$\rho(\psi) \rightarrow \rho_{reassign}(\psi) = \rho(\psi) + \sum_{r \in R_\psi} (1 - \delta_{v_\psi, v_r}) p_{reassign} \ . \tag{3.13}$$

   Here, δ_{v_ψ, v_r} is the Kronecker delta that is 1 if the vehicle v_ψ scheduled for plan ψ is the same as vehicle v_r, which is currently assigned to serve request r, and 0 otherwise. This formulation still allows full re-assignment, but depending on the choice of $p_{reassign}$, re-assignments with limited optimization potential are discouraged.

5. **Pick-up Time Window Tightening:** In this scenario, the goal is to increase the *pick-up time reliability*. Once an initial assignment of a customer to a vehicle is made, the pick-up time window for future (re-)assignments is tightened to a time interval Δ_{TW}. The expected pick-up time t_r^{pu} is determined from the first assigned schedule. For all upcoming re-assignments, the pick-up time window is then set to $[t_r^{pu} - \Delta_{TW}/2, t_r^{pu} + \Delta_{TW}/2]$. If the bounds are above (below) the current latest pick-up time (earliest pick-up time) of the request, the corresponding upper (lower) bound is set to the latest (earliest) pick-up time, while the other bound is set to maintain the interval of Δ_{TW}. This scenario allows for full re-assignment but increases the *pick-up time reliability* by reducing other possible allowed tours. All infeasible schedules will be removed automatically within the presented algorithm when the feasibility of previous candidate schedules is checked.

Finally, the flowchart in Fig. 3.5 summarizes the main actions within an epoch and state changes in between epochs described in this chapter.

3.2.6 Benchmark Algorithms

To evaluate the efficiency, the proposed algorithm is compared to the following algorithms:

Insertion Heuristic The insertion heuristic based on Jaw et al. [1986] is a popular assignment algorithm because of its simplicity and computational efficiency. Incoming requests are

inserted one by one into currently assigned schedules of vehicles that can reach the request within the maximum waiting time constraint. The insertion (Algorithm 1) selects the feasible schedule that reduces the objective value the most. Compared to the proposed algorithm, optimality losses emerge because the insertion heuristic does not allow for re-assignment of requests in a later epoch. Secondly, requests are assigned sequentially instead of finding a collective solution in batch. And thirdly, permutations of assigned stops are not possible, which could lead to a better schedule when a new request is added.

Linear Assignment The idea of this algorithm proposed by SIMONETTO et al. [2019] is to reduce the number of candidate schedules created to obtain a linear assignment problem when assigning the schedules to vehicles. On the one hand, this drastically reduced the number of schedules to be created. On the other hand, linear assignment problems can be solved efficiently by specialized algorithms like the Hungarian algorithm. This can be achieved by providing each pair of new requests and vehicles with only one feasible candidate schedule for assignment. The assignment problem of Eq. 3.12a to 3.12e can then be reduced to

$$\text{minimize} \quad \sum_{v \in V} \sum_{r \in R_u} \Delta\rho_{r,v} \cdot z_{r,v} \tag{3.14a}$$

$$\text{s.t.} \quad \sum_{r \in R_u} z_{r,v} \leq 1 \qquad \forall v \in V \tag{3.14b}$$

$$\sum_{v \in V} \sum_{v \in V} z_{v,\gamma} \leq 1 \qquad \forall r \in R_u \tag{3.14c}$$

where $\Delta\rho_{r,v}$ is the change in the objective value of the candidate schedule for vehicle v to serve request r, and $z_{r,v} \in \{0, 1\}$ is the decision variable of assigning request r to vehicle v. Candidate schedules are created by solving the single vehicle DARP for each pair of requests and vehicles, while currently assigned requests have to be preserved in the schedule. Similar to ALONSO-MORA et al. [2017a], an exhaustive search is only applied for vehicles with less than five assigned requests, while the insertion heuristic is used for vehicles with more than five assigned requests to create the candidate schedules. Compared to the insertion heuristic, the linear assignment algorithm allows the processing of multiple requests simultaneously in batch, while the same number of request-vehicle combinations are explored. Nevertheless, this algorithm still does not allow requests re-assignment in later epochs. Additionally, requests in the same batch cannot be assigned to the same vehicle, which might lead to further losses.

Full Re-Build In this variant of the proposed algorithm, the V2RBs are not stored in between optimization epochs, and the schedule-building process is started from scratch in each optimization epoch. The goal is to evaluate the benefit of storing the V2RBs between optimization epochs.

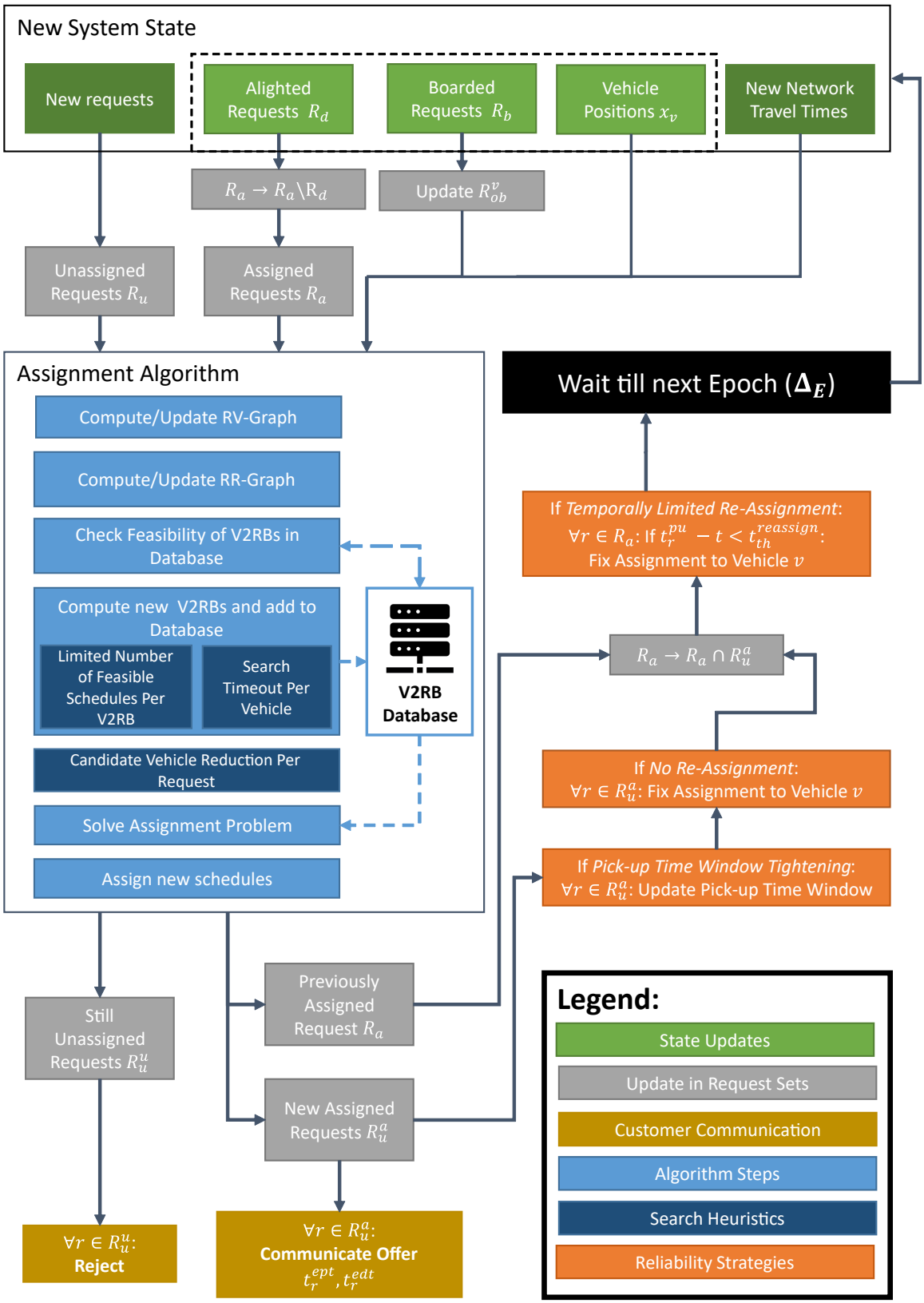


Figure 3.5: Flowchart for the solving the assignment problem.

3.3 Repositioning

The algorithms described in the last section can only assign schedules to vehicles if a trip request is made in the vicinity of the vehicle, which is constrained by the driving time corresponding to the maximum waiting time t_{max}^{wait} of a request. Therefore, vehicles can get stuck in regions of the operating area where demand is low, while further vehicle supply is needed in regions of high demand. Even if the maximum waiting time constraint would be relaxed and an assignment is possible, customers might have to wait a long time for service because of long approaching times of vehicles. To avoid being stuck in low-demand areas, vehicles can be actively repositioned (or rebalanced) to regions with under-supply to increase service availability and vehicle utilization.

Main features within the context of this section have been previously published in Engelhardt et al. [2023]. In the following, terms and definitions to model repositioning are described first. A sampling-based repositioning algorithm for ARP services is presented in the next step. Lastly, benchmark algorithms are described and used in the case study to evaluate the efficiency of the proposed algorithm.

3.3.1 Terms and Definitions

Determining repositioning trips usually requires three main steps:

1. A forecast of future demand. This demand is often aggregated on a zonal level within certain time intervals to estimate the future need for vehicles in specific zones.

2. A methodology to estimate the expected benefit of sending vehicles to a specific zone. This benefit can be measured in terms of expected profit, demand-supply imbalance, or similar metrics.

3. An algorithm to assign repositioning trips to idle vehicles for repositioning to specific zones.

In this thesis, the last two steps are mainly treated, while developing a model to forecast future demand is out of the scope of this thesis. Nevertheless, sensitivity analysis on the forecasted demand is conducted in the case study to evaluate the robustness of the proposed algorithm.

The ARP service is assumed to operate in a partitioned operating area with zones Z_R. Each node $n \in N$ in the network graph $G = (N, E)$ is assigned to precisely one zone $z \in Z_R$. Each zone $z \in Z_R$ is characterized by a centroid $c_z \in N$ that is used as a reference target for repositioning. In a real service, this centroid could be an operational hub or a place for parking where vehicles can wait for new assignments. It is assumed that there are no capacity limits for idle vehicles at these centroids. As the reference point for a zone, travel times and distances between zones are measured from the centroid of one zone to the centroid of the other zone.

Additionally, it is assumed that there is a forecast of future demand available that estimates the expected number of customers $\lambda_{i,j}^{T}$ requesting trips from zone $z_i \in Z_{FC}$ to zone $z_j \in Z_{FC}$ within a time window between $[T, T + \delta_T^{FC}]$. The demand forecast is therefore not

only aggregated spatially by zones Z_{FC} (that generally do not have to be the same as the repositioning zones Z_R), but also temporally by time intervals T with length δ_T^{FC}.

As discussed in section 3.1.1, the rebalancing algorithm is applied less frequently than the assignment algorithm in steps of Δ_R.

3.3.2 Sampling-based Repositioning

The proposed algorithm follows a sampling approach to address future vehicle imbalances and make informed decisions. By sampling artificial requests from the forecast distribution, the algorithm generates actual routes that accurately consider service design parameters (e.g., time constraints, objective function, or vehicle capacity). The goal is to mimic the behavior of the assignment algorithm under the assumption that the sampled requests will be realized in the future. The sampling approach can convert the demand forecast into a supply forecast by creating actual vehicle routes. On the one hand, it can estimate the number of customers that can be served by the same idle vehicle considering pooling and scheduling of customers. On the other hand, it also incorporates the capacity of currently en-route vehicles to accommodate future requests. As an output, the idle vehicles are sent towards the locations of the expected first pick-ups. The en-route vehicles remain following their original schedules.

The algorithm incorporates two additional features: 1) The algorithm can handle stochastic variation in the sampling by sampling multiple times from the forecast distribution. Repositioning trips are assigned to be beneficial in all sampled scenarios. 2) To overcome long-term supply-demand imbalances, the algorithm implements a multi-horizon approach, also considering possible repositioning trips in future epochs.

Figure 3.6 presents an overview of the rebalancing algorithm. Vehicles are separated into en-route vehicles currently serving customers and idle vehicles available for repositioning.

In the first step **(a)**, the algorithm takes as input only all currently en-route vehicles and their assigned schedule. These vehicles are used to estimate their ability to accommodate future requests, while currently idle vehicles are assigned in a later step to serve remaining future requests. In the sampling process **(b)**, future requests are drawn from the forecast distribution defined by $\lambda_{i,j}^T$ within a forecast horizon $\mathfrak{H}$, covering all temporal forecast bins $T \in \{t, t + \delta_T^{FC}, ..., t + \mathfrak{H}\}$. For each sample, future vehicle states are simulated to identify supply shortages. If en-route vehicles cannot accommodate a request, a new hypothetical vehicle is created at the zone centroid of the request's origin. The request is then assigned to this hypothetical vehicle, forming a new schedule for upcoming requests to be assigned to. Future supply shortages are therefore identified by these hypothetical vehicles. Each hypothetical vehicle represents the future demand for an actual idle vehicle that needs to be repositioned to the corresponding zone to serve future requests. To decrease the impact of stochastic variance, this step is repeated for N_S samples, indicated by different layers in Figure 3.6. Lastly, a zone-based assignment problem is formulated **(c)** that assigns idle vehicles to reposition to the zone of hypothetical vehicles **(d)**.

The sampling process and the assignment problem are described in detail in the following paragraphs.

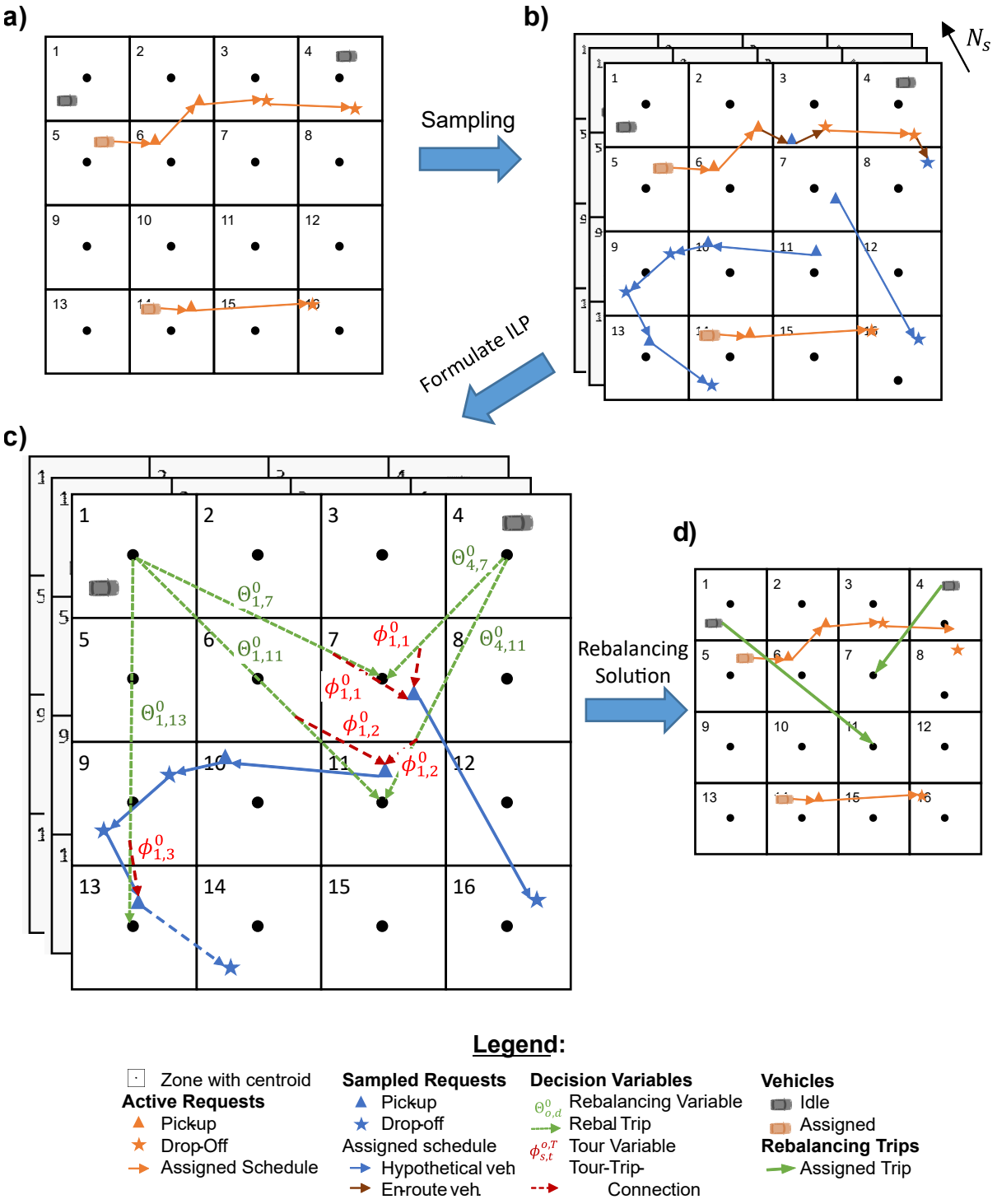


Figure 3.6: Sketch for solving the sampling-based repositioning problem for ride-pooling. a) Problem input. b) Assignment of sampled requests (N_S samples). c) Solving assignment problem. d) Assigning repositioning trips.

3.3.3 Sampling Future Fleet States

The algorithm to compute future vehicle states is sketched in Algorithm 2. Input to the algorithm are currently en-route vehicles with their assigned schedules as well as the forecast distribution parameterized by $\lambda_{i,j}^{T}$ with forecast horizon $\mathfrak{H}$. A Poisson process with rate $\lambda_{i,j}^{T}$ determines the number of trips requested from zone z_i to zone z_j within the temporal bin T. A random node from zone z_i and zone z_j is drawn as the origin and destination of the request, respectively. The request time is randomly chosen from the interval $[T, T + \delta_T^{FC}]$. N_S different request samples are created to reduce stochastic variance.

Fleet states are progressed into the future in time steps of Δ_E (the same time step as the duration between consecutive assignment epochs). In each time step, the assignment of new requests is treated at first. As the rebalancing time step Δ_R is generally smaller than the forecast horizon $\mathfrak{H}$, it is crucial that the request assignment is computationally efficient to maintain real-time applicability. Performing the previously described assignment algorithm can be computationally too costly to be applied in the rebalancing step. Therefore, the insertion heuristic formulated in Algorithm 1 is used to find feasible schedules for the request: The request is only inserted into the currently assigned schedule of each vehicle that can reach the origin of the request within t_{max}^{wait}. The resulting vehicle schedule that minimizes the objective of Equation 3.9 is assigned to the vehicle. If no solution is found, a new hypothetical vehicle is created at the zone centroid of the request origin and assigned to serve the request. After all sampled requests of the time step are assigned, vehicles move according to their assigned schedule.

Once all sampled requests are addressed, input parameters for the rebalancing formulation are constructed. For each hypothetical vehicle, the recorded schedule is concatenated with the remaining planned schedule of the vehicle to incorporate all sampled future requests. The start zone $o_{s,t}$ of each hypothetical vehicle marks a possible future supply shortage. The objective value $\rho_{s,t}$ of the created schedule is computed with Equation 3.9 and estimates the operator profit for providing an idle vehicle at this location. The starting time $\tau_{s,t}$ of the schedule estimates the latest arrival time of a vehicle in this zone to serve this schedule. Finding idle vehicles to reach the zone in time might not always be possible. In this case, none of the associated sampled requests would be served by a rebalancing vehicle. Instead, it might be beneficial to skip those sampled requests scheduled at the beginning of the hypothetical vehicle's schedule and serve at least those requests later in the schedule when more idle vehicles are available. Therefore, sub-schedules are defined for each hypothetical vehicle's schedule: At each stop, the algorithm checks whether the vehicle occupancy of the schedule would be zero. If this is the case, a new sub-schedule is created. Similarly, for each sub-schedule the parameters $o_{s,t}$, $\tau_{s,t}$ and $\rho_{s,t}$ are computed. An example of the sub-schedule of a hypothetical vehicle is sketched in Figure 3.6c). The dashed blue line indicates the sub-schedule of the superordinate schedule of the hypothetical vehicle. The vehicle in zone 1 can serve the superordinate schedule either by repositioning to zone 11 or zone 13. The decision to reposition to zone 13 would only serve a subset of the requests of the superordinate schedule.

Algorithm 2 Creating Future Schedules From Sampled Requests

Input: Assigned vehicles with current schedules, forecast distribution $\lambda_{i,j}^{T}$
Output: List of start_zone, start_time, objective, sub_tour_index, tour_index, sample

```
V_A ← Assigned vehicles with current schedules
V_R ← Initialize list of new rebalancing vehicles with schedules
T ← Initialize list of (start_zone, start_time, objective, sub_tour_index, tour_index, sample)
s ← 0 # Index of sample set
for N_S samples do
    request_sample ← Sample requests from λ^T_{i,j}
    for all time steps do
        for all sampled_requests in time step do
            best_schedule ← None
            for all vehicles with schedule in V_A + V_R do
                new_schedule ← insert(sampled_request, schedule)
                if objective(best_schedule) < objective(new_schedule) then
                    best_schedule ← new_schedule
                end if
            end for
            if best_schedule is not None then
                update schedule of corresponding vehicles
            else
                create new artificial vehicle at origin of request and add to V_R
            end if
        end for
        move vehicles in V_A + V_R according to assigned schedules
    end for
    u ← 0 # Index of tour set
    for all vehicles with schedule in V_R do
        t ← 0 # Index of sub-tour set
        for all stop in schedule with zero vehicle occupancy do
            sub_schedule ← remove preceding stops from schedule
            o_{s,t} ← start_zone(sub_schedule)
            τ_{s,t} ← start_time(sub_schedule)
            ρ_{s,t} ← objective(sub_schedule)
            add (o_{s,t}, τ_{s,t}, ρ_{s,t}, t, u, s) to T
            t ← t + 1
        end for
        u ← u + 1
    end for
    s ← s + 1
end for
```

3.3.4 Repositioning Trip Assignment

An ILP is formulated to assign rebalancing trips to idle vehicles to serve the sampled schedules. Idle vehicles are aggregated on a zonal level to decide for rebalancing trips between zone $o \in Z$ and $d \in Z_R$. As the forecast horizon $\mathfrak{H}$ is considered larger than the reposition period δ_T^{FC}, future rebalancing trips are also considered. The decision variable $\theta_{o,d}^0$ refers to immediate rebalancing actions that are performed after the problem is solved. $\tilde{\theta}_{o,d}^{T,s}$ on the other hand refers to potential future rebalancing trips in time step $T \in \{0, 1, 2, ..., T_{max} = \frac{\mathfrak{H}}{\delta_T^{FC}}\}$ in sample s. Depending on how the real system evolves, they might or might not be realized in a later epoch. Note that the immediate rebalancing decision variable $\theta_{o,d}^0$ is independent of the sample s as only one decision can be made, which should lead to a good performance across all possible realizations sampled. $\phi_{s,t}^T$ is the decision variable to assign sampled trips: It takes the value 1 if a rebalancing trip in time step T is assigned to trip t from sample s.

Before describing the optimization problem, the following sets are introduced, which are used to associate rebalancing decision variables ($\theta_{o,d}^0$, $\tilde{\theta}_{o,d}^{T,s}$) with feasible assignments of sampled schedules ($\phi_{s,t}^T$):

- $\mathbb{T}_s$: Set of all sub-tours from sample s
- $\mathbb{A}_{s,t}$: Set of all rebalancing decision periods eligible for assigning tour t in sample s.
- $\mathbb{U}_{\kappa,s}$: Set of all sub-tours referring to the same superordinate tour κ in sample s.
- $\mathbb{B}_{s,o,T}$: Set of all sub-tours in sample s that are reachable by rebalancing a vehicle from zone o in decision period T.
- $\mathbb{D}_{s,o,T}$: Set of sub-tours in sample s that terminate in zone o in decision period T. This set contains tuples of tour index and decision period for assignment of the corresponding decision variable ($\phi_{s,t}^T$).

The optimization problem is defined as follows:

Minimize:
$$\sum_{o,d \in Z_R} \left(c_{o,d} \theta^0_{o,d} + \frac{1}{N_S} \sum_{s=0}^{N_S} \sum_{T=1}^{T_{max}} \gamma^T c_{o,d} \tilde{\theta}^{T,s}_{o,d} \right) + \\ + \frac{1}{N_S} \sum_{s=0}^{N_S} \sum_{t \in \mathbb{T}_s} \sum_{T \in \mathbb{A}_{s,t}} \gamma^T \rho_{s,t} \phi^T_{s,t} \tag{3.15a}$$

s.t.:
$$\sum_{d \in Z_R} \theta^0_{o,d} \leq V^{idle}_o \qquad \forall o \in Z_R \tag{3.15b}$$

$$\sum_{d \in Z_R} \tilde{\theta}^{T,s}_{o,d} \leq \left(V^{idle}_o - \sum_{o \in Z_R} \theta^0_{o,d} \right) + \sum_{\tau=1}^{T-1} \left(\Delta V^{idle}_{\tau,s,o} + \sum_{(t,\nu) \in \mathbb{D}_{s,o,\tau}} \phi^\nu_{s,t} - \sum_{o \in Z} \tilde{\theta}^{\tau,s}_{o,d} \right) \\ \forall o \in Z_R, \forall s \in N_s, \forall T \neq 0 \tag{3.15c}$$

$$\theta^0_{o,d} = \sum_{s=0}^{N_S} \sum_{t \in \mathbb{B}_{s,o,0}} \phi^0_{s,t} \qquad \forall o, d \in Z_R, \forall s \tag{3.15d}$$

$$\tilde{\theta}^{T,s}_{o,d} = \sum_{t \in \mathbb{B}_{s,o,T}} \phi^T_{s,t} \qquad \forall o, d \in Z_R, \forall s, \forall T \in [1, ..., T_{max}] \tag{3.15e}$$

$$\sum_{t \in \mathbb{U}_{\kappa,s}} \sum_{T \in \mathbb{A}_{s,t}} \phi^T_{s,t} \leq 1 \qquad \forall s \in N_S, \forall \kappa \tag{3.15f}$$

$$\theta^0_{o,d}, \tilde{\theta}^{T,s}_{o,d} \in \mathbb{N}^+_0 \qquad \forall o, d \in Z_R, \forall T \in [1, ..., T_{max}], \forall s \in N_S \tag{3.15g}$$

$$\phi^T_{s,t} \in \{0, 1\} \qquad \forall T \in [1, ..., T_{max}], \forall s \in N_S, \forall t \tag{3.15h}$$

The first line of the objective in Equation 3.15a reflects the trade-off between costs and expected profit for repositioning. $c_{o,d} \geq 0$ are the costs (the travel time between the corresponding zone centroids). The factor $\gamma \in [0, 1]$ weights the costs for assigning future rebalancing trips in line with the Bellmann Equations (Equation 3.2). The first term in the first line considers immediate rebalancing decisions, while the second term considers future ones. The second line in the objective function reflects the expected profit from rebalancing trips. $\rho_{s,t} \leq 0$ is the objective value calculated in the sampling process for assigning trip t from sample s. Constraints 3.15b and constraints 3.15c limit the number of vehicles that can be rebalanced per zone $o \in Z_R$. While for immediate rebalancing trips in constraints 3.15b, only the number of currently idle vehicles per zone V^{idle}_o need to be considered, future rebalancing trips in constraints 3.15c also consider that vehicles already have been rebalanced out of the zone in a previous decision time steps, new vehicles with current assignments become idle ($\Delta V^{idle}_{\tau,s,o}$), or vehicles become idle after they finish their assigned tour after the rebal-

ancing trip. The constraints 3.15d and 3.15e relate rebalancing trips and the assignment of corresponding sampled sub-tours. Note that in constraint 3.15d the decision variable is not indexed by the sample s, i.e., immediate rebalancing trips can be assigned to multiple sub-tours, one per sample. With this constraint, efficient decisions for immediate rebalancing trips across all samples are made. In contrast, future rebalancing trips in constraint 3.15e differ for each sample. Constraint 3.15f ensures that each superordinate tour is assigned only once. Finally, constraints 3.15g and 3.15h define rebalancing trips and sub-tour assignment variables as integer and binary variables, respectively.

3.3.5 Benchmark Algorithms

Benchmark algorithms from the literature are introduced in this section to evaluate the performance of the proposed rebalancing algorithm.

No Repositioning

No rebalancing is applied.

Reactive Repositioning *(React)*

This algorithm is described in [Alonso-Mora et al., 2017a] and is based on an expected autocorrelation of demand. After each assignment step, the locations of unserved requests are tracked. Anticipating future demand at these locations, available idle vehicles are rebalanced to these locations by solving an assignment problem, minimizing the overall travel time. Alongside its simplicity, the advantage of this algorithm is that no forecast for future demand is necessary.

Queuing Theoretical Repositioning *(QT)*

This problem formulation uses queuing theoretical considerations to stabilize a Jackson network [Zhang and Pavone, 2016]. The assignment problem to be solved can be formulated as

$$\text{Minimize:} \quad \sum_{o,d \in Z_R} \tau_{o,d} \beta_{o,d} \tag{3.16a}$$

$$\text{s.t.:} \quad \sum_{d \neq o} (\beta_{o,d} - \beta_{d,o}) = -\mu_{QT} \sum_{d \neq o} (\lambda_{o,d} - \lambda_{d,o}) - (\sum_{d} \frac{I_d}{|Z_R|} - I_o) \qquad \forall d \in Z_R \tag{3.16b}$$

$$\beta_{o,d} \geq 0 \qquad \forall o, d \in Z_R \tag{3.16c}$$

$\beta_{o,d}$ is the (non-integer) decision variable to rebalance vehicles from o to d while $\tau_{o,d}$ is the interzonal travel time. The constraint of Equation 3.16b balances the expected in and out-flow of each zone. $\lambda_{o,d}$ are the expected number of trip requests between zones o and d within a forecast horizon $\mathfrak{H}_{QT}$. I_d are the number of idle vehicles in zone d. The last two terms try

to distribute the remaining idle vehicles evenly across zones. μ_{QT} is a demand scaling factor introduced in this study to consider sharing of trips.

To assign vehicles, the value $\beta_{o,d}$ is rounded to the next integer after solving the problem. Additionally, this formulation does not constrain the number of assigned vehicles to be smaller or equal to the number of idle vehicles. Therefore, for each origin zone idle vehicles are assigned randomly. The assignment terminates if no idle vehicle remains in a zone.

Horizon-based Repositioning *(Hor)*

This algorithm is proposed in [WALLAR et al., 2018] and considers the time when rebalancing vehicles arrive in their target zone. It is formulated as

$$\text{Maximize:} \quad \sum_{o,d \in Z_R} (\mathfrak{H}_{Hor} - \tau_{o,d}) \lambda_d \beta_{o,d} \tag{3.17a}$$

$$\text{s.t.:} \quad \sum_{d \in Z_R} \beta_{o,d} \leq I_o \qquad \forall o \in Z_R \tag{3.17b}$$

$$\beta_{o,d}(\mathfrak{H}_{Hor} - \tau_{o,d}) \geq 0 \qquad \forall o, d \in Z_R \tag{3.17c}$$

$$\sum_{o \in Z_R} \beta_{o,d}(1 - \frac{\tau_{o,d}}{\mathfrak{H}_{Hor}}) \leq \lambda_d \mu_{Hor} \qquad \forall d \in Z_R \tag{3.17d}$$

The objective (Equation 3.17a) is to maximize the number of requests each vehicle observes in its target zone given an expected request arrival rate λ_d in zone d within the forecast horizon $\mathfrak{H}_{Hor}$. With the interzonal travel time $\tau_{o,d}$, the objective incorporates the fraction of the forecast horizon that the vehicle is available in the target zone. Equation 3.17b constrains the number of vehicles that can be rebalanced, Equation 3.17c ensures that vehicles reach the rebalancing destination within the horizon. Finally, Equation 3.17d constrains the supply in target zones. The left-hand side computes the number of vehicles rebalancing to the zone weighted by the time they are available in this zone. The right-hand side estimates the expected demand for vehicles. μ_{Hor} is a scaling factor to specify an acceptable level of oversaturation.

3.3.6 Integration of Assignment and Repositioning

As both the assignment and repositioning algorithms adapt and assign new schedules to vehicles, it is crucial to define their interaction and how one of the algorithms can change assignments made by the other. As the repositioning algorithm only assigns trips to idle vehicles, no conflict arises in the trip assignment of the repositioning algorithm. However, the assignment algorithm might need to utilize the repositioning vehicles to serve new requests. As the hierarchy between assignment and repositioning (i.e., the trade-off between immediate and possible future reward) is not trivial, the following two approaches are evaluated:

1. **Assignment Priority *(No Lock)*:** In this approach, the assignment algorithm has the highest priority. In this case, repositioning vehicles are eligible to serve new requests. Therefore, assigned repositioning tasks are referred to as *not locked*. Technically, repositioning tasks are removed from the vehicles' schedules before the assignment algorithm is

executed. If the vehicle receives a new assignment to serve customers, the repositioning task will consequently not be executed. If no new assignment is made, the repositioning task will be added to the schedule again.

2. **Repositioning Priority *(Lock)*:** In this approach, the repositioning algorithm has the highest priority. Therefore, the repositioning task is *locked* in the vehicle's schedule and cannot be removed by the assignment algorithm. New request assignments can only be scheduled after arriving at the repositioning destination.

3.4 Reservation

While the first three sections of this chapter focused on the assignment and repositioning of vehicles, i.e., controlling the ARP vehicle fleet for a purely on-demand service, this chapter, which is based on the publication ENGELHARDT et al. [2022a], incorporates customers who reserve their trip in advance.

3.4.1 Terms and Definitions

With pre-bookings allowed, the customers requesting trips from the ride-pooling provider can be divided into two groups:

1. On-demand customers who request a service as soon as possible.

2. Pre-booking customers who request a trip at a specific pick-up time in the future.

Similar to the treatment of on-demand customers, pre-booking customers also expect a response to whether they can be served shortly after sending their request. If the operator accepts the trip request, the operator is bound to serve the customer. This is especially relevant for pre-booked trips, as the fleet control algorithm must ensure that these customers are scheduled despite the uncertainty regarding the fleet's status and the demand for on-demand trips at the time of the pre-booked service.

Pre-booking customers are characterized by an earliest pick-up time t_r^p, which does not coincide with the request time t_r. Compared to on-demand customers, similar time constraints apply also to pre-booking customers: 1) The pick-up time must be no earlier than the earliest pick-up time t_r^p. 2) The pick-up time must not exceed the latest pick-up time $t_r^l = t_r^p + t_{max}^{wait}$, with a maximum waiting time t_{max}^{wait}. 3) The maximum in-vehicle travel time tt_r^{max} must not exceed $tt_r^{direct}(1 + \Delta_{det})$; with the shortest possible travel time tt_r^{direct} for customer r to drive from origin to destination and Δ_{det} a detour factor. While generally the parameters t_{max}^{wait} and Δ_{det} could be selected differently for pre-booking customers and on-demand customers, in this thesis, the same parameters are used for both customer types to improve comparability of pre-booking and on-demand operation.

Problem Statement and Solution Approach

Depending on the reservation horizon, different solution approaches can be applied. For short- and mid-term reservations, the current state of the fleet is highly relevant for the decision of whether a pre-booking customer can be served or not. On the contrary, for long-term reservations, the current state of the fleet can hardly give an estimate of the available capacity to serve pre-booking customers. This thesis, therefore, elaborates on a multi-rolling horizon approach to classify reservation requests as well as the planning horizon of the online optimization (i.e., the assignment algorithm).

A central focus of this thesis is the evaluation of long-term reservations. Two main problems need to be addressed: 1) How can the operator decide whether a pre-booking customer can be served? 2) How can the operator ensure that pre-booking customers are served while

maintaining the service for on-demand customers? The solution approach is to create long-term schedules that are assigned to fleet vehicles. When new long-term requests are made, the operator can use these schedules to estimate the available capacity to serve the new requests. By defining waypoints, these long-term schedules are then used in the online optimization to ensure the service for pre-booking customers.

The remainder of this section is structured as follows: First, the multi-rolling horizon approach (i.e., the integration of assignment and reservation) is introduced to classify reservation requests and integrate the long-term schedules into the online optimization. Then, the methodology to create long-term schedules is described. In a next step, necessary adoptions to the repositioning algorithm are described to incorporate pre-booking customers. Finally, a benchmark method to treat reservation requests is described.

3.4.2 Integration of Assignment and Reservation

This section deals with the multi-horizon approach to integrate long-term schedules into the online optimization and to classify reservation requests.

In this thesis, two rolling horizons are introduced: 1) The "short-term horizon" T_h^{short}, and 2) the "revelation horizon" T_h^{rev}, with $T_h^{short} \leq T_h^{rev}$. An incoming reservation request r is classified as

1. *Short-term reservation request* if $t_r^p \leq t_s + T_h^{short}$,
2. *Mid-term reservation request* if $t_r^p \leq t_s + T_h^{rev}$ and $t_r^p > t_s + T_h^{short}$, and
3. *Long-term reservation request* if $t_r^p > t_s + T_h^{rev}$.

As the requested pick-up of short-term requests is shortly after the request time, no different treatment to on-demand requests is necessary. Therefore, short-term requests will be treated as on-demand requests and directly assigned (if possible) by the online assignment algorithm. On the contrary, the current fleet state hardly influences the assignment of long-term requests. Consequently, long-term requests are assigned by updating the long-term schedules, which will be described in the following section. The reservation horizon of mid-term requests, however, is short enough to be influenced by the current fleet state, i.e., many currently assigned schedules to serve on-demand requests are not terminated until the mid-term request pick-up time. Nevertheless, directly inserting mid-term requests into the current schedules would be computationally too costly[11]. Therefore, a simple insertion heuristic (Algorithm 1) is used to assign mid-term requests to vehicles. After a successful assignment, mid-term requests are marked as "revealed" requests. Revealed requests are considered in the online optimization but not eligible for re-assignment to other vehicles to reduce computational load. Only when their pick-up time is within the short-term horizon are they considered for re-assignment. More details are provided in the following when the incorporation of long-term schedules into the online optimization is described.

[11] As the latest pick-up time is high, a majority of vehicles would be able to serve the request, rendering the search strategy of the described algorithm inefficient.

Passing Long-Term Schedules to the Assignment Algorithm

The long-term module, which will be described in section 3.4.3, creates long-term schedules to serve all accepted long-term reservation requests. The online optimization (the algorithm presented in chapter 3.2.4 with adaptions that will be described in the following section) uses these schedules to assign vehicle schedules to serve pre-booked and incoming on-demand requests. This section discusses how the long-term solution is integrated into the online optimization process to ensure all accepted reservation requests are fulfilled within their time constraints while keeping computational time within acceptable limits.

A naive approach to solve this problem would be to assign the long-term schedules at the beginning of the simulation and insert on-demand requests into this solution when requested. This approach has been used in other studies (e.g., [Ma and Koutsopoulos, 2022; Wen et al., 2019; Dandl, 2022]), but does not allow for dynamic global optimization of the current fleet state (i.e., no reassignments of pre-booked requests would be possible). However, this approach would limit the solution space significantly, especially when the fraction of on-demand requests is much higher than pre-booked requests. In this setting, it can no longer be assumed that the long-term solution for pre-booked requests is good enough compared to the optimal solution for all revealed requests.

The approach in this study is to elaborate on the short-term horizon T_h^{short} and the revelation horizon T_h^{rev} that are sketched in Fig. 3.7. Fig. 3.7**(a)** shows the long-term schedule assigned to a given vehicle. The long-term solution dominates the length of the schedule and might cover scheduled stops for the whole book-ahead time (up to one day in the conducted case study). Nevertheless, to adapt the schedules for incoming on-demand requests, only scheduled stops in the foreseeable future are of relevance. Therefore, The idea is to use the two horizons T_h^{short} and T_h^{rev}. These horizons serve the purpose of revealing only relevant information to the online optimizer to reduce computational complexity and will be described in the following.

Let $\tilde{\psi}_k^{off}(v, R_\gamma)$ be the currently assigned plan of vehicle v (depicted in Fig. 3.7**(a)**), which schedules all stops, including far-ahead reservation requests. The revealed online schedule $\tilde{\psi}_k^{on}(v, R_\beta)$ includes only upcoming stops. Stops from $\tilde{\psi}_k^{off}(v, R_\gamma)$ are included in $\tilde{\psi}_k^{on}(v, R_\beta)$ until reaching the stop κ. κ refers to the first stop with planned arrival time after $t_s + T_h^{rev}$ while no passengers are on board the vehicle before this particular stop is executed. The latter condition ensures that the schedule $\tilde{\psi}_k^{on}(v, R_\beta)$ is a feasible schedule that delivers all scheduled customers. If it is ensured that the vehicle is still able to reach κ in time, all upcoming reservation requests can still be served punctually. Therefore, a quasi-stop (waypoint) is added to the schedule of $\tilde{\psi}_k^{on}(v, R_\beta)$: The vehicle is scheduled to arrive at the node of κ with the latest arrival constraint corresponding to the scheduled start time of κ. Additionally, no stop is allowed to be scheduled after this waypoint. A waypoint can, therefore, be interpreted as an end-constraint for the online optimization, which must be fulfilled if the decision is made to alter the online schedule to accommodate new on-demand requests. Fig. 3.7**(b)** sketches the revealed online schedule. The stops to pick up and drop off for r_1 and r_2 are revealed. Because the pick-up stop of r_3 is later than $t_s + T_h^{rev}$ (gray rectangle), it is added as a waypoint to the schedule.

The second horizon is the short-term horizon T_h^{short} with $T_h^{short} \leq T_h^{rev}$. All upcoming reservation requests $r \in R_\beta$ in the online schedule $\tilde{\psi}_k^{on}(v, R_\beta)$ are added to the set of online requests and will now be considered in the online optimization for possible re-assignment to

another vehicle. In Fig. 3.7**(b)**, the pick-up time of request r_1 (orange) is scheduled before $t_s + T_h^{rev}$ and is therefore considered in the online optimization. In the example of Fig. 3.7**(c)**, r_1 has been re-assigned to another vehicle v_2, while v received two new requests to serve that fit better into the schedule. Note that request r_2 cannot be re-assigned as its pick-up is not within the short-term horizon T_h^{short}. Nevertheless, the inserted waypoint is still served in time.

Lastly, as sketched in Fig. 3.7**(d)**, the whole schedule can be rebuilt to apply the formulation again in the following optimization time step.

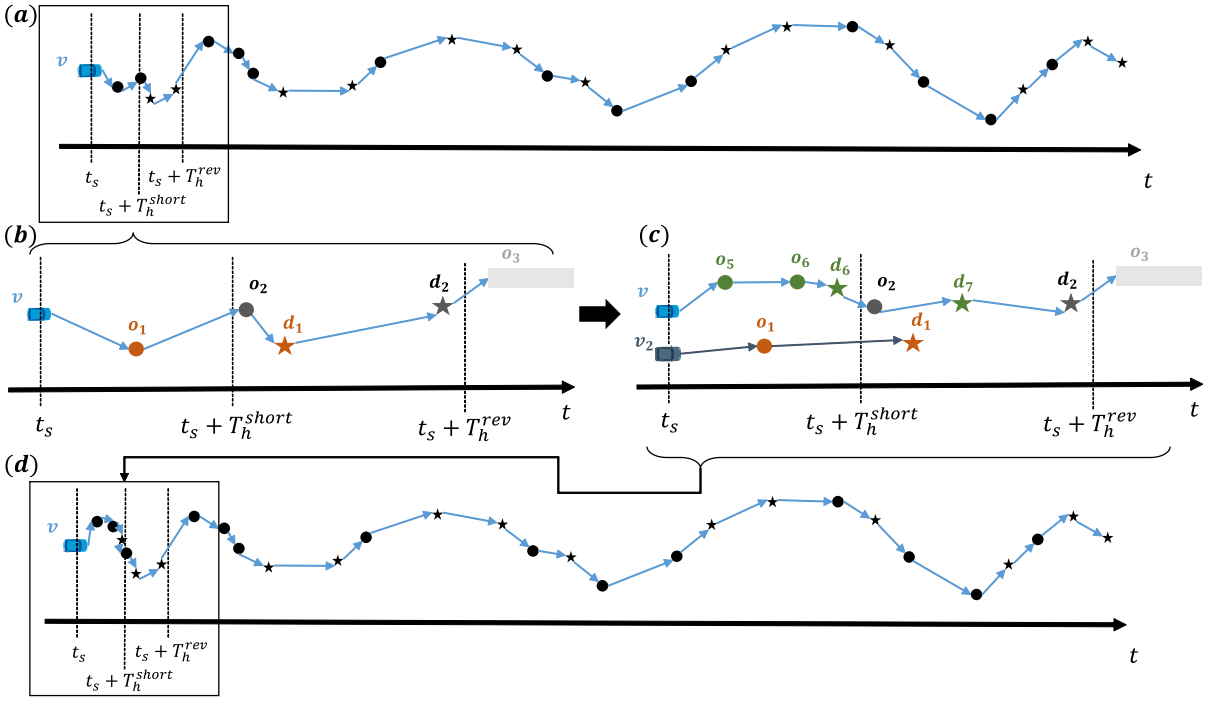


Figure 3.7: Sketch showing the effect of the two defined rolling horizons. (a) The complete assigned schedule including reservation requests for vehicle v at time t_s. (b) Schedule considered for online optimization: The schedule is revealed to the online optimizer until the first stop after $t_s + T_h^{rev}$ where no request is scheduled to be on-board of the vehicle. This stop (gray rectangle) is added to the online schedule as end-constraint (i.e. waypoint) of the schedule. The pick-up of r_1 (orange) at o_1 is scheduled before $t_s + T_h^{short}$. Therefore, it can be re-assigned within the online optimization. (c) After the online optimization, requests r_5 and r_6 (green) are assigned to vehicle v while request r_1 has been re-assigned. (d) As the end constraint ensures feasibility for the upcoming reservation schedule, the updated overall schedule can be recreated.

Adaptions to the Assignment Algorithm

Concerning incorporating reservation schedules, two minor adjustments are made to the assignment algorithm described in section 3.2.4.

Revisiting the formulation of the assignment algorithm, the first step involves creating feasible candidate vehicle schedules within the framework of V2RBs[12] for all vehicles and all currently active requests while the second step assigns the best schedules to the vehicles. Let R_{short} be reservation requests with an earliest pick-up time within T_h^{short} (orange circle and star in Fig. 3.7) and R_{rev} (the set of revealed requests) (the dark gray circle and star in Fig. 3.7) with an earliest pick-up time between T_h^{short} and T_h^{rev}. In the creation phase of V2RBs, requests in R_{short} are treated like incoming on-demand requests and all feasible V2RBs are actively created. This also includes V2RBs with vehicles other than the assigned one. For requests in R_{rev}, on the other hand, no re-assignment is allowed yet, and therefore they are not actively included in the creation phase. Similar to on-board requests, the order of their currently assigned pick-up and drop-off stops form the schedule of the V2RB with the lowest possible grade of the vehicle. New V2RBs can only be created by building upon this V2RB.

The second adaption is an additional feasibility constraint when constructing new schedules: If a waypoint for reservation requests is assigned, a schedule is only considered to be feasible if the corresponding waypoint (end-constraint), i.e. the start of a connected long-term schedule, is reachable in time (the light gray rectangle in Fig. 3.7).

When solving the assignment problem (Equations 3.12a-3.12e), all reservation requests are considered already assigned ensuring that their assignment is kept.

Adaption to Assignment Objective and Waypoint Re-Assignment

Revisiting the objective function of the assignment algorithm (Equation 3.9), a term is included in this function that evaluates the distance to travel for a given vehicle schedule. This may seem beneficial at first glance, but many waypoints are scheduled for far in the future, making the assignment of on-demand requests to guide the vehicle toward these waypoints unnecessary at the present moment and lead to suboptimal assignments of on-demand requests. Therefore, the objective function is adapted to not evaluate the distance to travel to waypoints.

Nevertheless, to reduce potential empty vehicle movements to reach the waypoints, a re-assignment of waypoints (and their corresponding long-term schedule) is conducted after the assignment of on-demand requests. Let $c_{v,w}$ be the cost of the vehicle v to reach the waypoint w, and V_w the set of vehicles that can reach the waypoint w in time after finishing their currently assigned schedule. If $|V_w| < |V|$, not all vehicles can reach the waypoint w in time. Therefore, the waypoint w is considered urgent, and its corresponding assignment cost are set to $c_{v,w} = d(\tilde{x}_v, x_w)$, with $\tilde{x}_v$ the position of vehicle v after finishing the currently assigned schedule and x_w the location of the waypoint. If $|V_w| = |V|$, all vehicles are able to reach the waypoint w in time. Therefore, the assignment cost is set to $c_{v,w} = 0.1 \cdot d(\tilde{x}_v, x_w)$, prioritizing the assignment of short trips to urgent waypoints, when the overall assignment cost is to be

[12] Or their schedules are updated from memory because they have been computed in previous epochs.

minimized:

$$\text{Minimize:} \quad \sum_{w \in W} \sum_{v \in V_w} c_{v,w} \phi_{v,w} \tag{3.18a}$$

$$\text{s.t.:} \quad \sum_{w \in W} \phi_{v,w} \leq 1 \qquad \forall v \in V \tag{3.18b}$$

$$\sum_{v \in V_w} \phi_{v,w} = 1 \qquad \forall w \in W \tag{3.18c}$$

$$\phi_{v,w} \in \{0,1\} \qquad \forall v \in V, \forall w \in W \tag{3.18d}$$

The decision variable $\phi_{v,w}$ is binary and indicates whether vehicle v is assigned to waypoint w. Constraints 3.18b ensure that each vehicle is assigned to at most one waypoint, while constraints 3.18c ensure that each waypoint remains assigned.

Vehicle Movement

If a new schedule is assigned to a vehicle, a vehicle should usually begin to fulfill this schedule immediately. However, in the presence of reservations, the planned arrival at a waypoint or a pick-up of a reservation request might be far in the future. In this case, an immediate approach might not be necessary and even lead to additional empty VKT if new assignments are made in the meantime. Therefore, before a vehicle starts its approach, the difference between the latest arrival time at a stop and the earliest possible arrival time. Only if this value falls below the parameter $T_{approach}$ the approach is started. Otherwise, the vehicle remains at its current location. $T_{approach}$ can be interpreted as a safety buffer for the vehicle to arrive at the stop in time.

Benchmark Algorithm

The main benefit of the presented approach in contrast to studies like [Ma and Koutsopoulos, 2022; Wen et al., 2019; Dandl, 2022] is that it allows for re-assignment of pre-booking customers to adapt the pre-computed schedules when on-demand customers emerge. To quantify the benefits achieved by re-assignment of pre-booking customers, the proposed approach is benchmarked against the Insertion Heuristic already defined in Algorithm 1. Applying this assignment algorithm, incoming on-demand requests are inserted into the currently assigned vehicle schedules. The order of stops (also of the long-term schedules) and the corresponding assigned pre-booking requests remain the same.

3.4.3 Creation of Long-Term Schedules

The key assumption for planning long-term schedules is that the planning horizon is so far in the future that the current state of the fleet (i.e., the current short-term assignments to serve on-demand customers) is nearly irrelevant. It is assumed that the current vehicle location, if idle, or the final stop of the current assignment, if en-route, is sufficient information to plan the long-term schedule.

Given a set of long-term reservation requests, the task is to assign vehicle schedules to serve these requests in the future. As all requests are known due to reservation, this problem can

be formulated as a static DARP. In theory, solution algorithms like those presented in the literature review (section 2.4.1) could be applied to solve this problem. However, as a solution has to be found for potentially thousands of reservation requests, the computational effort to solve the problem optimally is intractable, even with state-of-the-art algorithms. It is argued that the termination of the algorithm is not as time-critical as for the online optimization, but a runtime of several hours is not acceptable. Nevertheless, finding optimal solutions might not be necessary, as the final performed schedules likely differ from the long-term schedules because the online optimization will adapt these schedules to serve on-demand customers.

Therefore, heuristic methods suffice to create long-term schedules. These methods should be able to terminate within a maximum of a few minutes runtime, allowing for a fast response to new reservation requests while still providing good, feasible schedules for efficient long-term operation.

In the following, two heuristic methods are described to create long-term schedules for vehicles. The first method allows for continuously incorporating new reservation requests into long-term schedules. The second method can be used if long-term schedules have to be created only once (i.e., for day-ahead reservations) and will be used to evaluate long-term reservations in detail.

Consecutive Batch Optimization (CBO) Method

The goal of this method is to dynamically create and adapt long-term schedules for incoming long-term reservation requests. The basic idea is to batch unassigned long-term requests based on their earliest pick-up time and solve an assignment problem at the corresponding time, considering current long-term schedules and already assigned requests.

Let R^u_{long} be the set of unassigned long-term reservation requests. The requests are sorted by their earliest pick-up time to batch these requests. The first request r_0 is added to an empty batch, and the following requests are added until request r_i with $t^p_{r_i} > t^p_{r_0} + T^{short}_h$. In this case, r_i opens a new batch. Further, let $r_{l,b}$ be the last request in batch b. The associated batch time t_b is set to $t^p_{r_{l,b}} - T_{short}$ to give the assignment algorithm temporal flexibility in assigning the requests in the next step.

After batching the requests, the long-term schedules are adapted to accommodate reservation requests in the batches. Given the currently assigned long-term schedules of each vehicle, the state of each vehicle is progressed (i.e., planned movements and boarding processes are executed) until the batch time t_b. The resulting state of the vehicle is then used to solve the previously described assignment problem at the batch time, while all requests in the batch are considered on-demand requests. If a request cannot be assigned, vehicles are searched that could have repositioned (according to their long-term schedule) to serve the request in time. If no vehicle is found, the reservation request is rejected. Otherwise, the vehicle with the smallest repositioning distance is assigned to serve the request. The previously assigned long-term schedules are then updated to include the new assignment, and the next batch is processed.

Non-Causal Rolling Horizon (NCRH) Method

This method assumes that all reservation requests are known at the beginning of the planning horizon (i.e., reservations are made the day ahead). The method is based on a rolling horizon approach and uses the same procedure as simulating an ARP service for on-demand requests. Each reservation request is treated as an on-demand request, and the simulation is run to create feasible vehicle schedules. Within the simulation, for each request r, the request time is set to $t_r^p - T_h^{short}$, i.e., it is revealed at the short-term horizon. At the end of the simulation, the vehicle schedules performed over the whole rolling horizon simulation period are used as long-term schedules.

For repositioning, a different methodology is applied compared to the algorithm described in section 3.3. As the goal is to produce feasible vehicle schedules, the repositioning strategy does not have to be causal. Therefore, the time when vehicles became idle is tracked. Each time no schedule is found to serve an on-demand customer, a post-processing step scans idle vehicles that would have been able to serve the customers if they started their approach in time. The vehicle with minimal travel time is assigned to serve the customer. In contrast to causal repositioning, this method does not rely on inherently unreliable forecasts, reducing unnecessary empty vehicle trips.

Due to its similarity to the simulation of a purely on-demand service, this method allows the evaluation of the impact of the solution quality of the long-term schedules on the overall system performance. In contrast to the *CBO* method, one can expect that the *NCRH* method always produces overall better schedules than the simulation of an on-demand-only service because the rolling horizon allows a look-ahead of upcoming requests. For the *CBO* method, an improvement in the overall system performance compared to an on-demand-only service cannot be guaranteed.

3.4.4 Integration of Repositioning and Reservation

When considering repositioning within the proposed framework for treating reservations, some adaptions have to be made. As most vehicles might have an assigned long-term stop at some point in the future, the definition of idle vehicles must be adapted. Additionally, the location of scheduled reservation stops has to be considered when repositioning vehicles, because in the extreme case, it might not be reasonable to send a vehicle to one side of the city while the next reservation stop is on the other side.

The following adaptions are made to include the reservation in the sampling-based repositioning algorithm described in section 3.3:

Definition of Idle Vehicles A vehicle is considered idle (and therefore available for repositioning) if it has no assigned stops in the future or the next stop is a waypoint with a planned start time outside the forecast horizon $\mathfrak{H}$ of the repositioning algorithm.

Reservation Schedules for Sampling Future Fleet States If a long-term schedule is assigned to a vehicle, its plan is revealed to the repositioning algorithm until the first stop

after $t+\mathfrak{H}+T_h^{rev}$[13] when no passengers are on board the vehicle, given the current time t. The same methodology is applied here as when revealing the long-term schedule to the assignment algorithm. If the corresponding vehicle is not idle, sampled future requests can be inserted into its schedule, similar to how a currently on-route vehicle is handled in the repositioning algorithm without reservations. If the vehicle is considered idle (the scheduled reservation waypoint is far in the future), the long-term schedule is detached from its currently assigned vehicle in the sampling process. During the process of simulating future fleet states, these schedules become also available for assignment. Once the progressed time in the sampling process approaches $\tau_k^{sat} - T_h^{repo}$, the long-term schedule k with scheduled arrival time τ_k^{sat} is assigned to the nearest available hypothetical vehicle (vehicles that are created during the sampling process to indicate supply shortages). If no vehicle is available, a new hypothetical vehicle is created. The idea behind this approach is that long-term schedules can be re-assigned to other vehicles based on potential repositioning trips if future sampled requests can be served before the next reservation waypoint is due.

Adaption to Repositioning Assignment Problem To maintain a feasible solution to the long-term schedules, the repositioning assignment algorithm (i.e., the ILP formulated in Equations 3.15) has to ensure that each long-term schedule is assigned to a vehicle. Let $\mathbb{K}_s^{odm}$ and $\mathbb{K}_s^{res}$ be the sets of hypothetical vehicle schedules that do not and do include long-term schedules for reservation requests, respectively. The constraint to assign maximally one long-term schedule to a vehicle for repositioning (Constraint 3.15f) can be replaced by the following two constraints:

$$\sum_{t \in \mathbb{U}_{\kappa,s}} \sum_{T \in \mathbb{A}_{s,t}} \phi_{s,t}^T \leq 1 \qquad \forall s, \forall \mathbb{K}_s^{odm} \tag{3.19a}$$

$$\sum_{t \in \mathbb{U}_{\kappa,s}} \sum_{T \in \mathbb{A}_{s,t}} \phi_{s,t}^T = 1 \qquad \forall s, \forall \mathbb{K}_s^{res} \tag{3.19b}$$

Constraint 3.19b ensures that all long-term schedules are assigned to a vehicle.

Assigning Repositioning Trips and Long-Term Schedules The solution of ILP 3.15 with constraints 3.19 not only defines repositioning trips but also (re-)assigns long-term schedules to vehicles. By evaluating the solution, each assignment of rebalancing trips and hypothetical vehicle sub-schedules can be retraced to an idle vehicle within a specific zone for each sample s. Three possible outcomes change the assignments of vehicles:

1. Only a repositioning trip is assigned to a vehicle: Similar to the on-demand-only case, the vehicle receives a repositioning assignment to the corresponding zone.

2. A sub-schedule from $\mathbb{K}_s^{res}$ is assigned, that only contains stops from the long-term schedule. In this case, the corresponding waypoint (i.e., long-term schedule) is assigned to the vehicle.

[13] As the repositioning algorithm progresses the state of the vehicle by $\mathfrak{H}$, the waypoint has to be at least $\mathfrak{H} + T_h^{rev}$ ahead to mimic the behavior of the assignment algorithm.

3. A sub-schedule from $\mathbb{K}_s^{res}$ is assigned that serves sampled requests before serving stops from the long-term schedule. Therefore, a vehicle is assigned a repositioning trip and a long-term schedule. In this case, the vehicle's schedule is extended by two stops: First, the repositioning trip, and second, the waypoint of the assigned long-term schedule. This situation can occur when the waypoint is scheduled far in the future, allowing the vehicle to be repositioned to a zone where it can serve other on-demand requests in the interim.

It should be noted that the assignment of reservation waypoints can differ for each sample s if, for example, a waypoint fits a repositioning trip in one sample but not the other. This issue usually occurs for waypoints that are scheduled far in the future, leaving time for re-assignment in the following repositioning epochs before they become urgent. To maintain a feasible assignment, waypoints from the first sample are assigned.

Benchmark Method

The repositioning method with reservations is compared to the Reactive Repositioning *React* described in section 3.3.5. In this method, when an on-demand request cannot be served, its origin is marked as a repositioning target. Without reservations, idle vehicles are assigned to reposition to these targets by minimizing the overall travel time. Similar to the method described above, vehicles with reservation waypoints are considered idle if the waypoint is at least T_h^{repo} ahead. In this case, the cost for a repositioning trip is updated to $tt_{eff}(x_v, x_r) = tt(x_v, x_r) + tt(x_r, x_b) - tt(x_v, x_b)$, with x_v the position of the vehicle, x_r the position of the repositioning target and x_b the position of the waypoint. This formulation also considers the needed trip to the waypoint after repositioning. A repositioning trip is only allowed if the vehicle can reach the waypoint in time.

3.4.5 Alternative Treatment of Pre-Booking Requests

An alternative, more straightforward approach to treat pre-booking requests is to handle them similarly to on-demand requests but prioritize their assignment. In this approach, no long-term schedule is created. Consequently, since no estimation of free capacity to accommodate reservation requests is available, all requests are initially accepted when requesting a pre-booked trip. The assignment and repositioning algorithm is adapted to prioritize the assignment of pre-booking requests to commit to the initial acceptance. Nevertheless, an assignment when the planned pick-up is approaching might still not be feasible within this approach. In this case, a late rejection is communicated by the operator[14].

Assignment

When the current time reaches $t_r^e - T_h^{short}$, i.e., the short term horizon T_h^{short} before the earliest pick-up time, the pre-booked request is revealed to the assignment algorithm. The request is

[14] Another approach could involve updating time constraints for pick-up and at least offering a delayed service in real-time operations. However, to compare this method with the presented approach that uses long-term scheduling to ensure feasibility, the KPI of late rejections is used as a measure of the approach's success.

added to the set R_u to find a feasible assignment in the next optimization epoch. To prioritize the assignment of pre-booking requests, the objective function of the assignment algorithm (Equation 3.4 and 3.9) is adapted to

$$\rho_{served}(\psi) = - \sum_{r \in R_\psi^{odm}} p_r^{odm} - \sum_{r \in R_\psi^{res}} p_r^{res} \ , \tag{3.20}$$

with an assignment reward p_r^{odm} for on-demand requests R_ψ^{odm} and a higher reward $p_r^{res} > p_r^{odm}$ for pre-booking requests R_ψ^{res}.

Repositioning

To further increase the chance of serving pre-booked requests, the repositioning algorithm can also be updated to proactively send vehicles to areas where pre-booking requests are scheduled. When using the sampling-based repositioning approach, unassigned pre-booking requests are added to the set of sampled requests. By the updated objective function (Equation 3.20), also the repositioning algorithm prioritizes rebalancing trips that bring vehicles closer to the pre-booking requests.

Chapter 4

Simulation Framework

This chapter describes the simulation framework used to evaluate the proposed methods. First, the simulation environment and details of the implementation are described. Then, the three case studies (Chicago, Munich, and Manhattan) are introduced, including the data sets used for the simulations. Finally, the KPIs used to evaluate the methods are defined.

4.1 Simulation and Implementation

As part of this thesis, the simulation framework FleetPy [Engelhardt et al., 2022b] has been developed in cooperation with colleagues, which is tailored to study MoD services and, therefore, answer the research questions of this thesis. FleetPy is an open-source agent-based simulation framework [TUM-VT, 2022]. The framework is written modularly to allow the user to enable modules of interest for specific studies. Most modules are implemented in Python and can be extended by the user. Additionally, input data formats are pre-defined to allow users to easily integrate their data and compare different case studies. The following section provides a high-level description of FleetPy, and relevant modules for this study are described in more detail.

4.1.1 Modules

The main modules of FleetPy and their interrelation are shown in Fig. 4.1. The heart of the framework is the *FleetSimulation*-Class. It has two main tasks: 1) It is responsible for loading input data, e.g., configuration files and input data paths, initializing all modules, and specifying file paths for simulation outputs. 2) The *FleetSimulation*-Class defines the simulation flow and thereby controls the central simulation time and the interaction of all other modules via time-based and event-based triggers.

Demand Modules

Green blocks in Figure 4.1 refer to demand-related modules. The *Traveler*-Class represents a single traveler trip. It is mainly characterized by its origin, destination, and the request time attributes. For the reservation use case, an earliest trip time attribute can be defined. Additionally, mode choice and dynamic behavior can be implemented. Travelers are initialized and collected in the *Demand*-Class. This class is responsible for loading demand data from input files and creating *Traveler*-Objects. Additionally, this class manages temporal behavior,

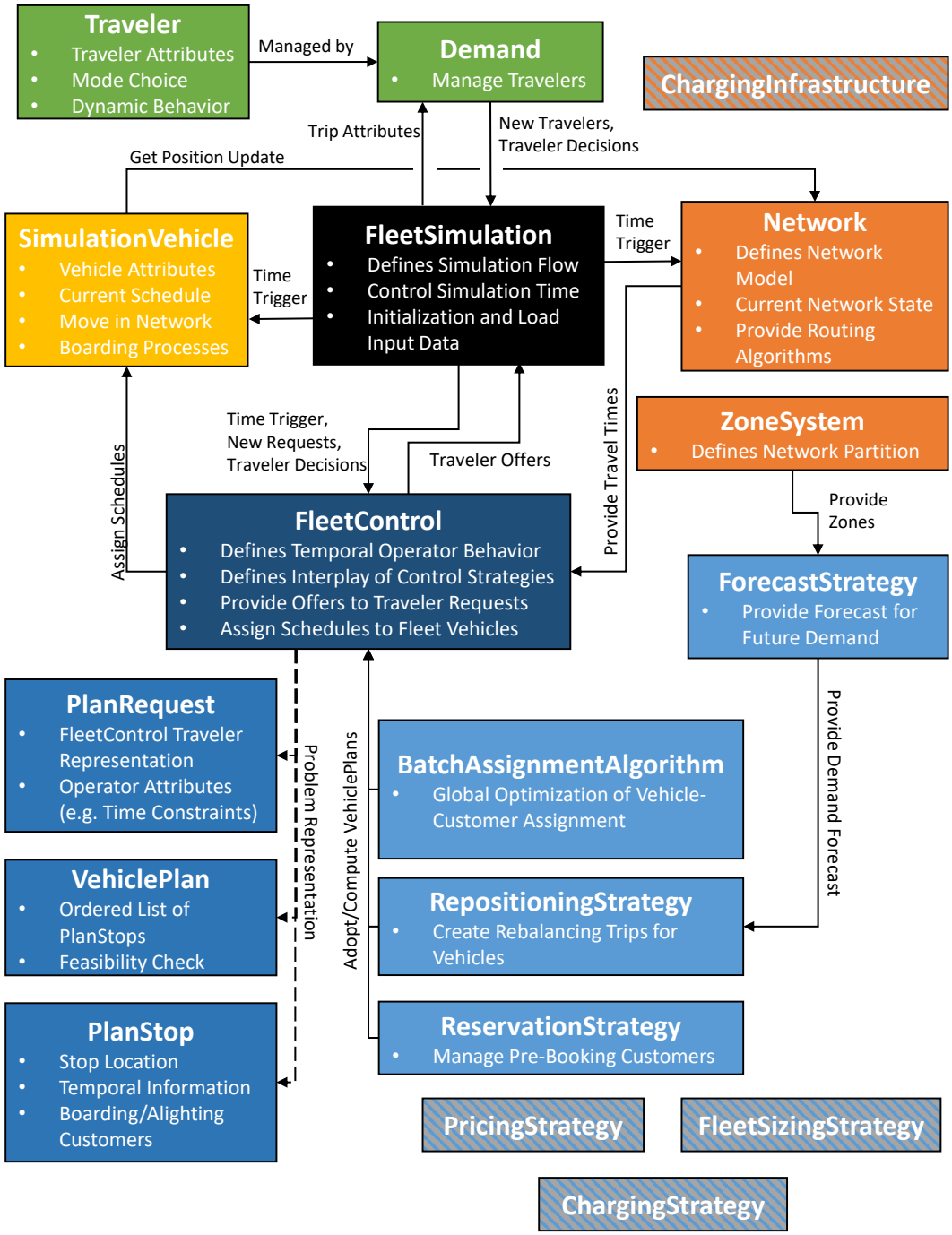


Figure 4.1: Sketch showing on a high level the main classes, their tasks in the simulation and their connection. Colors indicate simulation flow (black), infrastructure (orange), demand (green), vehicles (yellow), and fleet control classes (blue). Shaded blocks refer to modules implemented in FleetPy but are not treated in this thesis.

i.e., travelers request trips from available modes (mainly, but not necessarily only the MoD-

operator(s)), forwarding the corresponding trip offers, and triggering interface functions like mode choice evaluations. This class tracks the experience of each traveler during the simulation and writes them to an output file.

Implemented Traveler Model In the context of this thesis, the *Traveler*-Class implements impatient travelers. Impatient travelers are travelers who are not willing to wait for a trip offer for an indefinite amount of time. They request a trip from the ARP-operator and expect a trip offer as soon as possible, which is implemented as the time the next optimization epoch calculation finishes. Travelers only accept trip offers fulfilling their maximum waiting time and maximum travel time constraint. At the same time, they are modeled insensitive to the fare[1]. This thesis assumes that the operator is aware of each traveler's behavior and can optimize the trip offers accordingly, i.e., the operator can apply these time constraints when creating feasible schedules for vehicle assignments. If a traveler does not receive an offer fulfilling the mentioned attributes, the traveler cancels the request and leaves the system, assuming the traveler uses an alternative mode of transport. While the above-mentioned behavior is similar for travelers requesting on-demand and pre-booking trips, the latter also communicates a desired earliest trip time (pre-booking time) to the operator.

Infrastructure Modules

Orange blocks refer to infrastructure-related modules. For this thesis, mainly the *Network*-Class and the *ZoneSystem*-Class are relevant. The *Network*-Class is responsible for loading and storing the road network and its attributes. The class provides routing functionalities and is responsible for vehicle movement. Depending on the implementation, dynamic or stochastic behavior can also be included to model congestion by updating attributes of the corresponding edges. The *ZoneSystem*-Class defines the aggregation of the network into zones to allow a coarser spatial network analysis, e.g., for aggregation of demand forecast, which cannot be provided accurately on a node level.

Network Implementation This thesis implements the *Network*-Class as a directed graph $G = (N, E)$, with nodes $n \in N$ and edges $e \in E$. Edges are associated with a distance and travel time attribute. The travel time attribute is used to calculate the fastest routes for vehicles and travelers. Travel times are assumed to be deterministic, i.e., each vehicle takes the same time to travel along an edge. The travel time attributes can be updated at specific simulation time steps to account for congestion but remain constant and deterministic during the resulting time slices. As the ride-pooling assignment problem requires evaluating a vast number of routing queries, efficient routing algorithms are crucial for the performance of the simulation. The following implementations are available to provide fast routing queries:

- If the network is sufficiently small (up to around 5k nodes depending on available RAM), a lookup table can store the travel times and distances between all nodes.

[1] This assumes that travelers already have an experience of fares charged and request a trip merely to check for service availability.

- If the network is too large for a complete lookup table, a lookup table is still provided for a subset of nodes that are frequently used in routing queries (e.g., access nodes where customers can board or alight a vehicle).
- For remaining routing queries, the bidirectional Dijkstra algorithm is used to calculate the fastest route between two nodes.
- For one-to-many or many-to-one routing queries (i.e., finding available vehicles in the vicinity of a customer), the classic Dijkstra algorithm is used.
- Already computed routes are stored in a cache to avoid redundant routing queries.
- Dijkstra's algorithms are implemented in C++, which provides an additional speed-up of roughly 30x compared to a Python implementation [ENGELHARDT et al., 2022b].

Vehicle

The *SimulationVehicle*-Class (yellow) represents a vehicle in the simulation. It is mainly characterized by the operator it is associated with, its capacity, and its range (if charging or fueling processes are considered). Dynamic attributes include its current position, remaining range, onboard travelers, and currently assigned schedule. The assigned schedule defines a list of legs to be performed. These legs include routing tasks, where the vehicle is routed from its current position to the next destination on the fastest route in the network. For evaluation purposes, routing legs are subdivided into the states „route“, „repositioning“, and „to_reservation“. The „route“-state reflects the approach to the next boarding task. The „repositioning“-state is used when the vehicle repositions to a new location, while the „to_reservation“-state is used when the vehicle drives to a location where a traveler reserved a trip further in the future. Static legs refer to tasks where the vehicle remains at its location. These states of these legs include „boarding“(i.e., waiting for a boarding process to finish) and „waiting“(i.e., waiting for a traveler to arrive at the vehicle). If no task is scheduled, the vehicle is in the „idle“state (i.e., waiting for a new assignment).

The leg specifies the tasks to be performed (e.g., a list of boarding and alighting travelers) as well as the temporal information of the task, like its duration or earliest start time. If the vehicle state is triggered to be updated for a certain time step by the *FleetSimulation*-Class, the vehicle updates its state according to the current leg and starts the following one if applicable. The progress along the route is calculated in the *Network*-Class. The *Network*-Class calculates the new position of the vehicle after moving it in the corresponding time step. Within the implementation of this thesis, the vehicle progresses along the fastest route with currently set deterministic travel times.

Fleet Control

Finally, different shades of blue in Figure 4.1 indicate fleet control classes, the framework's core. The *FleetControl*-Class models a fleet operator in the simulation. It replicates the decision-making process of the operator and its interaction with the environment, e.g., the demand, the vehicles, and the infrastructure. The *FleetControl*-Class is responsible for the assignment of

schedules to its fleet vehicles to serve incoming demand, i.e., solve the underlying ride-pooling problem. It has access to multiple submodules (colored in bright blue) that subdivide the problem into smaller sub-problems.

The *BatchAssignmentAlgorithm*-module globally optimizes the current customer-to-vehicles assignments and their respective schedules. It implements the modules described in chapter 3.2.

The task of the *RepositioningStrategy*-module is to redistribute idle vehicles in the operating area to meet future demand. It has access to a *ForecastStrategy*-module that provides estimates for future demand. Implementation yields the modules described in chapter 3.3.

The *ReservationStrategy*-module treats pre-booking customers. The corresponding implementation is described in chapter 3.4.

FleetPy also implements modules for pricing (dynamically setting the fares), charging (managing charging processes), and fleet-sizing (adopting the active fleet size, e.g., due to driver shifts). However, as those modules are irrelevant to this thesis, further description is omitted.

Problem Representation

Darker blue objects in Fig. 4.1 refer to classes representing the ride-pooling problem. The *PlanRequest*-Class represents the operator's information about a traveler request. It collects request attributes like origin, destination, request time, and earliest trip time. Additionally, it stores service design parameters related to the request, like associated time constraints. In contrast to the *Traveler*-Class (although not applicable in this thesis), the *PlanRequest* may not have all information about the traveler, e.g., internal preferences or mode choice behavior. The *PlanStop*- and *VehiclePlan*-Class represent a possible planned schedule for a vehicle. The *PlanStop*-Class represents a single stop of a vehicle schedule. It is characterized by its location, task at the location (e.g. boarding and alighting of customers), and related time constraints (e.g., earliest and latest arrival time or duration at the stop).

The *VehiclePlan*-Class consists of an ordered list of *PlanStop*-Objects, which can be assigned to fleet vehicles to perform the planned tasks. By assigning a *VehiclePlan*, a list of route legs for the *SimulationVehicle* is created. Note that the *VehiclePlan*-Class only refers to the planning state of a schedule. The actual performed schedule might still differ due to incomplete operator knowledge, e.g., traffic congestion. *VehiclePlan*-Class additionally implements feasibility checks (i.e. whether constraints are fulfilled) and calculation of the objective function value of a schedule.

4.1.2 Simulation Flow

The simulation begins by initializing the *FleetSimulation*-Class. The *FleetSimulation*-Class loads input data, initializes all modules, and specifies file paths for simulation outputs. Network and zones are loaded from input files. The *Demand*-Class initializes travelers by reading corresponding start and end nodes and request times from a Comma-Separated Values (CSV)-file. Next, vehicles are initialized. In this thesis, the initial position of vehicles is chosen randomly from the set of access nodes.

The simulation flow is depicted in Fig. 4.2. The simulation time is divided into time steps Δ_S. The high-level interaction between the modules in each time step is shown in Fig. 4.2a.

First, vehicles move in the network according to their assigned schedules, and customers board or alight them. Then, the network travel times are updated if new travel times are available. The *Demand*-Class reveals new traveler requests and sends them to the *FleetControl*-Class, which decides if the request is treated as an on-demand or pre-booking request.

The next step triggers the *FleetControl*-Class for a new optimization epoch. The single steps of the optimization epoch are shown in Fig. 4.2b and trigger computations described in the methodology chapter 3. It is checked if upcoming reservations are due, and if so, these requests are revealed to the online optimization algorithm and treated as on-demand requests. Then, the *BatchAssignmentAlgorithm*-module is triggered to solve the ride-pooling assignment problem for all revealed requests and assign new schedules to the vehicles. Next, it is decided whether reservation plans are re-optimized and whether new pre-booking customers must be considered. Finally, the *RepositioningStrategy*-module is triggered to reposition idle vehicles to meet future demand.

After all travelers in this time step have been treated in batch by the *FleetControl*-Class, currently assigned schedules are evaluated to create trip offers to each traveler that has not been assigned yet, if available. Travelers decide if they book or cancel the trip and forward their decision to the operator.

4.1.3 Input Data

The input for the simulation consists of the following parts:

1. **Scenario Configuration:** The scenario configuration file(s) specify the simulation settings, which includes constant parameters (e.g., see Table 4.2), the modules to be used, and the path for input data files.

2. **Network:** Network data consists of CSV-files for nodes, specifying IDs and coordinates, and edges, specifying the start and end node of an edge, the distance, and the travel time. For dynamic network behavior, additional files provide either travel times for each edge or travel time scaling factors for each given simulation time interval. Additionally, preprocessed look-up tables can be provided to speed up routing queries.

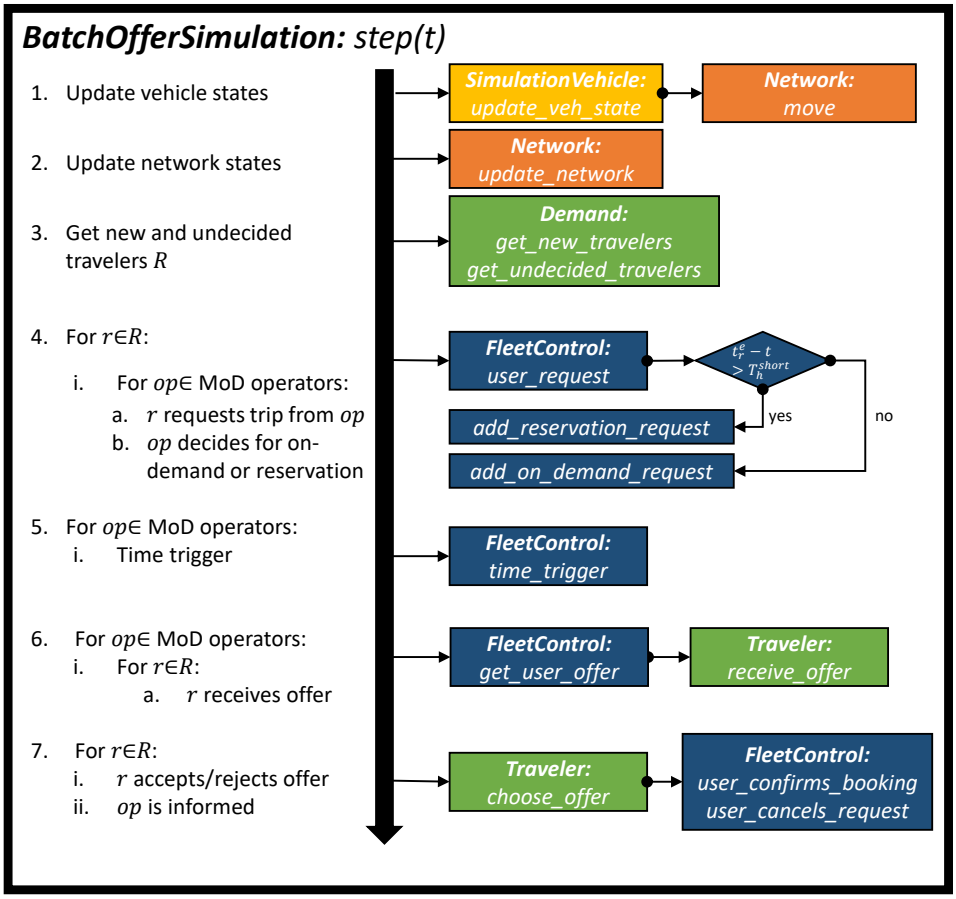
3. **Demand:** Demand data consists of CSV-files for traveler requests, specifying the origin node, destination node, request time, and earliest trip time.

4. **Zone:** Zone data consists of CSV-files for zones, specifying the assigned zone ID for each network node.

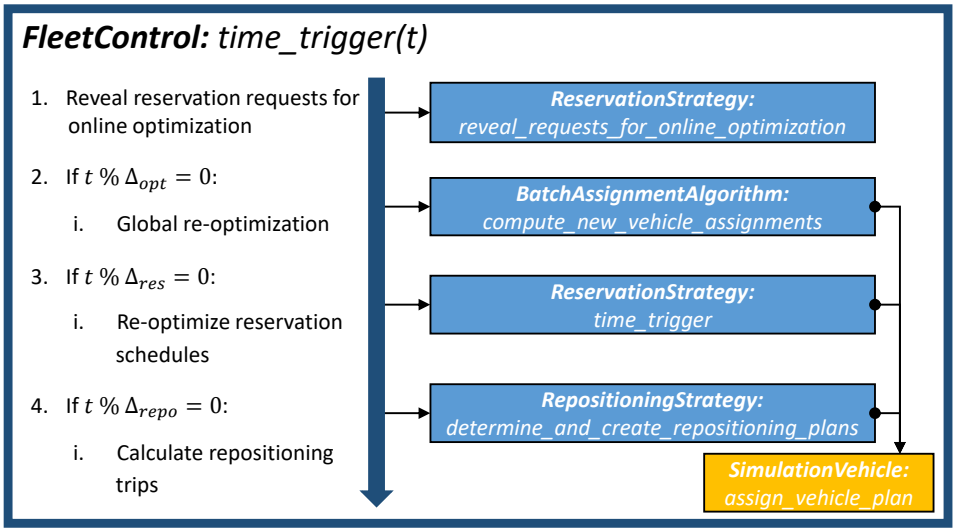
4.1.4 Output Data

The output of each simulation consists mainly of the following CSV-files:

1. **Travelers:** This file contains each traveler's experience during the simulation. Besides the request attributes, the file contains the assigned vehicle, boarding and alighting times (if served), and the initial offer created by the ARP-operator.



(a) Functionality of the time step method for the applied simulation flow.



(b) Functionality of the time trigger in the fleet control module.

Figure 4.2: Flow functionality of two main components in FleetPy and their high-level interaction with involved modules.

2. **Operator:** This file contains each completed leg of each vehicle during the simulation. Besides the vehicle ID, the leg state (e.g., boarding, repositioning, or route), the start and end node, the start and end time, the driven distance, and boarding, alighting, and on-board travelers are stored.

3. **Dynamics:** This file contains indicators tracked within each simulation time step. Mainly, computational time for relevant tasks is stored for later evaluation.

This data allows for a detailed spatio-temporal analysis of each simulated scenario. The main KPIs used to evaluate the simulation results are defined in section 4.3.2.

4.2 Case Studies

Three case studies are considered to evaluate the proposed methods in different settings: Chicago, Munich, and Manhattan. In the following, the data sets and resulting input data for the simulations are described for the three case studies and compared.

4.2.1 Chicago, Illinois

The street network for Chicago is extracted from OpenStreetMap (OSM) using the Python *OSMnx* package [Boeing, 2017]. To reduce the size of the network, edges labeled as "residential" or "living_street" are removed from the network, resulting in $12,585$ nodes with $27,446$ edges. Customers can only start and end their trip at specific access nodes. Similar to [Dandl et al., 2020b], boarding is prohibited on major roads like highways. Therefore, all nodes with adjacent edges not labeled as "primary", "secondary", "tertiary", or unlabeled edges are not considered access nodes. Due to the size of the network, the set of access nodes is further reduced by randomly removing access nodes if another access node can be found within a distance of 300m. This procedure is repeated until $4,000$ access nodes are left, resulting in a number small enough to preprocess travel time lookup tables between those nodes to reduce the computational time needed for routing queries. Figure 4.3a shows the resulting network with all access nodes.

Demand for the ARP service is created using the publicly available TNC data set for Chicago, Illinois [Chicago Department of Business Affairs & Consumer Protection, 2022]. This thesis uses TNC trips for the randomly chosen Tuesday 06/07/2022. Trips that start or end outside the Chicago city boundary are removed. Additionally, presumably, faulty data entries and round trips are removed, characterized by a trip distance larger than 100km or lower than 0.1km, a trip time larger than 5hours or lower than 60seconds, and an average speed higher than 130km/h or lower than 5km/h. After the filtering process, $12,7528$ trips remain. Requests are created by choosing a random access node for origin and destination within the reported pick-up and drop-off area. As request time, a random value in second steps is drawn from the reported 15min start time interval of the trip.

To calibrate network travel times, the reported trip duration in the data set is compared to the travel time of the fastest path when considering the maximum allowed speed from the OSM data on each edge. For each hour of the day, a scaling factor is calculated for the fastest

path travel times to resemble the reported travel times of all trips in the data set in the same hour on average.

Subsamples of the trip data are prepared to reduce computational load and evaluate the effects for different demand densities (demand penetration of the ARP service). The subsamples are created by randomly selecting a fraction of the trips from the original data set. Subsamples of 10%, 25%, 50%, and 100% demand penetration of the original data set are created. Except for the 100% case, five different random seeds are used to create five different subsamples for each demand penetration level.

4.2.2 Manhattan, NYC

A similar data set is used for Manhattan, New York City, which is a popular case study to evaluate MoD services (e.g., Alonso-Mora et al. [2017b], Dandl et al. [2019], and Zhang and Pavone [2016]). The network is also extracted from OpenStreetMap. As the Manhattan network is smaller than the Chicago network, all edges are kept. The network consists of $4,410$ nodes and $9,574$ edges. This relatively small number of nodes allows a complete lookup table to store the travel times and distances between all nodes. Similarly to the Chicago case study, all nodes with adjacent edges not labeled as „primary“, „secondary“, „tertiary“or unlabeled edges are not considered access nodes. Edges and access nodes are shown in Fig. 4.3b.

The publicly available taxi data set is used to create demand for the ARP service [City of New York, 2024]. TNC trips for 2018/11/12 (a randomly chosen Monday) are extracted, which were also used in the studies by Dandl et al. [2020b], Engelhardt et al. [2022a], Engelhardt et al. [2022c], and Syed et al. [2021]. A similar trip filtering process as for the Chicago data set is applied, but values have been slightly adapted to account for the smaller network and possibly more congested traffic states: In contrast to the Chicago data set, trip distances larger than 50km and trip times larger than 3hours are removed. The minimum average speed is reduced to 3km/h. Other filtering criteria remain the same. After the filtering process, $213,996$ trips remain.

Similarly, travel time scaling factors are computed for each hour of the day.

Subsamples of 10%, 25%, 50%, and 100% demand penetration of the original data set are created.

4.2.3 Munich, Germany

The final case study is Munich, Germany. In contrast to Manhattan and Chicago, no publicly available TNC data set is available for Munich. Although Uber or taxi services generally operate in Munich, the mode share of TNC and taxi services is significantly lower than in the US. Therefore, this data would be of limited use for evaluating large-scale ARP services.

Instead, a synthetic demand data set is created based on private vehicle trips in Munich, resembling a what-if scenario for the case private vehicle trips are replaced by ARP trips. Network and demand data is based on a microscopic traffic simulation model of Munich, described in [Bracher, 2019; Dandl et al., 2017]. The network has been constructed manually in large parts to provide correct lane numbers, capacities, and speed limits. It includes all roads within the inner ring road B2R, while a major arterial network is provided until the

outer highway ring A99. Major roads reach even further outside the city boundary to include commuter traffic. After exporting the network from the microscopic traffic simulation model, the input network includes $35,474$ and $64,888$ edges. The relevant part for the simulation in this network is shown in Fig. 4.3c. The figure also shows the assumed operating area in this thesis, which spans around Munich's city center and a major part of the city area.

The calibrated microscopic traffic simulation model includes private vehicle trip Origin-Destination (OD) matrices for a typical weekday in Munich from 2016 for each hour of the day. These matrices are used to create trip requests for the ARP service. Entries in the OD matrices are used as Poisson rates for the number of requests between each OD zone pair. If zones do not fully overlap with the operating area, the Poisson rates are decreased by the areal fraction of the zone outside the operating area. Up to 15% of the private vehicle trips are converted to requests for the ARP service, which results in a total of around $104,000$ requests depending on the random seed for the Poisson process.

Similarly to the other case studies, access nodes are defined where customers can board or alight a vehicle. Access nodes are defined as all intersections within the operating area not connected to road sections with a speed limit higher than 50km/h to prevent boarding on major roads. The resulting 5113 access nodes are shown in Fig. 4.3c. During the demand generation process, request origin and destination are chosen randomly from the set of access nodes within the corresponding OD zones.

Network travel times are extracted from the microscopic traffic simulation model as well. After running the traffic simulation model, the average travel time for vehicles to pass each edge is extracted for each hour of the day. These are used as the deterministic travel times for the ARP simulations.

Subsamples of 1%, 2%, 5%, 10%, and 15% demand penetration of the original data set are created. Five random seeds are used to sample from the OD matrices for each demand penetration level.

4.2.4 Comparison of the Case Studies

Table 4.1 compares the main characteristics of the input data for the three case studies. The operating area is the largest for Chicago, followed by Munich and Manhattan. Thereby, Chicago's operating area is around 10 times larger than Manhattan's and over 3 times larger than Munich's. The number of trips in the data set is the largest for Manhattan, followed by Chicago and Munich. Due to the high number of trips and the small operating area, the Manhattan data set has by far the highest trip density, with 151 trips per square kilometer per hour, followed by Munich and Chicago. The average trip length directly correlates with the operating area, with the longest trips in Chicago and the shortest in Manhattan. Due to heavy traffic conditions throughout Manhattan, the average speed[2] in Manhattan is only around half the speed of Munich and Chicago. Comparing average trip lengths and travel speed in Munich and Manhattan, the average trip travel time results in a similar value of around 10 minutes. Due to long trips, the average travel time in Chicago is around 17 minutes.

Figure 4.3 shows the street networks, access nodes, and spatial demand distribution of

[2]Average speed is defined as the average travel speed of all trips in the data sets on the fastest path during the corresponding travel time slice.

Case Study	Operating Area	No. Trips*	Trip Density	Avg. Trip Length	Avg. Speed**
Munich	175 km^2	∼104,000	∼24.8 $\frac{1}{\text{km}^2\text{h}}$	6.0 km	34.6 $\frac{\text{km}}{\text{h}}$
Manhattan	59 km^2	213,996	151 $\frac{1}{\text{km}^2\text{h}}$	2.9 km	16.1 $\frac{\text{km}}{\text{h}}$
Chicago	627 km^2	127,528	8.47 $\frac{1}{\text{km}^2\text{h}}$	9.2 km	31.7 $\frac{\text{km}}{\text{h}}$

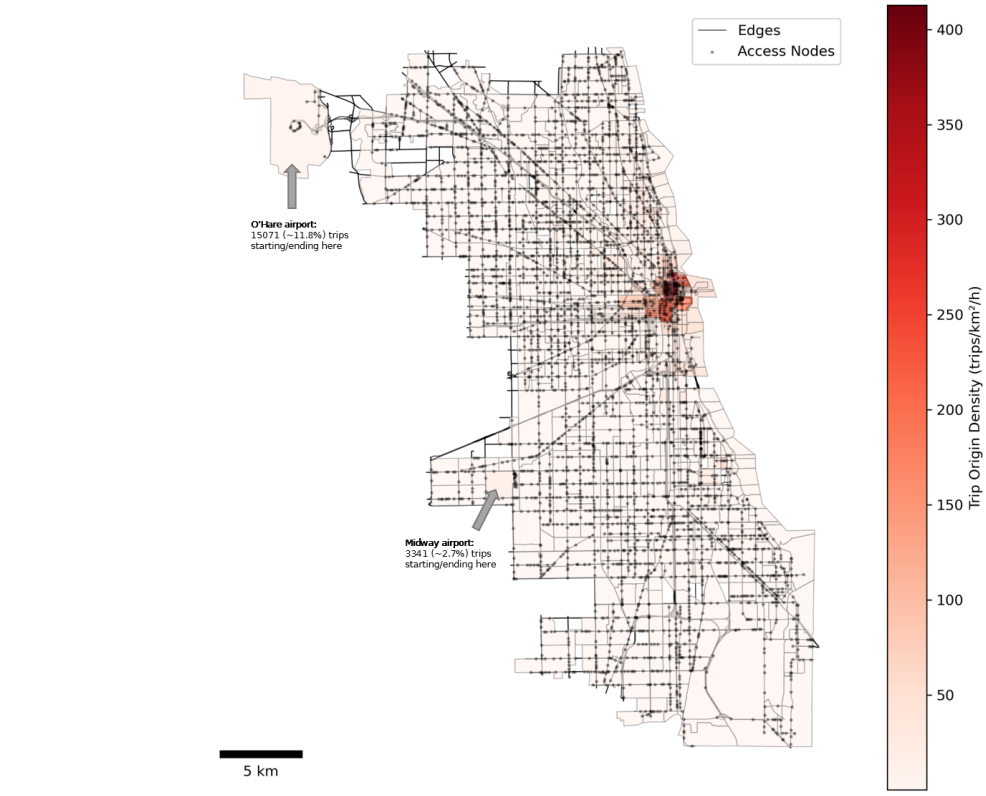
Table 4.1: Comparison of main characteristics of the input data for the three case studies. *Number of Trips refers to all trips in the Manhattan and Chicago data set and 10% of private vehicle trips in the Munich case study. **Average speed is the average travel speed of all trips in the data sets on the fastest path during the corresponding travel time slice.

request origins for the different case studies. The map of Chicago in Fig. 4.3a shows the large spatial spread of the operating area. A clear demand hot spot around the city center can be observed at the density of trip origins, with a lower demand density in the surrounding areas. Taking a closer look at the data, two further hot spots can be observed that are masked in the presentation of Fig. 4.3a by the large size of the respective census tract zones: One hot spot is located around O'Hare International Airport in the northwest of the city, and the other is around Midway International Airport in the southwest. Especially the former attracts a considerable portion of the TNC trips in the data set, aggregating to around 12% of all trips starting or ending at O'Hare. Midway Airport attracts around 3% or all trips starting or ending there, a smaller but significant portion of the trips.

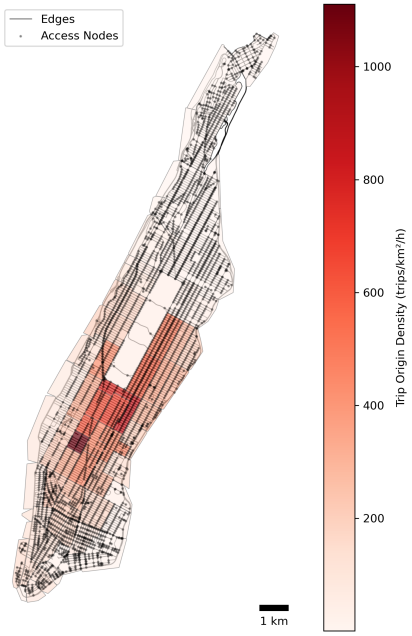
Fig. 4.3b shows the elongated island of Manhattan with its corresponding network and demand distribution. In contrast to Chicago, the Manhattan case study does not show multiple prominent hot spots but rather a demand centering around Manhattan midtown and the area around the southern part of Central Park. The highest demand density is observed around Penn Station and Grand Central Terminal, two major transportation hubs in Manhattan. These areas show extreme demand density, with over $1,000$ requests per square kilometer and hour originating there. The demand density decreases towards the northern and southern parts of the island.

Finally, Fig. 4.3c shows the street network, access nodes, and spatial demand distribution of request origins for the Munich case study. The street network does not show a clear grid structure as in the US cities, but rather a more organic structure with a higher density of roads in the city center and a lower density in the outskirts. The demand distribution shows an apparent demand centering around the city center, with a lower demand density in the surrounding areas. In contrast to the case studies of the US cities, the incline of the demand density towards the outskirts is less steep, and the demand density is generally lower.

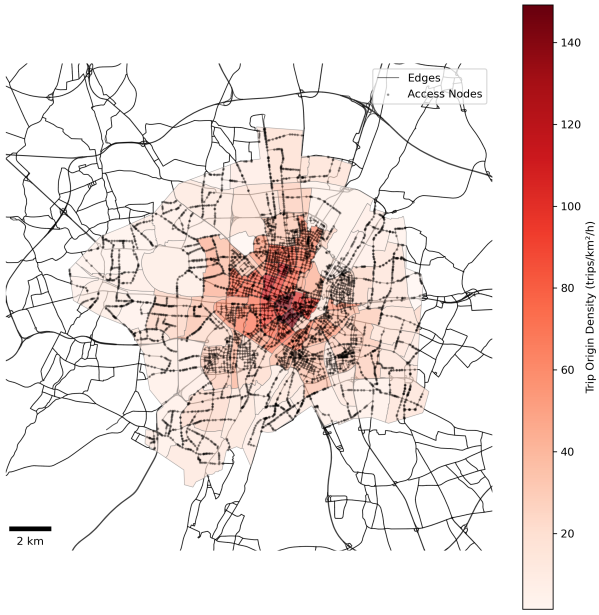
Fig. 4.4 compares the temporal request distribution, trip distance distribution, and average network speed for the three case studies. The temporal request distribution in Fig. 4.4a shows, at first glance, the overall highest demand in Manhattan, compared to Munich and Chicago. In Manhattan, the demand is spread more evenly during day times from 7 a.m. to 10 p.m., with a peak around 6 p.m. Munich and Chicago, on the other hand, show a clear peak in demand in the morning and the evening, with a lower demand during the day. Both peaks show similar demand levels in the morning and the evening, with a slightly broader evening



(a) Chicago.



(b) Manhattan.



(c) Munich (10% PV Demand).

Figure 4.3: Street networks, access nodes and spatial demand distribution of request origins for the different vase studies. Zonal aggregation refers to census tracts, taxi zones and municipalities for Chicago, Manhattan and Munich, respectively.

peak. The main differences in the distributions between Munich and Chicago are, on the one hand, a slightly delayed morning and evening peak, with the maxima appearing 1-2 hours later in Chicago. On the other hand, there is a higher demand during late evening and nighttimes in Chicago. Considering the data sources, this might be due to the fact the Chicago data set includes TNC trips, while the Munich data set includes private vehicle trips, which are less likely to be conducted during late nighttimes.

Observing the trip distance distribution in Fig. 4.4b, the number of short trips under 6km in Manhattan is striking. On a relative scale, also for the Chicago case study, a more significant portion of very short trips below 4km can be observed compared to Munich, which might result from short trips being more convenient to be conducted with a TNC service compared to a private vehicle as parking is more difficult in the city center. Due to the large operating area, the Chicago trip length distribution shows a long tail of long trips above 20km, which are rarely observed in Munich and Manhattan. Especially prominent in the Chicago data set are trips with a length of around 30km, likely to be trips to or from the airports.

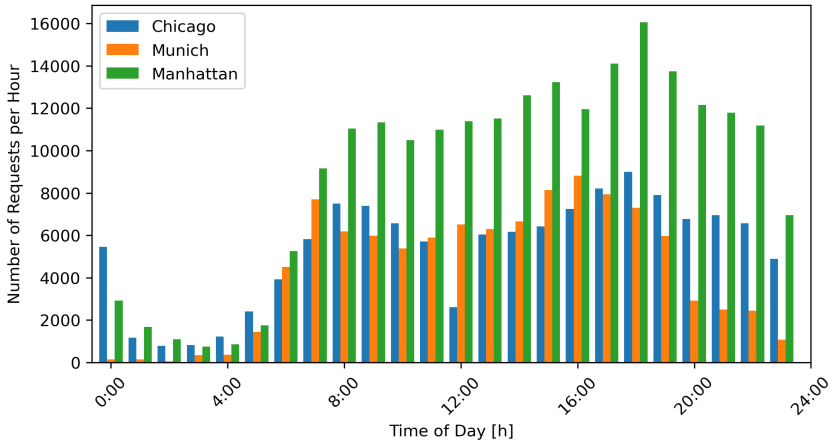
Finally, the average network speed in Fig. 4.4c shows as lower average speed in Manhattan compared to Munich and Chicago. Due to high congestion levels and only a few highways, the average speed falls below 14km/h in the morning and does not recover until the evening. In contrast, Munich and Chicago show typical traffic patterns of congestion during morning and evening peak hours. During the morning peak, the average speed drops below 30km/h in both cities, while the evening congestion appears more severe in Chicago. As the operating area includes a larger part of the highway network in Chicago, compared to the excluded outer highways ring in Munich, the average speed in Chicago is higher than in Munich late at night. Another reason might be that the travel times of TNC trips are used to calibrate the network travel times in Chicago, which might be biased towards long trips using (not congested) highways at nighttimes.

Overall, this comparison shows that the three case studies represent different urban mobility scenarios, which allows for a comprehensive evaluation of the proposed methods in various settings. The Munich case study represents a typical European city with a moderate demand density centered around the city center and a minor decline of demand towards the outskirts. The Manhattan case study represents a highly dense urban area with a high demand density and many short trips. The Chicago case study represents a large urban area with a lower demand density but distinct hot spots, especially within the city center and its airports.

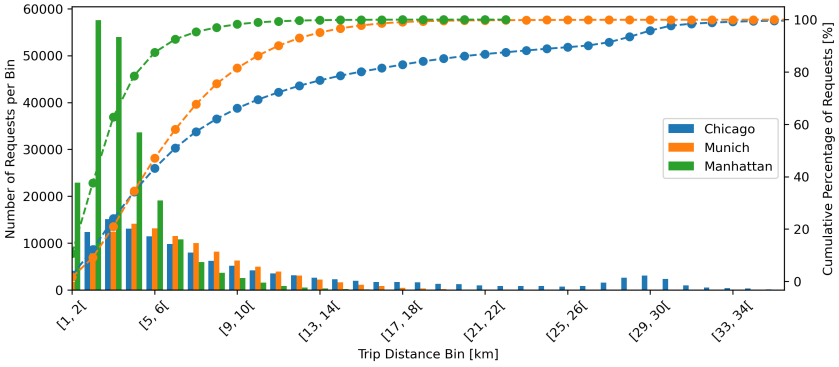
4.2.5 Zone System

When defining zone systems for repositioning, the service design should be taken into account. For the studied ARP service, the maximum waiting time constraint is crucial for the zone system design as it defines the maximum distance a vehicle can cover to serve requests when repositioned to a particular zone. If zones are chosen too big, gaps in the service might occur, while too small zones might lead to unnecessary repositioning.

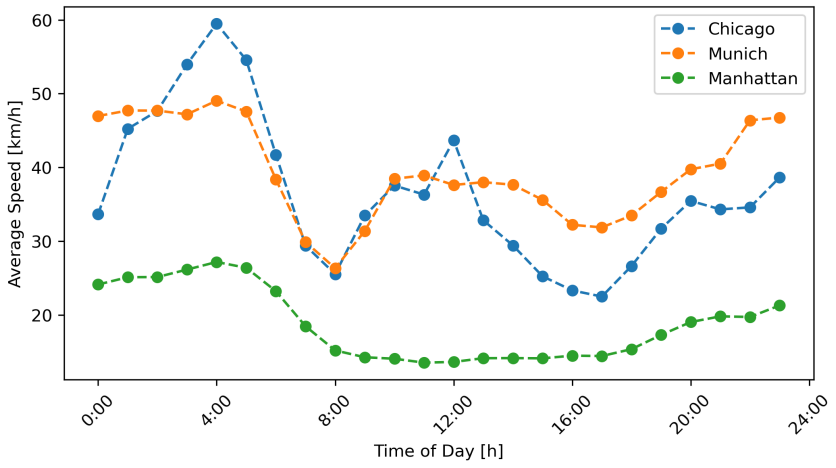
Therefore, this thesis implements a zone system that incorporates the underlying network structure and the operational constraints of the ARP service. Zones and corresponding centroids are created solving a maximum coverage problem defined in the Appendix I. The minimum set of zone centroid nodes is determined that guarantees that each access node is



(a) Temporal request distribution.



(b) Trip distance distribution. Bars refer to the number of trips in each distance bin. Dashed lines refer to the cumulative distribution.



(c) Average network speed.

Figure 4.4: Comparing temporal request distribution, trip distance distribution and average network speed for the three case studies.

reachable by at least one centroid node within a maximum driving time of t^Z_{max}. Zones are created by assigning each access node to the closest centroid node in terms of network travel time. In the base case, the goal is to set $t^Z_{max} = t^{wait}_{max} = 8$min to receive the minimal number of zones while guaranteeing that each access node is reachable from at least one centroid node within the maximum waiting time. As travel times in the network change each hour of simulation time, the zone system is created based on the time bin with the lowest network travel times.

Figure 4.5 shows the resulting centroids for the Chicago, Manhattan, and Munich case study with different values for t^Z_{max}. The depicted zones are convex hulls of nodes associated with the same centroid for the base case of $t^Z_{max} = t^{wait}_{max} = 8$min. The convex overlap, because of the directionality of the network and the different travel times in the network. Nevertheless, each node is assigned to exactly one centroid node, which is the closest in terms of network travel time. For $t^Z_{max} = 8$min, 60, 17, and 14 zones are created for the Chicago, Munich, and Manhattan case studies, respectively. The high value for the Chicago case study shows the large operating area covered. For $t^Z_{max} = 4$min these values increase to 208, 87, and 58 zones, while for $t^Z_{max} = 12$min these values decrease to 28, 14, and 8 zones, respectively.

4.3 Input Parameters and Key Performance Indicators

4.3.1 Standard Input Parameters

Table 4.2 summarizes all parameters introduced and shows the standard values used for the simulations in the three case studies. These values are used in all simulations performed if not explicitly stated otherwise.

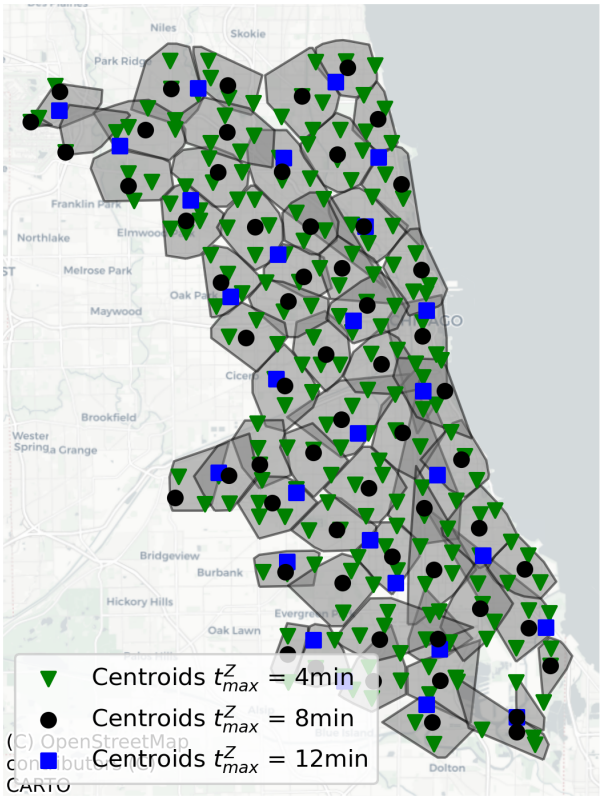
Values for fleet size ($|V|$), forecast horizons ($\mathfrak{H}, \mathfrak{H}_{QT}, \mathfrak{H}_{Hor}$) and demand scaling factors (μ_{QT}, μ_{Hor}) were chosen based on the results, which will be presented in the following section. Cost parameters for the objective function ω_{dis} and ω_{del} are set based on Tirachini and Antoniou [2020] and Frei et al. [2017], respectively.

For the assignment algorithm, the method described in section 3.2.4 is used, while the sampling-based repositioning algorithm described in section 3.3 is applied for repositioning. When evaluating the assignment and the repositioning algorithm, only on-demand requests are considered, while pre-booking of trips is only considered in the last section of the results chapter.

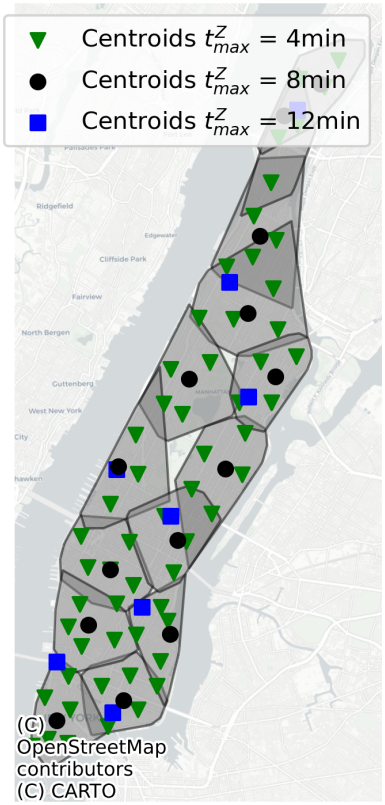
As described above, zones with centroids that are reachable from each access node within $t^Z_{max} = t^{wait}_{max} = 8$min are used in the base case. The corresponding forecast is assumed to be perfect in terms of average values within the spatio-temporal aggregation.

A simulation runs for a whole day in time steps of 30 seconds. Vehicles are randomly initiated at the beginning of the simulation at access nodes. At least 3 random seeds are simulated for each scenario using different initial vehicle distributions and demand samples. Results show average values over all seeds, with error bars indicating minimum and maximum values[3].

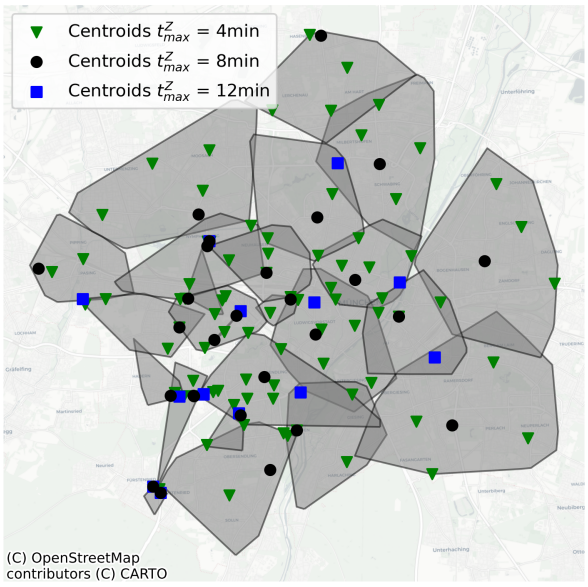
[3]The source of randomness comes from the initial vehicle distribution and the demand sample. The results show that stochastic variation is very low (i.e., error bars might not even be visible) in most evaluations. This is likely due to the large number of requests and the stability of the vehicle distribution due to the



(a) Chicago.



(b) Manhattan.



(c) Munich.

Figure 4.5: Centroids for the zone systems in the Chicago, Manhattan, and Munich case study with different vales for t^Z_{max}. Depicted zones are convex hulls of nodes associated to the same centroid with t^Z_{max}=8min.

The simulation is implemented in Python 3.7, while optimization problems are solved using *Gurobi 10*[4]. Except for the reservation results chapter, all simulations are conducted on a working station with an Intel Xeon Silver 4216 CPU with 32 cores and 192GB of RAM. The reservation results are computed on a Linux Cluster with 28 Cores with 2.6 GHz frequency per Node with 64GB RAM per node[5]. All simulations are conducted in a single-threaded environment.

4.3.2 Key Performance Indicators

Based on the output data of the simulation described in section 4.1.3, the following KPIs are calculated to evaluate the performance of the ARP service:

Served Customers This KPI refers to the fraction of customers served by the ARP service.

$$\text{Served Customers} = \frac{|R_{served}|}{|R|} \ , \tag{4.21}$$

with $|R_{served}|$ being the number of requests that have been served and $|R|$ being the total number of requests.

Vehicle Kilometers Traveled (VKT) This KPI refers to the total distance traveled by all vehicles in the simulation.

$$\text{Vehicle Kilometers Traveled} = \sum_{v \in V} \sum_{l \in L_v} \text{distance}(l) \ , \tag{4.22}$$

with L_v being the set of legs of vehicle v and distance(l) being the distance of leg l.

Average Vehicle Occupancy The average vehicle occupancy measures the average number of travelers on board a vehicle weighted by the distance they spent on-board the vehicle:

$$\text{Average Vehicle Occupancy} = \frac{\sum_{v \in V} \sum_{l \in L_v} \text{occupancy}(l) \cdot \text{distance}(l)}{\sum_{v \in V} \sum_{l \in L_v} \text{distance}(l)} \ , \tag{4.23}$$

with occupancy(l) being the occupancy of leg l.

Saved Distance While the average vehicle occupancy already measures the efficiency of pooling, and is a metric that is easy to comprehend, this KPI benefits from customers driving a long distance inside the vehicle, potentially with long detours. Therefore, the quantity „saved distance“is introduced to measure the pooling efficiency:

$$\text{Saved Distance} = \frac{\sum_{r \in R_{served}} \text{direct distance}(r) - \text{Vehicle Kilometers Traveled}}{\sum_{r \in R_{served}} \text{direct distance}(r)} \ . \tag{4.24}$$

repositioning strategy.

[4]https://www.gurobi.com/

[5]The Cluster was provided by the Leibniz Supercomputing Center (LRZ) in Garching, Germany.

Category	Parameter	Symbol	Description	Standard Value
General	Start Time	-	Simulation start time	0s
	End Time	-	Simulation end time	86400s
	Time Step	Δ_S	Simulation time step	30s
	Max. Waiting Time	t_{max}^{wait}	Maximum request waiting time	8min
	Max. Detour Time	Δ_{det}	Maximum request detour time relative to direct trip	40%
	Boarding Time	t_B	Time duration, a vehicle stops to board/alight customers	30s
	Vehicle Capacity	c_v	Number of passenger seats per vehicle	4
	Fleet Size	$\|V\|$	Number of vehicles operated	Chicago: 340 Munich: 240 Manhattan: 320
	Demand Penetration	-	Fraction of subsampled trips for ARP demand	Chicago: 20% Munich: 2% Manhattan: 20%
Assignment	Optimization Epoch	Δ_E	Time duration between consecutive batch optimizations	30s
	Obj. Assignment Reward	p_r	Reward for assignment of customer in objective	100 $
	Obj. Distance Cost	ω_{dis}	Cost weight on driven distance in objective	$0.694\$/km$
	Obj. Value of Time	ω_{del}	Cost weight on customer travel time	$16.5\$/h$
	RV vehicles	N_{RV}^{max}	Max. number of vehicles per request in RV heuristic	20
	Max. Tour per V2RB	N_{V2RB}^{max}	Max. number of schedules per V2RB	4
	Vehicle Search Time Out	$\nu_{TO,v}$	Max. computational time per vehicle to create V2RBs	1s
	Lock Assignment	$t_{th}^{reassign}$	Time before pick-up to lock current assignment	-
	Re-Assignment Penalty	$p_{reassign}$	Penalty for customer-vehicle re-assignment	-
	Re-Assignment Time Window	Δ_{TW}	Pick-up time window size after first assignment	-
Repositioning	Repo. Epoch	Δ_R	Time between consecutive repositioning epochs	900s
	Repo. Zone System	Z_R	Zones for repositioning	Centroids within $t_{max}^Z = 8$min
	Forecast. Zone System	Z_{FC}	Spatial aggregation of demand forecast	Same as Z_R
	Forecast Horizon	$\mathfrak{H}$	Horizon to forecast future requests for repositioning	Chicago: 3600s Munich: 2700s Manhattan: 2700s
	Forecast Bin Size	δ_T^{FC}	Temporal bin sizes for demand prediction	900s
	Forecast Method	-	Method to create request forecast	Perfect Distribution
	Forecast Samples	N_S	Number of forecast samples considered in sampling repo algorithm	1
	Future Weight	γ	Weight on future rewards in repo objective	0.5
	Forecast Horizon (QT)	$\mathfrak{H}_{QT}$	Forecast horizon for QT-algorithm	2700s
	Demand Scaling (QT)	μ_{QT}	Demand scaling factor for QT-algorithm	0.3
	Forecast Horizon (*Hor*)	$\mathfrak{H}_{Hor}$	Forecast horizon for *Hor*-algorithm	2700s
	Demand Scaling (*Hor*)	μ_{Hor}	Demand scaling factor for *Hor*-algorithm	0.05
Reservation	Reservation Requests	-	Distribution of reservation requests	None
	Short Term Horizon	T_h^{short}	Horizon to consider upcoming reservation requests in online optimization	540s
	Revelation Horizon	T_h^{rev}	Horizon to reveal upcoming reservation requests in online schedule	1080s
	Repos. Buffer	T_h^{repo}	Buffer time for vehicles considered available for repo.	900s
	ODM Assignment Reward	p_r^{odm}	Obj. reward for assigning on-demand customers	100 $
	Res. Assignment Reward	p_r^{res}	Obj. reward for assigning reservation customers	1000 $

Table 4.2: Collection of parameters and their standard values used in the case studies.

The direct distance is the distance between the origin and destination of a request, i.e., the distance of the trip if made by private vehicle. This KPI therefore measures the fraction of the direct distance saved by pooling. If the KPI is negative, the pooling service increases the VKT. (e.g., by a large fraction of empty vehicle trips) compared to private vehicle trips. If it is positive, sharing of rides exceeds introduced empty vehicle trips, and the pooling service is more efficient than private vehicle trips, thus „saves“ VKT in the system.

Vehicle Revenue Hours Vehicle Revenue Hours (VRH) measures the time a vehicle spends in operation actively serving customers and therefore producing revenue for the operator.

$$\text{Vehicle Revenue Hours} = \frac{1}{|V|} \sum_{v \in V} \sum_{l \in L_v^{rev}} \text{duration}(l) \ , \tag{4.25}$$

with L_v^{rev} being the set of legs of vehicle v that are revenue generating (not idle and not repositioning) and duration(l) being the duration of leg l. Because one day of simulation is conducted in this thesis, this quantity can take values between 0h and 24h.

Average Waiting Time Waiting time measures the difference between the earliest pick-up time and the actual pick-up time of a request.

$$\text{Average Waiting Time} = \frac{\sum_{r \in R_{served}} t_r^{pu} - t_r^e}{|R_{served}|} \ . \tag{4.26}$$

Note that the earliest pick-up time is chosen as a reference time to measure waiting time, also for pre-booking requests. For on-demand requests, the earliest pick-up time coincides with the request time.

Average Travel Time The average travel time measures the time a traveler spends on board a vehicle.

$$\text{Average Travel Time} = \frac{\sum_{r \in R_{served}} t_r^{do} - t_r^{pu}}{|R_{served}|} \ . \tag{4.27}$$

Average Relative Detour Time The average detour time measures the difference between the direct travel time of a request and the actual travel time. It, therefore, measures the additional time a traveler spends in the vehicle due to detours by sharing rides.

$$\text{Average Relative Detour Time} = \frac{\sum_{r \in R_{served}} t_r^{do} - t_r^{pu} - t_B - \text{direct time}(r)}{|R_{served}|} \ . \tag{4.28}$$

As the pick-up and drop-off time is set at the beginning of the corresponding boarding leg, the boarding time t_B is subtracted from the actual travel time to obtain 0 detour on a direct trip. direct time(r) is the direct travel time of request r, which is set according to network travel times at the earliest pick-up time of the request.

Average Delay Time The average delay accounts for the waiting and detour time of a traveler compared to a direct trip:

$$\text{Average Delay Time} = \frac{\sum_{r \in R_{served}} t_r^{do} - t_r^e - \text{direct time}(r)}{|R_{served}|} \ . \tag{4.29}$$

Further KPIs that are used in specific sections of the results chapter are introduced in the respective sections.

Chapter 5

Results

This chapter presents the results of the simulation study conducted in this thesis. The chapter is structured as follows: First, the general impacts of the ARP service in the three different case studies are presented and discussed in Section 5.1. Second, the assignment process is evaluated in more detail, focusing on the efficiency and impact of re-assignment in Section 5.2. The proposed repositioning approach is assessed in Section 5.3. Finally, the implications of reservations on the ARP service are analyzed in Section 5.4.

5.1 Impacts of Ride-Pooling

This section evaluates the impact of an on-demand-only ride-pooling service in the three case studies presented. The goal is to evaluate the potential of the ARP service to serve a given demand and to analyze the implications of different fleet sizes and vehicle capacities on the service quality. Analyzing the three case studies allows the evaluation of varying demand and network structures.

5.1.1 Scenarios and Parameters

With respect to the base parameters defined in Table 4.2, the following parameters are varied in the scenarios analyzed in this section: Firstly, the fleet composition is varied by evaluating different fleet sizes and vehicle capacities. By assessing a vehicle capacity of c_v=1, pooling can be compared to a hailing service, where trips are not shared (the capacity constraint would not allow a shared ride with c_v=1). In the second part, the demand penetration, i.e., the fraction of the overall number of trips converted to ride-pooling requests, is varied to evaluate the impact of the service at different demand levels.

5.1.2 Fleet Size and Vehicle Capacity

Figure 5.1 shows variations in fleet sizes and vehicle capacities for the different case studies with varying demand penetrations.

Served requests are the main KPI quantifying sufficient supply to serve the given demand. The figure shows a steady increase in served requests with a saturating behavior for large fleet sizes close to 100% served requests. This relation can be explained by vehicles added in the under-supply regime (low number of served requests) can serve unsatisfied demand during

the whole operational time, while in the over-supply regime, additional vehicles only serve the remaining unsatisfied demand during peak times.

In this thesis, 90% served requests are used as a benchmark for choosing the vehicle fleet to balance demand and supply. The Figures 5.1a, 5.1c, and 5.1e show a detailed analysis of varying fleet sizes. For the Munich case study, 240 vehicles of capacity four are required to serve 90% of demand, corresponding to 18.7k customers at 2% of the private vehicle demand. For Chicago and Manhattan, 340 and 320 vehicles of capacity four serve 22.3k and 38.5k customers at 20% TNC trip demand, respectively. On average, an ARP vehicle serves 120 trips in Manhattan, while only 66 and 78 customers are served by each vehicle in Chicago and Munich, respectively. This results from shorter customer trips in Manhattan compared to Munich and Chicago, reducing the time for vehicles to serve customers. Additionally, the compact operating area in Manhattan allows short vehicle trips between serving customers (see Table 4.1).

Different colors in Figure 5.1 indicate varying vehicle capacities used for the service. It is evident that, with the same fleet size, the number of served customers increases with higher vehicle capacity because en-route vehicles gain greater flexibility to pick up other customers on the way to share trips of customers, increasing the effective service rate per vehicle. A huge increase in service rate is observable when changing the vehicle passenger capacity from one to two. As a vehicle capacity of one prevents pooling, this scenario effectively represents a ride-hailing service. Even allowing the sharing of trips between only two customers increases the effective supply of the vehicle fleet massively. For the Chicago case study, for example, the fraction of served requests increases by around 15% for a fleet size of 350 vehicles at 20% demand penetration (Figure 5.1c). A further, although less prominent, increase in service rate is observable when increasing the vehicle capacity from two to four as the potential of pooling further evolves. Nevertheless, another increase to a vehicle capacity of six rarely brings any benefits. In this case, vehicle capacity is no longer the limiting factor for assigning shared schedules to vehicles. Instead, time constraints of pick-up and travel time do not allow finding many shared routes between more than four passengers simultaneously. Small deviations can be observed between the case studies: While rarely any benefit can be observed for introducing capacity six vehicles in Munich, slight benefits can be observed in the high-demand scenario for Manhattan (Fig 5.1f) and for the Chicago scenarios (Figure 5.1c and 5.1c). Two trends come into play: On the one hand, the extreme demand density in Manhattan increases the chance of finding customers traveling in the same direction simultaneously. On the other hand, especially the Chicago data set consists of many trips between the airport and the city. These trips are especially suitable for pooling as almost no detour is needed for a shared trip[1].

Figures 5.1b, 5.1d and 5.1f show large scale scenarios for the respective case studies. Due to computational time, only three fleet sizes are computed for each case study. For Chicago (Figure 5.1d), 1,250 vehicles of capacity six can serve approximately 90% of the TNC Demand, while 1,700 vehicles can serve nearly all the 128k trips. On the contrary, over 2,200 vehicles would be needed to serve 90% demand if sharing of rides is not allowed (capacity one), again showing the efficiency of pooling. Similarly, 1,500 vehicles of capacity four or higher can serve almost all taxi trips in Manhattan. This fleet size corresponds to only 11% of the currently

[1] An occupancy-based spatial analysis is presented in Figure II.1 in the Appendix.

13578 licensed yellow taxi cabs New York[2]. In Munich, 1,250 vehicles of capacity four or higher serve around 102k private vehicle trips. Assuming on average three private vehicle trips per day, the ARP service would potentially replace roughly 34k private vehicles, corresponding to a replacement rate of 97%.

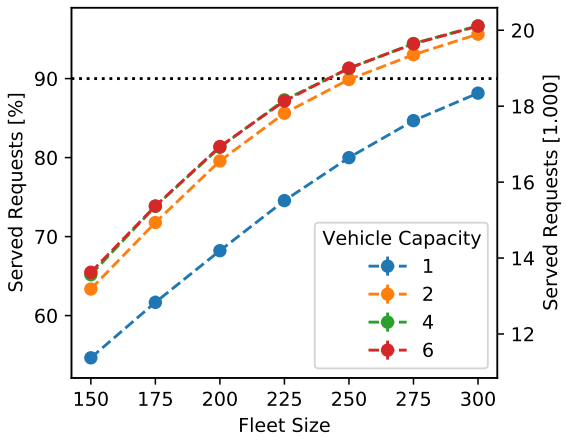
Figure 5.2 compares the effect of different vehicle capacities in more detail. The figure shows scenarios from Figure 5.1d with a similar service level of approximately 90% for the corresponding vehicle capacities as indicated in Figure 5.2a. Thereby, Figure 5.2b shows a considerable reduction in required fleet size from 2,250 vehicles when a ride-hailing service is applied (capacity one) compared to 1,250 vehicles when the service is operated in a ride-pooling mode, and more than three customers can be pooled into the same vehicle.

Figure 5.2c shows the effect of pooling from an operational perspective in more detail by evaluating the VKT. While the fleet travels over 1.2 million km to serve around 90% customers when no sharing is possible, less than half of VKT is required in pooling mode. The reduction in VKT when pooling is applied results from two effects: On the one hand, at least two customers share the highest fraction of VKT. On the other hand, empty vehicle kilometers decrease as en-route vehicles are available for service instead of only idle vehicles in the hailing case. Looking at the fraction of occupancy states for a vehicle capacity of six reveals that a further increase in capacity would not improve the pooling service further. While occupancy states of up to four are observed relatively frequently, only a tiny fraction of VKT is driven with five passengers on board. An occupancy of six is nearly not visible. This effect results from time constraints in customer pick-up and travel time, which limits the accommodation of further customers, if many customers are already scheduled.

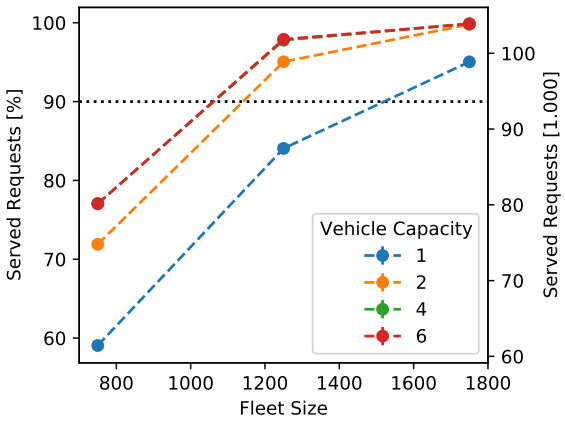
Finally, Figure 5.2d shows the trade-off of pooling from the customer perspective. The average customer delay, i.e., the increase in travel time compared to a direct trip, is differentiated by waiting and detour time. In a ride-hailing service, the customer only has to wait for the vehicle for a pick-up. When pooling is considered, detours are added to pick up and drop off other customers. The more pooling is possible; in this case, the higher the vehicle capacity, the more detours are driven by customers. Compared to a ride-hailing service, therefore the overall travel delay approximately doubles compared to vehicle capacities of four and six. Nevertheless, it is also observable that the customer waiting time slightly decreases as additional en-route vehicles can pick up a customer leading to a shorter approach on average.

As the analysis showed, pooling with a vehicle capacity of four appeared to be the best choice for the given service design for all case studies: A lower vehicle capacity constrains pooling, while larger vehicles do not benefit the operation. For the remainder of this thesis therefore vehicles of capacity four are considered.

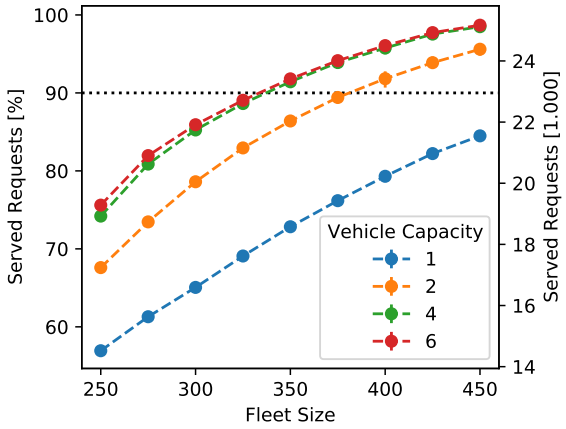
[2]https://www.nyc.gov/site/tlc/businesses/yellow-cab.page; This value refers to the whole of New York City. It can be assumed that most of the taxis serve trips in Manhattan as 84% of the taxi trips start and end within Manhattan on the day used for the Manhattan case study.



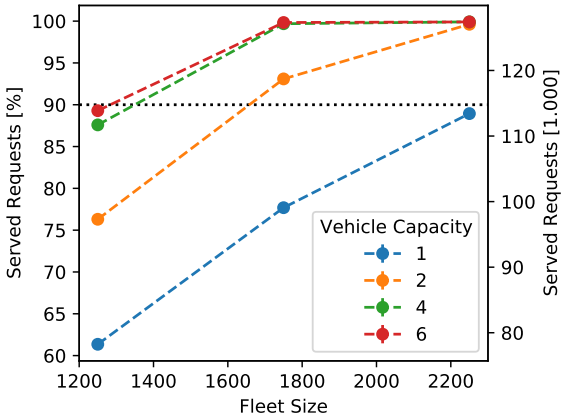
(a) Munich - 2% Demand Penetration.



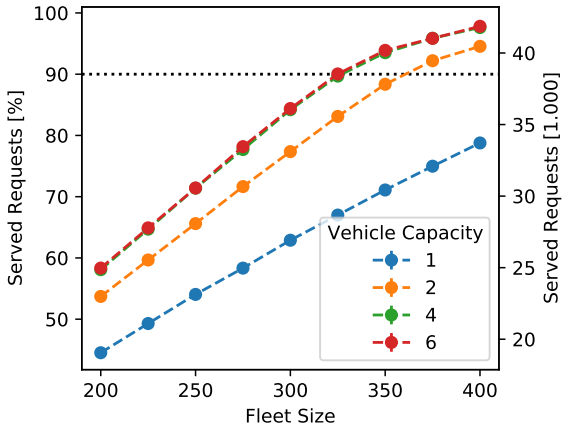
(b) Munich - 10% Demand Penetration.



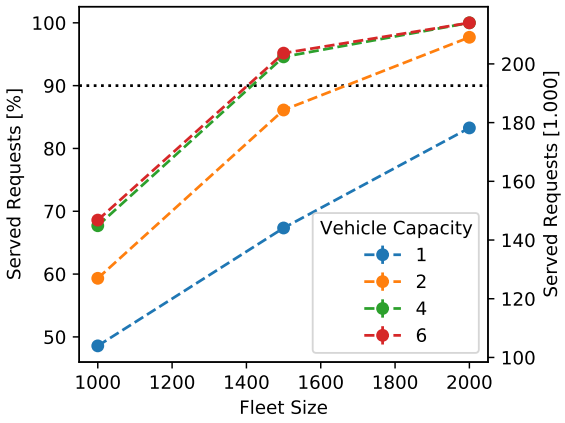
(c) Chicago - 20% Demand Penetration.



(d) Chicago - 100% Demand Penetration.

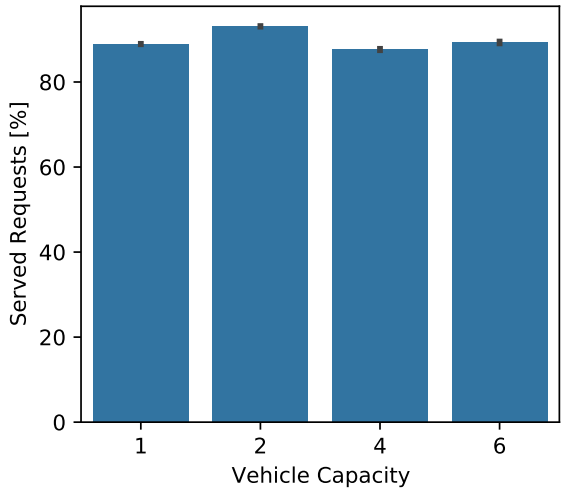


(e) Manhattan - 20% Demand Penetration.

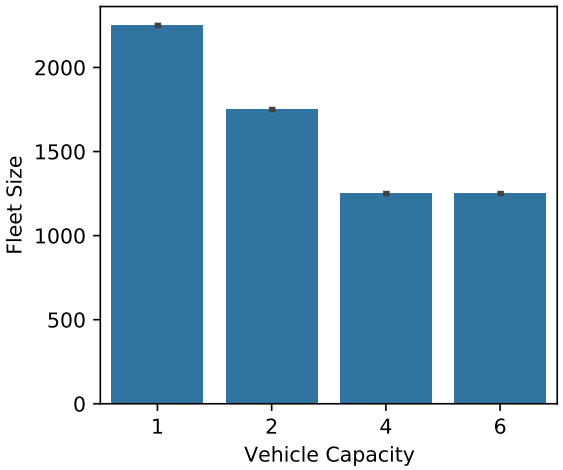


(f) Manhattan - 100% Demand Penetration.

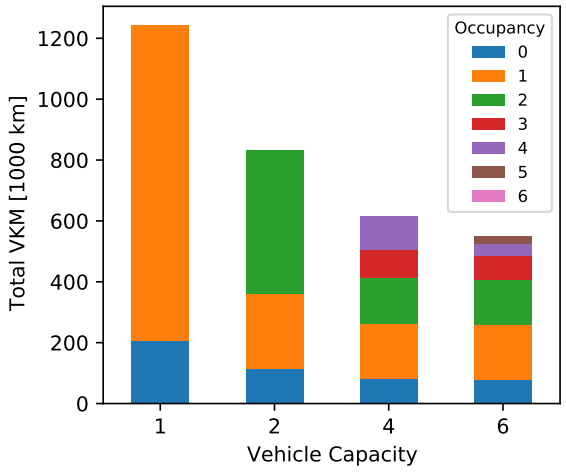
Figure 5.1: Served Requests for Different Fleet Sizes and Vehicle Capacities.



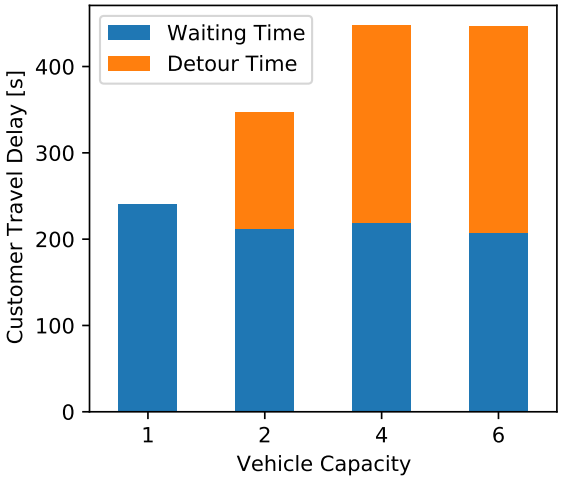
(a) Served Requests.



(b) Fleet Size.



(c) VKT differentiated by Occupancy.



(d) Customer Delay.

Figure 5.2: Comparing KPIs for different applied vehicle types with similar service level for the Chicago case study with 100% demand penetration.

5.1.3 Service Scaling

While the previous section focused on evaluating the applied vehicle capacity for an ARP service, this section emphasizes the demand for the service and its implication on operation.

Figure 5.3 therefore shows different ride-pooling KPIs for varying demand penetrations (D.P.). For Chicago and Manhattan, demand penetrations of 10%, 20%, 50%, and 100% of the overall TNC and taxi demand are evaluated, respectively. For Munich, demand varies in steps of 1%, 2%, 5%, 10%, and 15% of all private vehicle trips within the operating area. In each case study, the operated fleet size is scaled linearly with demand to vaguely maintain a balance between demand and supply. As two properties change between the case studies in Figure 5.3, two x-axes are given: One for fleet size and one for demand penetration. As different scaling factors are applied for each case study to evaluate the fleet size, the upper x-axis varies between the case studies. Colored ticks and dotted horizontal lines indicate the matching between both x-axes and data points.

Figure 5.3a and 5.3b show served requests and served requests per vehicle, respectively. While for low-demand penetrations, only up to 85% of all requests can be served, a simultaneous linear increase in demand and fleet size also increases the fraction of overall served requests, indicating higher fleet efficiency. As the probability of finding shareable trips increases with overall demand, also the number of served requests per vehicle can increase when more trips are shared. Comparing the different case studies, substantial absolute differences in served requests per vehicle are notable. Because of the high trip densities of short trips in a compact operational area, vehicles in Manhattan can serve over 120 requests per day on average. In comparison, requests in Chicago hardly reach 80 requests per vehicle.

The increase in pooling efficiency can be explained by Figures 5.3c and 5.3d showing the KPIs saved distance and empty VKT, respectively. In all case studies, saved distance increases with demand and fleet size, resulting from more efficient schedules that can be assigned to vehicles by sharing more trips with fewer detours. Saved distance is positive for nearly all scenarios tested, indicating that sharing of trips trespasses empty VKT from pick-up and repositioning trips. With a saved distance of around 42% in Chicago at 100% demand, the VKT can be reduced by 42% compared to if all customers would use the private vehicle on their trip. As the applied data stems from TNC trips, the replacement rate would be even higher, as this KPI does not account for additional empty idling trips of TNC drivers. Nevertheless, in Munich, the lowest saved distance is observed and is even negative when only 1% of private vehicle trips can be replaced. This observation indicates a more dispersed demand than the other case studies, making it harder to find shareable trips. This can be due to two reasons: 1) While Chicago and Manhattan have directional demand patterns (e.g., the airport in Chicago and the longitudinal shape of Manhattan), Munich has demand patterns directed from and to the city center. 2) In contrast to the other case studies, the request data set for Munich is generated for OD-matrices where demand is distributed homogeneously over zones and time slices, which might average out hot spot demand patterns, which are beneficial for finding shareable trips. Focusing on empty VKT, the scaling effect is also observable. While for low demand penetrations, empty VKT can be as high as 22% of the total VKT, this fraction decreases to less than 14% for high demand penetrations. Interestingly, even if Chicago showed the highest saved distance (which includes empty VKT), the fraction of empty VKT is the highest among the case studies. This observation can be explained by the directed demand

patterns in Chicago, which lead to many empty repositioning trips between the city center and the airport.

Finally, Figures 5.3e and 5.3f show average customer and waiting time, respectively. Two trends can be observed: On the one hand, there is a decrease in waiting time with higher demand penetration and fleet size, and on the other hand, there is an increase in detour time (except for Manhattan). The former observation mainly results from a higher vehicle density, resulting in an, on average, faster approach to pick-up customers. The latter results from finding more shareable trips that inevitably induce customer detours in a pooling service. The contrary observation for Manhattan likely results from the combination of short trips and a maximum detour constraint relative to a direct trip. For these trips, it seems more beneficial to assign more efficient scheduled (not necessarily) shared trips that can be found with higher demand density, increasing saved distance while decreasing customer detours.

Spatial Analysis

Figure 5.4 shows a spatial analysis of the impact of an ARP on the network of the corresponding case studies. On each link l in the network, the flow reduction is calculated by

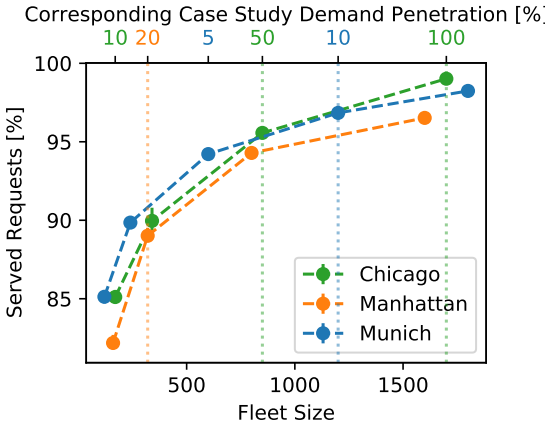
$$\text{flow reduction}(l) = \frac{\text{counts}_{direct}(l) - \text{counts}_{arp}(l)}{\text{counts}_{direct}(l)} \ . \tag{5.30}$$

$\text{counts}_{arp}(l)$ evaluates the overall number of ARP vehicles that passed a link during the simulation, while $\text{counts}_{direct}(l)$ refers to the count of vehicles passing the corresponding link l if all served customers would have taken a private vehicle on their fastest path. To reduce the visual focus on sections with large changes in relative flow (low nominators or denominators resulting from a very low number of vehicles passing the link in the simulation), the thickness of the shown links in Figure 5.4 is scaled by $\text{counts}_{direct}(l)$. Each case study shows a low-demand and a high-demand scenario.

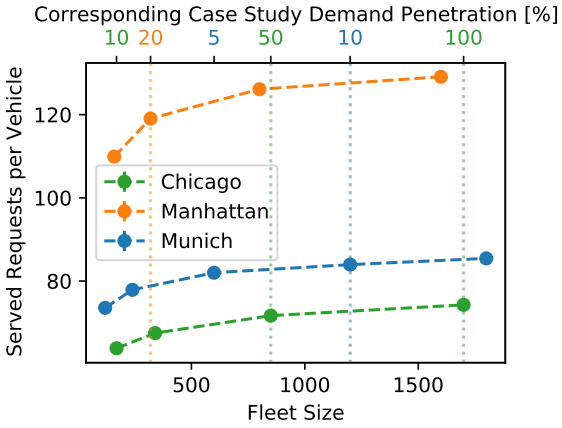
Figure 5.4a and 5.4b show the Munich case study. A prominent flow reduction and, therefore, shift in traffic can be observed for B2R („Mittlerer Ring“), a major ring road encircling the city center of Munich, indicating high occupancy of ARP vehicles traveling these links and/or ride pooling routes getting shifted to alternative paths compared to customer direct paths. Nevertheless, while the flow reduction is positive on most links in the high-demand scenarios, indicating an efficient ride-pooling service, additional traffic is induced in the low-demand scenario. This increase in flow is evident on minor roads in the outer areas of Munich, which results from additional vehicle detours to pick up and drop off customers.

A similar observation can be made for Chicago (Figure 5.4c and 5.4d): For small demand penetrations, additional trips are induced on the secondary road network because of additional pick-up and drop-off trips that are only canceled out within the high-demand scenario, which enables the finding of shareable trips also in these low-demand areas. Flow reduction can be observed on primary roads and highways in small and high-demand scenarios. The high flow reduction and trip counts on the highway connecting O'Hare airport with Chicago's city center are very prominent. This OD-relation is especially suitable for assigning pooled rides in the Chicago case study.

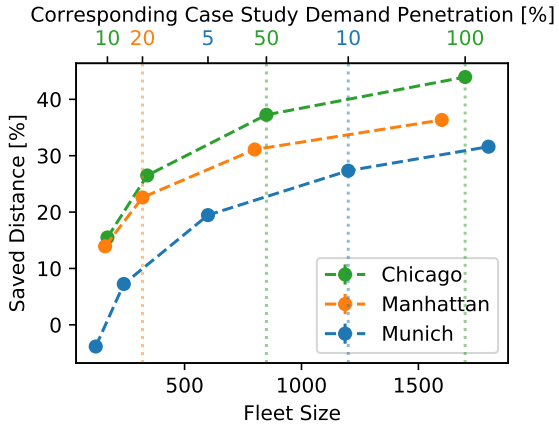
For Manhattan (Figure 5.4e and 5.4f) links with high flow reduction focus around Manhattan midtown south of Central Park, the area with the highest demand for the simulated ARP service



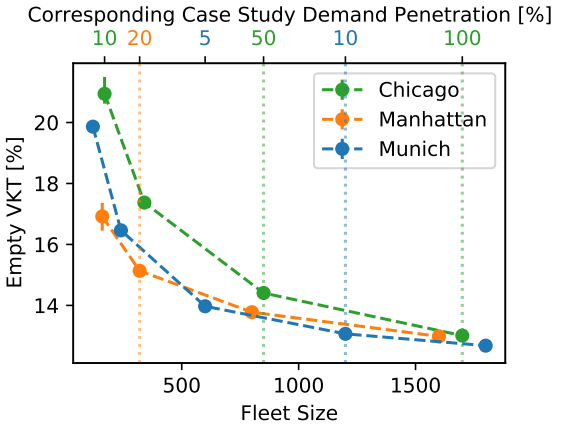
(a) Served Requests.



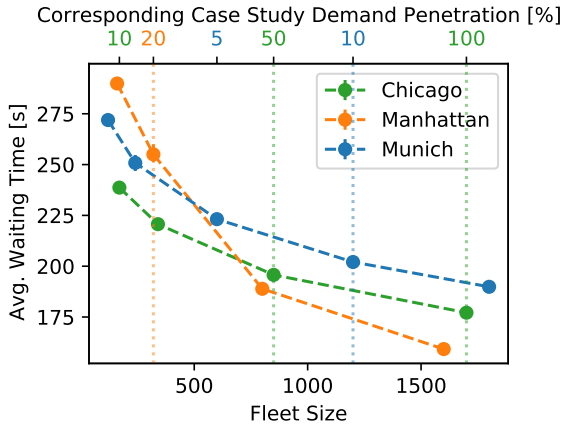
(b) Served Requests per Vehicle



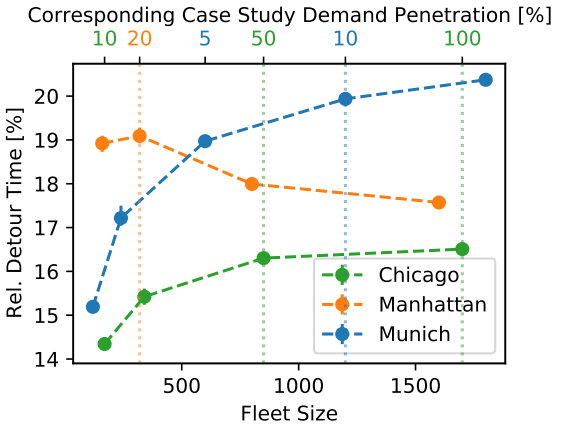
(c) Saved Distance.



(d) Empty VKT.



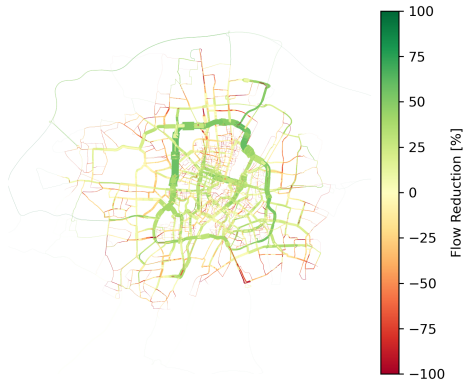
(e) Avg. Waiting Time.



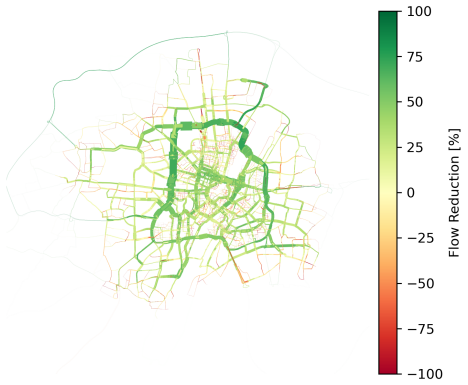
(f) Avg. Relative Detour.

Figure 5.3: Scaling Effects of Ride-Pooling. Demand penetration (D.P.) and fleet size are varied by a linear factor (Chicago: 170 veh per 10% D.P., Manhattan: 160 veh per 10% D.P., Munich: 120 veh per 1% D.P.). The lower x-axis is the same for all case studies. The color of the labels in the upper x-axis indicates the corresponding case study. Matching of the axes with data points for a given case study is indicated by vertical lines with corresponding colors.

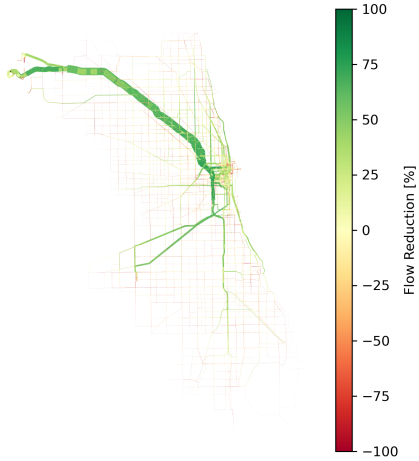
(see Figure 4.3b). Again, additional trips are induced on non-primary roads far from hot spots in demand for low-demand scenarios. This is also the case for the northern part of Manhattan, which is hardly visible in the Figure due to the very low demand compared to the center of Manhattan.



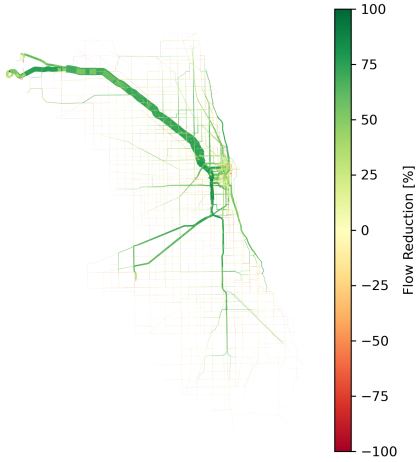
(a) Munich - 240 veh - 2% D.P.



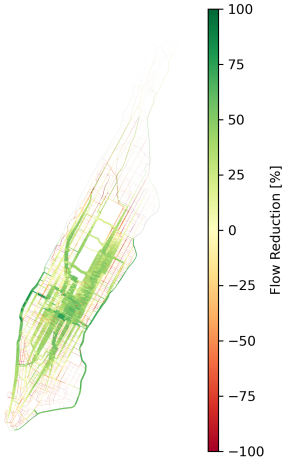
(b) Munich - 1200 veh - 10% D.P.



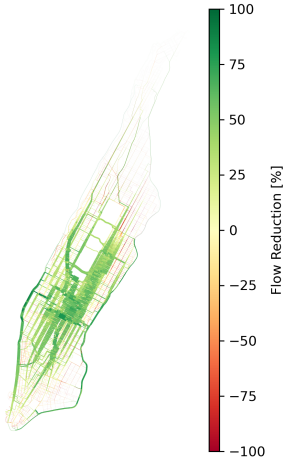
(c) Chicago - 340 veh - 20% D.P.



(d) Chicago - 1700 veh - 100% D.P.



(e) Manhattan - 320 veh - 20% D.P.



(f) Manhattan - 1600 veh - 100% D.P.

Figure 5.4: Flow Reduction of ARP service compared to direct trips (see Equation 5.30) for different case studies and demand penetrations (D.P.). Line width proportional to direct trip flows ($\text{counts}_{direct}(l)$).

5.2 Assignment

This section deals with the evaluation of assignment algorithms, the effects of re-assignment, and the assessment of the proposed strategies for increased reliability.

5.2.1 Scenarios and Parameters

The algorithms that are compared in this section are summarized in Table 5.1. As the *OPT*-algorithm is the algorithm proposed in this study, it is always used if not explicitly mentioned.

Next to variants of the assignment algorithm, different heuristics are applied to the *OPT*-algorithm. These heuristics have been described at the end of Section 3.2.4 and are summarized as follows:

- **Limited Number of Feasible Schedules Per V2RB *(LS)***: The maximum number of schedules per V2RB is constraint by N_{V2RB}^{max}.
- **Candidate Vehicle Reduction Per Request *(RV)***: The number of candidate vehicles per request is reduced to N_{RV}^{max} after the first assignment.
- **Search Timeout per Vehicle *(TO)***: The maximum time to compute V2RBs per vehicle is limited to $\nu_{TO,v}$.

Thereby, the abbreviations *OPT:TO+RV20*, for example, means that the *OPT*-algorithm is used with the *TO* and *RV* heuristic applied. For the *RV* heuristic, the number of candidate vehicles is reduced to 20 after the first assignment.

As a base case, demand penetrations of 20%, 2%, and 20% are evaluated for Chicago, Munich, and Manhattan, respectively. To also evaluate the impact of high-demand scenarios, demand penetrations of 50%, 5%, and 50% are evaluated for Chicago, Munich, and Manhattan, respectively. Higher demand penetrations are not evaluated in this section due to the excessive computational time required to assess these scenarios without the application of heuristics.

Algorithm	Short Description	Reference Section
OPT	The algorithm proposed in this thesis: If run to termination, the optimal solution to the DARP is found. Schedules are stored for later evaluation in V2RB Database.	3.2
OPT(Full)	Similar to *OPT*, but no V2RB Database is used. Instead, all feasible vehicle schedules are computed from scratch.	3.2.6 (Full Re-Build)
LA	Linear Assignment: Request-Vehicle pairs are created by solving a single vehicle DARP with the currently assigned schedule to assign new requests in batches.	3.2.6 (Linear Assignment)
LA(IH)	Similar to *LA*: Instead of solving a single vehicle DARP, candidate schedules are created by applying the insertion heuristic (*IH*).	3.2.6 (Linear Assignment)
IH	Insertion Heuristic: New requests are inserted iteratively into currently assigned vehicle schedules.	3.2.6 (Insertion Heuristic)

Table 5.1: Summary of assignment algorithms for this section.

5.2.2 Impacts of Assignment Optimality

Figure 5.5 compares the impact of different assignment algorithms on the performance of the ARP-service for a low demand scenario (Figure 5.5a) and a high demand scenario (Figure 5.5b) in an example for the Chicago case study. The figure shows different fleet KPIs for different algorithms on the y-axis. The algorithms are sorted by their available solution space for the assignment, with the largest solution space on the left and the smallest on the right.

While *OPT* refers to building the whole solution space (all feasible V2RBs) but keeping previously computed V2RBs in memory, *OPT(Full)* refers to building every feasible V2RB from scratch in each optimization epoch. Due to computational complexity, the *OPT(Full)* scenario has been only computed in the low demand scenario of Figure 5.5a. The benefits of keeping already computed V2RBs in memory are very concise when comparing the computational time of the *OPT(Full)* and *OPT* method: By keeping V2RBs in memory, the average computational time per assignment epoch can be reduced by 76% from 13.5s to 3.2s, providing a considerable advantage for real-time applications. The trade-off is minimal, as the number of served requests, saved distance, and average customer delay only deteriorate slightly and remain within the error bars. Even though *OPT* and *OPT(Full)* should produce exactly the same results in theory, a slight difference can be observed nevertheless. This is because of a slightly different procedure when travel times are updated: The *OPT* variant does not recompute all feasible schedules of current assigned V2RBs but reuses only schedules within the currently assigned V2RB. In case all schedules become infeasible, it keeps the currently assigned one still in memory to ensure a feasible assignment. Nevertheless, in the case the network travel times decrease and new schedules of this V2RB become feasible that were deemed infeasible before, slightly fewer schedules are accessible compared to the *OPT(Full)* method that always solves the single vehicle DARP.

Next, the number of computed candidate schedules for assignment is reduced by limiting the number of feasible schedules per V2RB to maximally four (*LS*) and introducing a timeout of maximally 3s to compute V2RBs per vehicle (*TO*). In the small-scale scenario, these two measures hardly have an impact on the service quality and the computational time as the number of V2RBs per vehicle and the number of feasible schedules per V2RB is low, such that these constraints are not binding. In the high-demand scenario, however, these constraints are activated, resulting in a decrease in average computational time per epoch of 35%, with only a minor deterioration in served requests while saved distance and average customer delay are not affected.

Orange scenarios in Figure 5.5 refer to scenarios with activated *RV*-heuristics that reduce the number of candidate vehicles after a first assignment is made, limiting the potential for re-assignment and, therefore, computational complexity. Similar to the *LS* and *TO* heuristics, the *RV* heuristics are merely influencing the low-demand scenario. In the high-demand scenario, on the other hand, the computational time can be reduced further by 46% due to the limited solution space. As a trade-off, slight service deterioration is observable: Served requests further decrease, while also saved distance decreases and customer delay increases, resulting from reduced options for re-assignment.

Green scenarios do not allow for re-assignment. *OPT:LS+TO+RV1* effectively fixes the

assigned vehicle per customer after an initial assignment[3]. The Linear Assignment *LA* algorithm inherently excludes re-assignments. Comparing *OPT* and *LA* therefore shows the value of re-assignment in the ARP service: In the high-demand scenario, 1.4% more customers can be served on average when re-assignment is allowed, while saved distance increases by 2.56% and average customer delay decreases by 35s. In the low-demand scenario, the differences are less pronounced but still better with re-assignment, indicating the importance of re-assignment from an operational perspective, especially for large-scale ARP services.

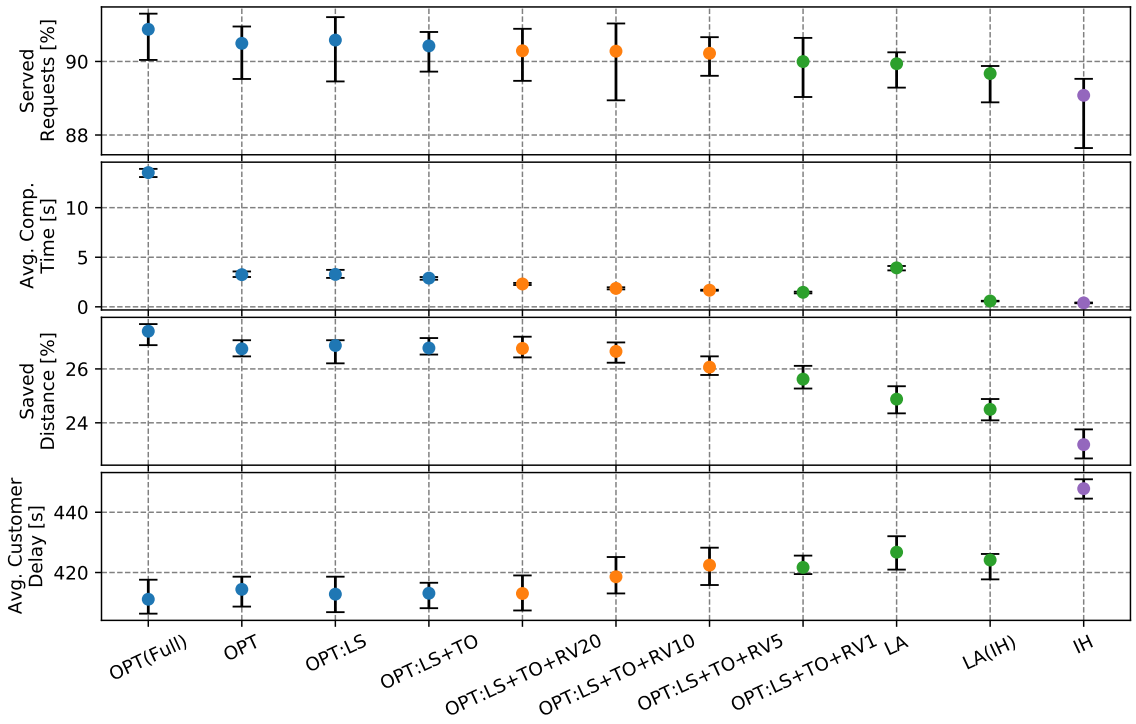
Finally, the *IH* algorithm is introduced, a simple insertion heuristic that assigns customers to vehicles iteratively in a greedy manner. In this case, the most significant service deterioration is observable. By comparing *IH* with *LA(IH)*, the Linear Assignment algorithm that only uses insertions to create candidate schedules, the deterioration can be traced back to assigning customers iteratively or in batch. Compared to the batch assignment in the high-demand scenario, 1.5% fewer customers can be served, while saved distance decreases by 3.1% and average customer delay increases by 38s. Therefore, the value of assigning customers in batch is on a similar scale as the value of re-assignment discussed in the previous paragraph. With respect to computational time, the *IH* algorithm is the fastest algorithm but also the one with the lowest service quality. Nevertheless, the *LA(IH)* algorithm is only slightly slower (0.78s per epoch on average) than the *IH* algorithm but provides a much better service quality. The *LA(IH)* algorithm might, therefore, be especially suitable for simulation studies where computational time might be even more crucial than in real-time applications to evaluate a large number of scenarios. The originally proposed *LA* is significantly slower as it exhaustively solves the DARP. Nevertheless, as the results show, the benefits are limited.

The goal of Figure 5.6 is to provide a deeper understanding of the proposed algorithm on the example of *OPT:LS+TO*. It shows the temporal analysis of a simulation for the Chicago case study with 850 vehicles. First, a stacked plot of the computational time subdivided into the computational phases in each assignment epoch is shown. It is striking that the computational time is dominated by building new V2RBs. Building V2RBs makes up 87.1% of computational time, while updating previously computed V2RBs only makes up 8.7%. Solving the assignment problem and pre-processing the *RV*-graph[4] only contributes to 3.7% and 1.2%, respectively. Another observation is the strong fluctuation in computational time, regularly exceeding the limit for real-time application of 30s in this thesis. Real-world operators must include stronger time-outs in the assignment algorithm to enforce real-time application. Nevertheless, the issue of computational time is less pronounced than it appears: On the one hand, all computations for a simulation are made on a single core. Especially building V2RBs can be parallelized easily by distributing computational tasks for each vehicle. On the other hand, algorithms are implemented in Python, which is known to be significantly slower than other programming languages like C++ (by a factor of 10-100 [VERCEL, 2024]), which would rather be used in professional applications.

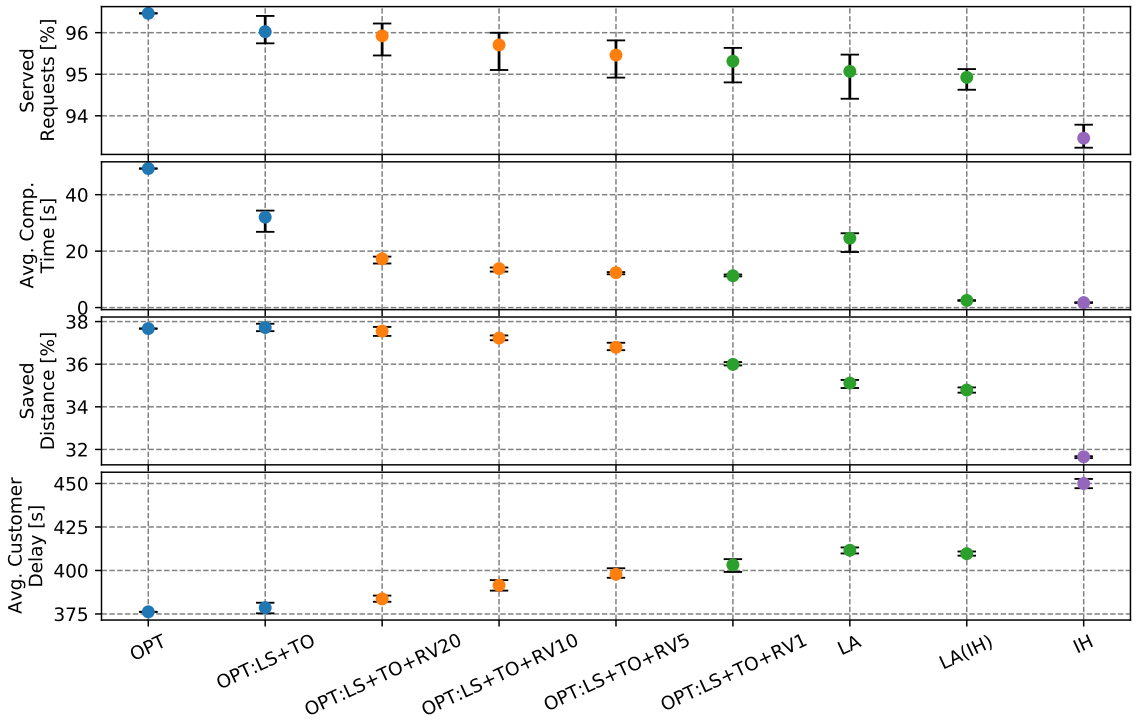
Figure 5.6 shows that the computational time is strongly correlated to number of V2RBs found. The fluctuations likely result from cliques of requests and vehicles in the *RV* and

[3] In this version, some re-assignments are still possible when network travel times changes as then the V2RB database is rebuilt from scratch

[4] Building the *RR*-graph is not directly reported, as computations are made on-demand in the building step of V2RBs.



(a) Chicago - 20% - Fleet Size: 340



(b) Chicago - 50% - Fleet Size: 850

Figure 5.5: Comparison of Assignment Algorithms. Available solution space reduces from left to right. Blue indicates full re-assignment, orange constraint re-assignment; green, no re-assignment and purple, no batch assignment. See Section 5.2.1 for abbreviations.

RR-graph that enable a magnitude of feasible schedules. Out of all V2RBs available in an optimization epoch, on average, 34.5% are still available after updating previously computed V2RBs. This shows the computational benefit of keeping them in memory: A significant amount of V2RBs remain feasible while the computational time needed to check for their feasibility only consumes a small amount (8.7% of computational time) compared to building new V2RBs.

Finally, the number of active requests is shown for each optimization epoch. Interestingly, this quantity does not always correlate with the number of feasible V2RBs or the computational effort. This observation is especially notable in the early evening between 5 and 7 p.m. when most requests are active, but computational time is comparatively low. This effect can be explained by two reasons: 1) During this period, the vehicle speed in the network is lowest, which reduces the number of feasible schedules with respect to customer time constraints, and 2) a significant fraction of fleet vehicles are occupied with customers, which reduces the number of options for incoming requests.

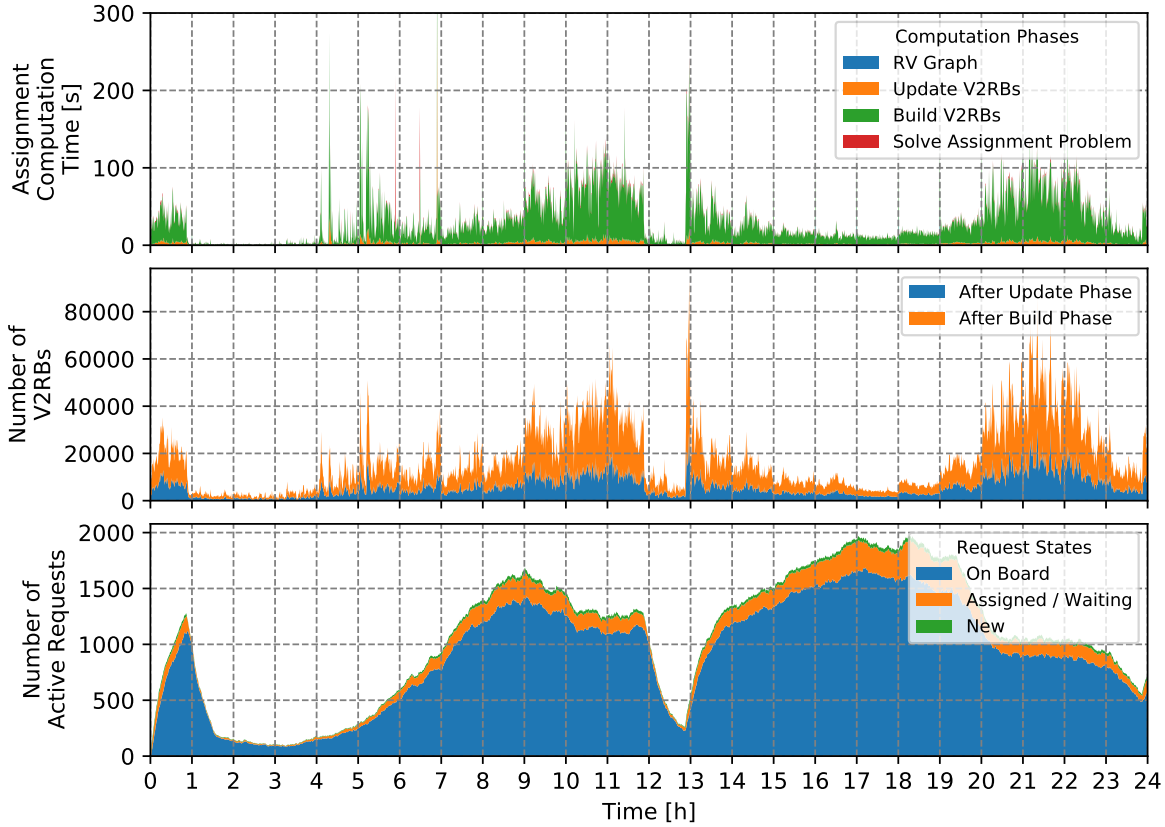


Figure 5.6: Temporal Analysis of Algorithm Performance. Case Study: Chicago - 50%; Fleet Size: 850; Algorithm: *OPT:LS+TO*

		Number of Vehicle Re-Assignments			
		0	1	2	≤ 3
Chicago	50% - Fleet Size 850	85.0%	12.7%	2.0%	0.3%
	20% - Fleet Size 340	85.4%	12.6%	1.8%	0.2%
Munich	5% - Fleet Size 600	79.7%	16.4%	3.3%	0.6%
	2% - Fleet Size 240	79.5%	17.2%	2.9%	0.3%
Manhattan	50% - Fleet Size 800	85.2%	12.4%	2.1%	0.3%
	20% - Fleet Size 320	78.6%	17.7%	3.3%	0.4%

Table 5.2: Number of Vehicle Re-Assignments per Customer for different Case Studies and Demand Scenarios.

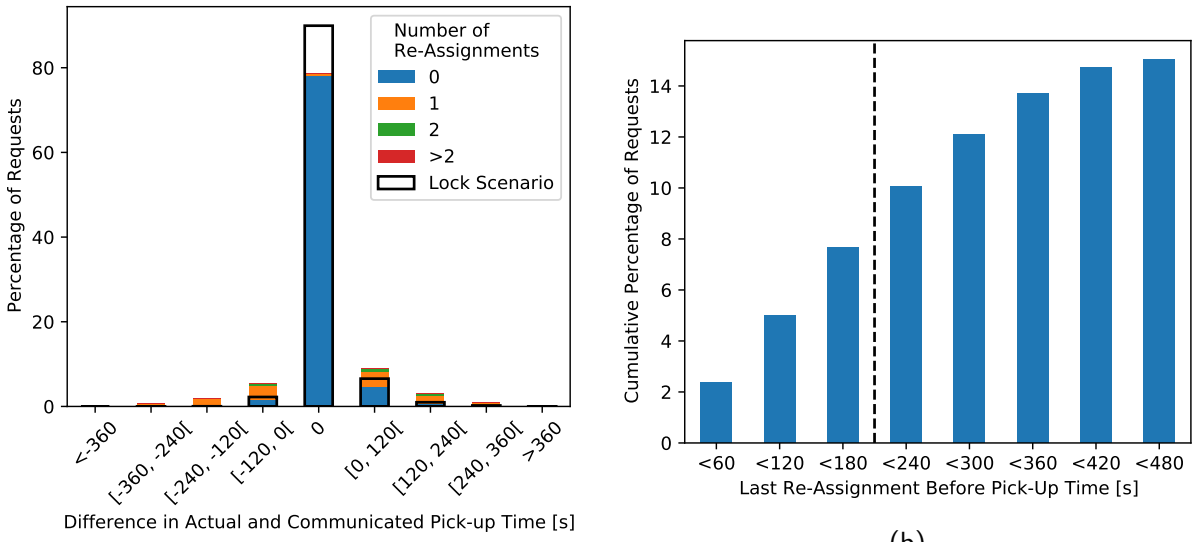
5.2.3 Assignment Reliability

The previous evaluation mainly focused on the operational aspects of the assignment algorithm and the value of allowing re-assignments. This section aims to evaluate the impacts of re-assignment from a customer point of view. As discussed in section 3.2.3, re-assignment might not only have positive implications as evaluated in the previous section but might also lead to unreliability in pick-up for customers as the assigned vehicle and the expected pick-up time can change after each optimization epoch.

Table 5.2 therefore evaluates the abundance of re-assignments for a customer if full re-assignment is possible for the assignment algorithm. Depending on the case study and demand penetration, 78.6 to 85.4% of customers do not experience re-assignment, i.e., the vehicle assigned initially also picked up the customer. Most of the remaining customers experience only a single re-assignment, while more than two re-assignments rarely happen, indicating a relatively stable matching process. In the Munich case study, re-assignments tend to be more probable. This might result from inhomogeneous travel times in the network that can vary on each network edge when new travel times are introduced, compared to the other two case studies where the same factor scales each edge travel time. This can lead to new shortest paths between stops, and therefore, new candidate schedules in a V2RB can emerge.

Further evaluation of the impact of re-assignment for customers is shown in Figure 5.7. Figure 5.7a shows the distribution of the difference in actual and initially communicated pick-up time. The distribution shows a dominant peak at zero, depicting customers who are picked up at the time that was initially communicated. This bar is dominated by customers who have not experienced any re-assignment. The average shift in pick-up time is only 4.1s, indicating that re-assignment does not tend to delay pick-ups. However, with a standard deviation of 61.0 seconds, re-assignment tends to distribute pick-ups fairly evenly between earlier and later times. Even in the scenario, which immediately locks a customer to its initially assigned vehicle (indicated by the bar with the black edge in Figure 5.7a), shifts in pick-up times compared to the initially communicated pick-up time can occur. These shifts may happen due to changes in travel times or the insertion of another customer before the scheduled pick-up. However, with a mean shift of 4.3 seconds and a standard deviation of 25.3 seconds, these effects are less significant than the variations caused by re-assignment.

From a customer point of view, not only the general shift in pick-up time from the initial



(a) (b)

Figure 5.7: Customer Effects of Re-Assignment. Chicago - 50% Demand Penetration - Fleet Size: 850.

assignment but also the timing of re-assignments relative to the actual pick-up time is relevant for the perceived reliability of the service. Short notice updates might be perceived as unfavorable or lead to customers missing their vehicles or showing up late, which could result in further operational deficits. Figure 5.7b therefore evaluates the timing of the last re-assignment before the actual pick-up of a customer. Of the 15% of customers who experience a re-assignment (compare Table 5.2), more than half of these re-assignments occur on a relatively short notice, which is here defined as a re-assignment earlier than 3 min before the pick-up. For almost 2.5% of the customers, the re-assignment even occurs less than a minute before the pick-up.

Evaluation of Reliability Strategies

As the results suggest, re-assignments can have significantly favorable effects on the operational level of the ARP service as an increased solution space allows assigning more efficient vehicle-customer assignments and schedules, but also brings the trade-off of reduced reliability and uncertainty for customer pick-ups. Based on the discussion of section 3.2.3, the impact of different methods is depicted in Figure 5.8 that are designed to reduce uncertainty for customers but still enable operational benefits from re-assignment for the operator. The three proposed methods are depicted in different colors. The corresponding control parameters are ordered to attach to the scenario with unconstrained re-assignment and the scenario without re-assignment. This is particularly evident in the primary operational objective of serving as many customers as possible, as there is a continuous increase in served requests with less constrained re-assignment (from left to right). Nevertheless, differences in other KPIs are observable when comparing different methods.

For the *Re-Assignment Time Window* method, after the initial assignment, customer pick-up time constraints are adjusted to a time window of Δ_{TW} around the communicated pick-up

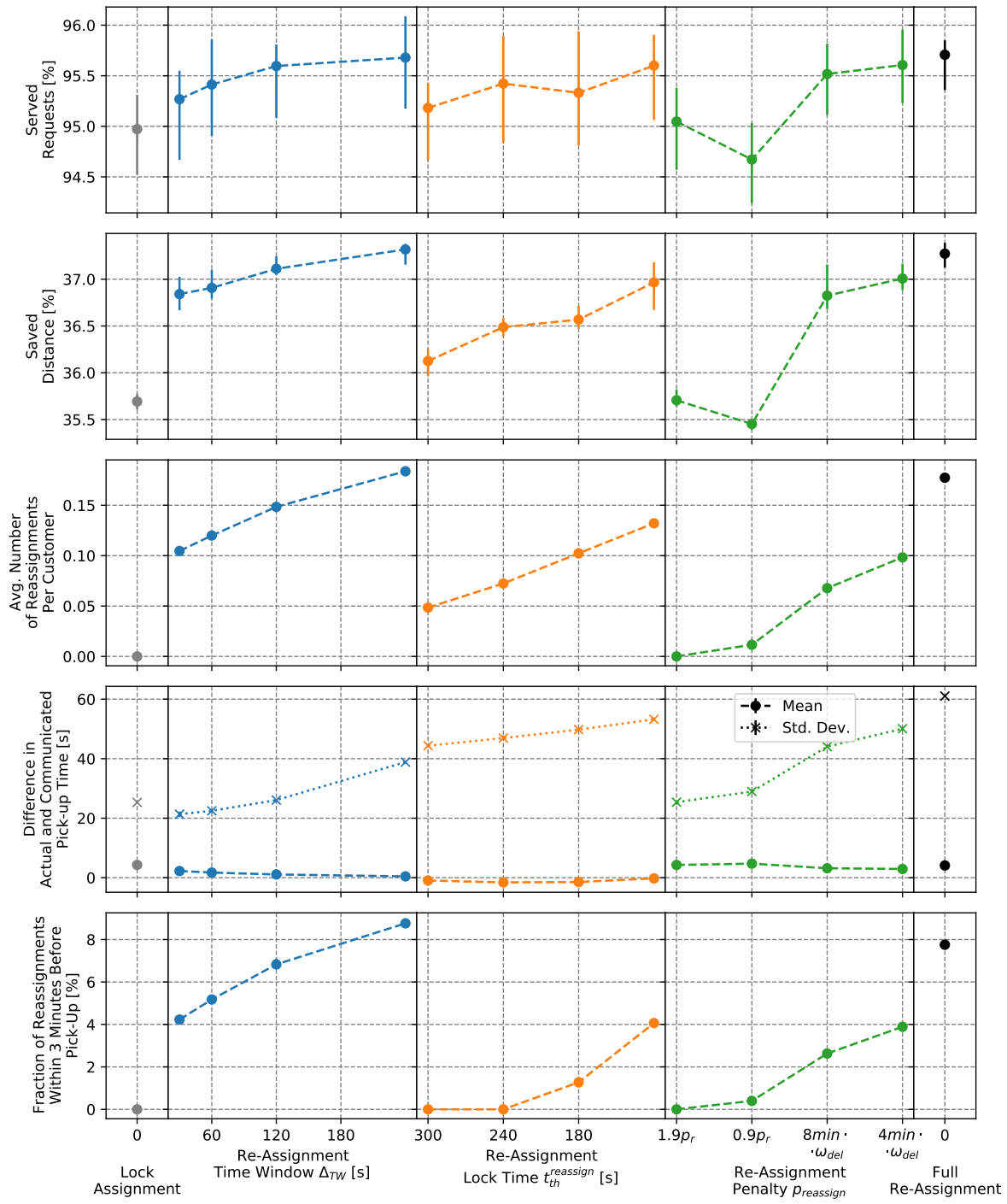


Figure 5.8: Comparison of methods to limit re-assignment for improved customer convenience. Chicago - 50% Demand Penetration - Fleet Size: 850.

time to prevent significant shifts in communicated and actual pick-up time. Re-assignments are still possible as long as the adjusted time constraints are not violated. In this scenario, the standard deviation in the difference in actual and communicated pick-up time tends to be

the smallest across all methods. For a time window of $\Delta_{TW} = 30$s, the standard deviation is even smaller than in the scenario that does not allow re-assignment at all because not only re-assignments are influenced but also the potential insertion of other customers into the schedule. Interestingly, there is only a limited trade-off with respect to operational efficiency: Served requests remain similar to the unconstrained re-assignment scenario, while saved distance is the highest across all methods. As a trade-off, this methodology shows the highest amount of re-assignments, and consequently also the highest amount of re-assignments on short notice.

In the *Re-Assignment Lock Time* method, the customer-vehicle assignment is locked once the scheduled pick-up time falls below the time horizon $t_{th}^{reassign}$, which would allow customers to track the vehicle on the map and anticipate pick-up. Consequently, this method shows lower amounts of re-assignments the higher the time horizon is set and a significantly lower amount of late re-assignments. For values of $t_{th}^{reassign}$ larger than 3 min, this method completely avoids re-assignments on short notice per design. From an operational perspective, this method shows a decrease in saved distance compared to the *Re-Assignment Time Window* (except for $t_{th}^{reassign} = 120$s), but still a higher saved distance compared to the scenario without re-assignment.

Finally, the goal of the *Re-Assignment Penalty* method is to avoid using hard constraints to limit re-assignments but rather let the optimization algorithm decide on the necessity of re-assignments. A penalty is introduced, representing a threshold in an improvement of the objective function that must be achieved by a re-assignment to be considered. The following four penalty values are tested:

1. $p_{reassign} = 1.9p_r$: In this case, re-assigning is penalized higher than serving an additional customer. In this case, a re-assignment is only considered if two additional customers can be served due to the re-assignment.

2. $p_{reassign} = 0.9p_r$: In this case, the penalty for re-assignment is still high compared to the secondary objective (minimizing driving distance and customer delay) but not as high as the assignment reward. Therefore, a re-assignment is considered if at least one additional customer can be served due to the re-assignment.

3. $p_{reassign} = 8\text{min} \cdot \omega_{del}$: Re-assignment is only possible if the (overall) objective improves by at least a value of 8 minutes worth of customer delay.

4. $p_{reassign} = 4\text{min} \cdot \omega_{del}$: Re-assignment is only possible if the (overall) objective improves by at least a value of 4 minutes worth of customer delay.

Results show that in the first two scenarios, where re-assignment requires serving at least one additional customer, the number of re-assignments nearly vanishes, and the operational KPIs, served requests and saved distance significantly degrade. This observation indicates that the value of re-assignment does merely originate from serving additional customers in the current optimization epoch but rather from finding more efficient schedules that enable serving more customers in the future. When $p_{reassign}$ is in the order of magnitude of customer delay, operational KPIs are on a similar level compared to the full re-assignment scenario. Nevertheless, the number of re-assignments and late re-assignments remains fairly low compared to the other methods. Even with a fairly small penalty of $p_{reassign} = 4\text{min} \cdot \omega_{del}$, the average number

of re-assignment per customer is reduced from 17.7% in the scenario with full re-assignment to 9.8%. This indicates that many re-assignments are not necessary to improve the operational KPIs significantly. Rather, a few well-placed re-assignments can significantly impact the service quality. The penalty method is beneficial in identifying these re-assignments and stabilizing vehicle-customer assignments.

Similar conclusions can be drawn from the Munich case study, which is shown in the appendix in Figure II.2.

5.3 Repositioning

This section presents the results of the presented repositioning method in chapter 3.3. First, scenarios and parameters specific to this section are described, followed by the presentation of simulation results.

5.3.1 Scenarios and Parameters

Algorithm	Short Description	Reference Section
Sampling	The algorithm proposed in this thesis: By sampling requests, fleet state is progressed into the future. Assignment of repositioning trips to fill supply shortages.	3.3
Hor	Supply estimation proportional to expected demand in zone. Single horizon; Scaling factor to account for pooling.	3.3.5 (Horizon-based Repositioning)
QT	Repositioning of idle vehicles to stabilize queues in zones. Single horizon; Scaling factor to account for pooling.	3.3.5 (Queuing Theoretical Repositioning)
React	Reactive repositioning algorithm. Vehicles are sent to locations of unserved requests.	3.3.5 (Reactive Repositioning)
No Repo	No Repositioning is applied.	

Table 5.3: Summary of rebalancing algorithms for this section.

The parameters for this section are summarized in Table 4.2 while their base values are still valid as long as they are not explicitly mentioned otherwise.

In the following, the impact of repositioning is first evaluated and discussed in detail. In the next step, the proposed algorithm is compared to the benchmark algorithms. A summary of these algorithms is provided in Table 5.3. Additionally, the impact of forecast accuracy is evaluated. Two different methods are tested to forecast future trips:

1. Perfect Forecast: From the input request, set the number of requests between zone i and j in forecast interval T is used as the Poisson rate $\lambda_{i,j}^{T}$.

2. Myopic Forecast: The number of trip requests in the simulation during the past interval $\{t - \delta_T, t\}$ between zone i and j is used as Poisson rate $\lambda_{i,j}^{T}$ at time t for each T.

It can be assumed that more sophisticated forecast algorithms based on historic trip data should perform at least as well as the myopic forecast. In contrast, the perfect forecast acts as an upper bound. An evaluation of operational parameters is conducted, including an assessment of different repositioning frequencies, various repositioning zone sizes (see Figure 4.5), and whether repositioning trips should be locked. Finally, the influence of the hyperparameters, such as forecast horizon $\mathfrak{H}$, number of forecast samples N_S, and the weight of assigning future repositioning trips γ, is evaluated.

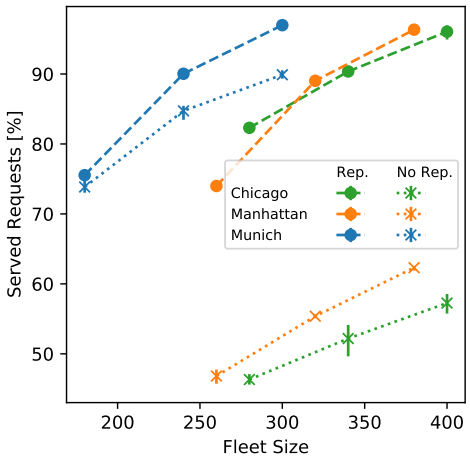
5.3.2 Impact of Repositioning

Figure 5.9 shows different evaluations comparing a service that applies the proposed rebalancing algorithm and a service without any rebalancing. Figure 5.9a shows the number of served requests for three different fleet sizes for each case studies. When repositioning is applied, the fleet sizes of 340, 240, and 320 serve close to 90% of the demand in the Chicago, Munich, and Manhattan case studies, respectively. When the fleet size increases to 400, 300, and 275 vehicles, the service rate accordingly increases to roughly 95% for all case studies. Without repositioning, vastly different results are observable: An enormous drop in service rate is notable in the Chicago and Manhattan case study. The 340 and 320 vehicles that serve 90% of the demand with repositioning only serve 52% and 55% of the demand without repositioning for Chicago and Manhattan, respectively. This drop results from vehicles ending up in network regions with low demand. Vehicles in these regions remain idle until a new customer requests a trip. In the Munich case study, the drop is significantly less pronounced. Even without repositioning, 85% of the customers can still be served when deploying 240 vehicles, compared to 90% with repositioning. This result can be explained by the more homogeneous demand distribution in the Munich case study compared to the other two case studies, which reduces the evolution of demand-supply imbalances.

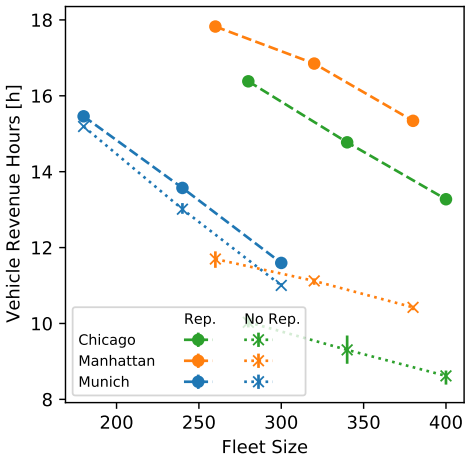
Figure 5.9b shows the average vehicle revenue hours, i.e., the absolute time interval during the day fleet vehicles carry customers and produce revenue for the operator. Vehicles ending up in regions with low demand remain idle and do not produce revenue. For scenarios with rebalancing vehicles, revenue is produced for 14.8 and 16.8 hours of the day in the Chicago and Manhattan case studies with 340 and 320 vehicles, respectively. Similar to the fraction of served requests, this quantity drops significantly to 9.3 hours and 11.1 hours without rebalancing, indicating that a significant fraction of vehicles remain idle, although demand is not fully served. Again, this effect is less pronounced in the Munich case study but still observable.

To get a deeper understanding of the observations, Figure 5.9c and Figure 5.9d show the temporal evolution of fleet states during the day for a service with 340 vehicles for the Chicago case study without and with rebalancing, respectively. Without repositioning, a significant fraction of the fleet remains idle the whole day, resulting in low vehicle revenue hours. By repositioning, vehicles' idle time can be reduced significantly, leading to almost full utilization except for times with low demand at night and noon. Figure 5.9e and Figure 5.9f show the spatial distribution of unserved requests on a logarithmic scale for the same scenarios. In both scenarios, most requests are rejected close to the city center. Nevertheless, the absolute number is much lower when rebalancing is applied. Black circles indicate the time vehicles spend idle in the corresponding zone. Without rebalancing, many vehicles end up idle at the O'Hare airport in the northwest corner of the operating area. They are therefore not available to serve customers in the city center. With rebalancing, on the other hand, idle times are reduced overall, and vehicles tend to be located in areas with high demand, where they are needed to serve demand.

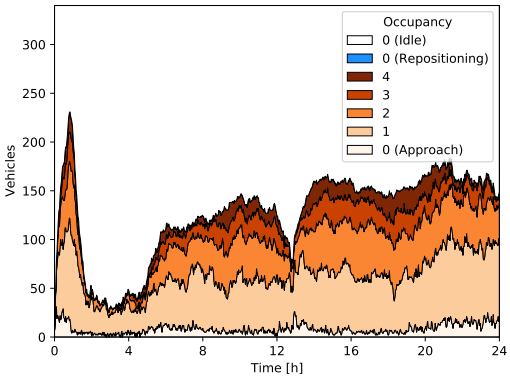
Similar spatial and temporal evaluations for the Munich and Manhattan case study can be found in the appendix in Figures III.4 and III.3, respectively.



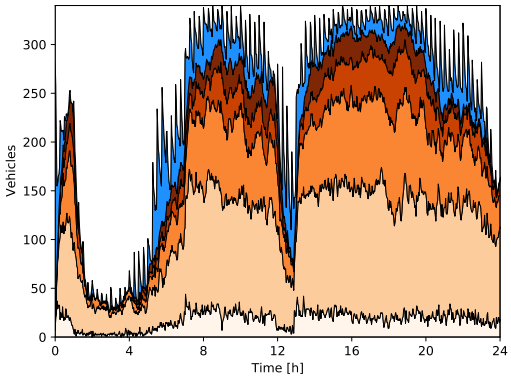
(a) Served Requests for different Fleet Sizes and Case Studies.



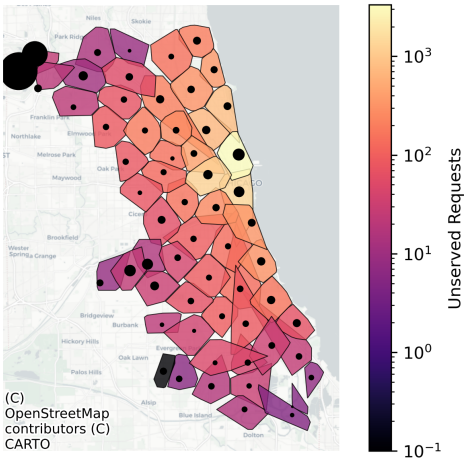
(b) VRH for different Fleet Sizes and Case Studies.



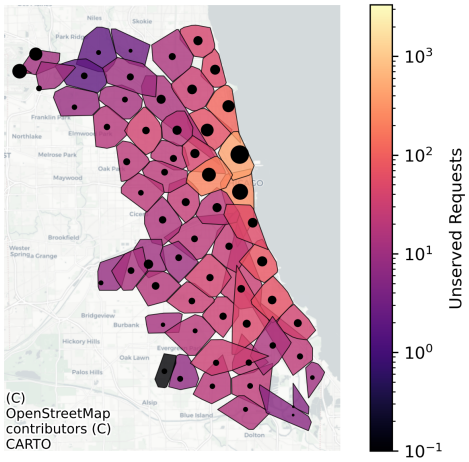
(c) Without rebalancing: Temporal vehicle states of the simulation period.



(d) With rebalancing: Temporal vehicle states of the simulation period.



(e) Without rebalancing: Spatial distribution of unserved requests. The size of black circles indicates idle times of fleet vehicles in zones.



(f) With rebalancing: Spatial distribution of unserved requests. The size of black circles indicates idle times of fleet vehicles in zones.

Figure 5.9: Comparison of results with and without rebalancing. Sub-figures 5.9c- 5.9f show the Chicago case study with 340 vehicles.

5.3.3 Comparison of Rebalancing Algorithms

Figure 5.10 compares different rebalancing algorithms and forecast methods for the Chicago, Munich, and Manhattan case studies.

The first row (Figure 5.10a, Figure 5.10b, and Figure 5.10c) shows the difference in served requests compared to the scenario that applies the *React*-method (compare Figure 5.9a for absolute values). Regardless of the repositioning method or forecast method applied, the comparison to Figure 5.9a reveals that the ARP service always benefits from repositioning. With respect to *Sampling*-method proposed in this thesis, benefits are especially notable in the Chicago case study, where served requests can be increased by up to 2% compared to all other repositioning algorithms. Next closest is the *Hor*-method, which performs especially well in the Manhattan case study in terms of overall served requests, where even slightly more customers are served compared to the *Sampling*-method. For the Munich case study, the *Hor*-method and *Sampling*-method show a similar service rate. The QT-method shows the lowest service rate compared to the *Sampling*- and *Hor*-method. Nevertheless, except for small fleet sizes, the QT-method still outperforms the *React*-method in terms of served requests, indicating the benefits of predictive repositioning in contrast to reacting to unserved demand. Comparing the forecast methods, the *Perfect* forecast shows — as expected — the best results regarding served requests, at least for the *Sampling*-method. Nevertheless, the gain in performance from different forecast methods is significantly less pronounced compared to the applied repositioning algorithm.

The second row (Figure 5.10d, Figure 5.10e, and Figure 5.10f) shows the saved distance for the different algorithms. Although serving most customers in most scenarios, the *Sampling*-method shows a significantly higher saved distance compared to the other predictive repositioning methods *Hor* and QT. Only for the Manhattan case study does the QT-method show a slightly higher saved distance when a large fleet size is applied. While performing well in serving customers, the *Hor*-method shows the lowest saved distance across all scenarios. This indicates an aggressive repositioning strategy resulting in numerous empty vehicle trips degrading this KPI. The difference in saved distance to other methods is especially notable in the Chicago case study, where long distances must be covered by repositioning due to the large operating area. As repositioning trips are only assigned when requests remain unserved in the *React*-method, this method can be interpreted as relatively conservative, assigning only necessary repositioning trips. Therefore, the *React*-method shows the highest saved distance across all scenarios but with the trade-off of serving the least number of requests.

Finally, the third row (Figure 5.10g, Figure 5.10h, and Figure 5.10i) shows the average waiting time for customers evaluating the placement of vehicles after repositioning. In this KPI, too, the *Sampling*-method demonstrates strong performance across all scenarios. Only the *Hor*-method shows slightly lower average customer waiting times, which comes with the discussed trade-off of extensive repositioning. The benefits of predictive repositioning are especially notable when compared to the *React*-method, which shows the highest average waiting times across all scenarios. In contrast to the *React*-method, the predictive algorithms can anticipate future demand and place vehicles in areas with expected demand before the trips are requested. In this case, the chances increase that vehicles are already close to the customer when the request is made, reducing the waiting time for the customer. This effect is especially notable in scenarios with large fleet sizes, where the chance of finding repositioned

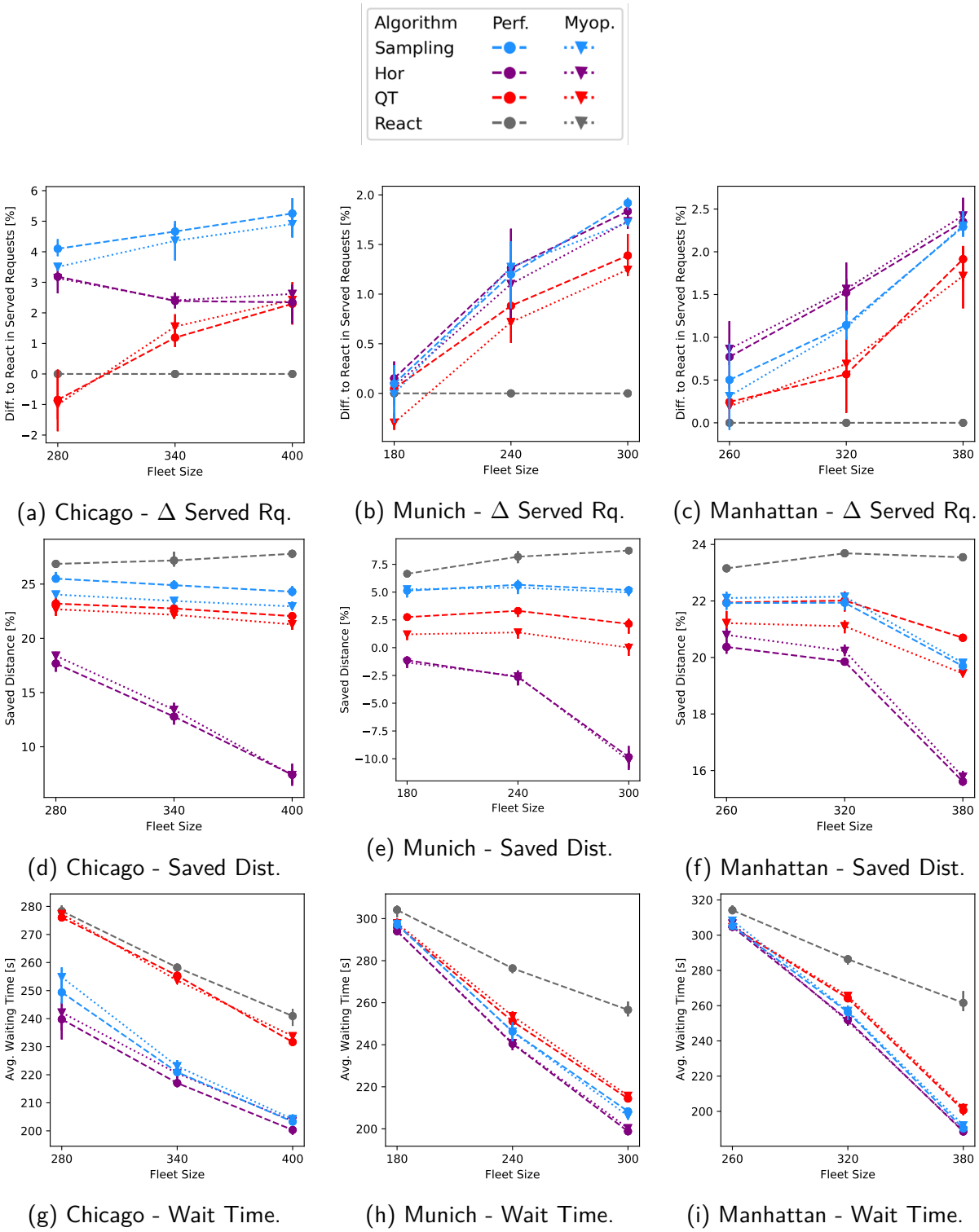
idle vehicles in close vicinity to the customer is higher.

The previous analysis showed that the proposed *Sampling*-algorithm shows excellent results regarding served requests, saved distance, and average waiting time. Only the *Hor*-algorithm can compete in terms of served requests (in the Munich and Manhattan case study) and average waiting time, but at the cost of an increased number of repositioning trips. To better understand the difference between both algorithms, Figure 5.11 shows the difference in the spatial distribution of unserved requests and vehicle idle times for the Chicago, Munich, and Manhattan case studies. The color scale indicates the difference in unserved requests between both algorithms. High values reflect areas where the ARP-service with applied *Sampling*-algorithm rejects more requests compared when the *Hor*-algorithm is used. It is striking that the *Hor*-algorithm serves more customers in the high-demand areas in all case studies (see Figure 4.3 for spatial demand distribution). Nevertheless, in most other areas of the operating area, the *Sampling*-algorithm shows higher service rates. Additionally, the evaluation of vehicle idle times indicated by the markers shows that the *Hor*-algorithm strongly prioritizes sending vehicles to high-demand areas, while the *Sampling*-algorithm tends to distribute vehicles more evenly across the operating area. This results from the formulation of the *Hor*-algorithm, which assigns repositioning trips proportional to the number of requests in a zone. The introduced scaling factor reduces this overestimation of the required supply in high-demand areas but tends to underestimate the necessary supply in low-demand areas. The *Sampling*-algorithm on the other hand assigns repositioning trips based on sampled trips, that includes the possibility of multiple customers being transported by the same vehicle. As the probability of finding shared trips is higher in areas with high demand, the *Sampling*-algorithm does not overestimate the required supply in those areas, enabling sending excess vehicles to less profitable areas.

5.3.4 Operational Parameters

The evaluation of this section focuses on the impact of operational parameters on the performance of the ARP service, namely the applied zone size, the repositioning interval Δ_R, and whether repositioning trips should be locked or can be aborted if a nearby request is to be served. The results are shown in Figure 5.12.

The Figures 5.12a to 5.12d show the impact for different KPIs for the Chicago case study with a fleet size of 340 vehicles. The evaluation of served requests in Figure 5.12a shows that the impact of all parameters is minor compared to stochastic variations within the scenarios. Nevertheless, some trends can be observed: The number of served requests increases if the repositioning interval Δ_R is decreased. Thereby, vehicles that end up in areas with low demand can be sent faster to areas where they are needed. If the repositioning trips are not locked, the impact of the zone size is negligible. In general, when repositioning trips are made more frequently, such as with $\Delta_R = 5$min and $\Delta_R = 15$min, allowing vehicles to interrupt their repositioning trips to serve an incoming request tends to be a more effective strategy in terms of the number of requests served. In this case, flexibility in the assignment is more beneficial than the long-term planning of repositioning trips. If repositioning is made frequently enough, new repositioning trips in the upcoming epoch can replace potentially aborted trips. For a repositioning interval of $\Delta_R = 5$min and locked trips, the impact of the zone size is more pronounced. The smallest zone size of $t^Z_{max} = 4$min shows the best performance in terms of

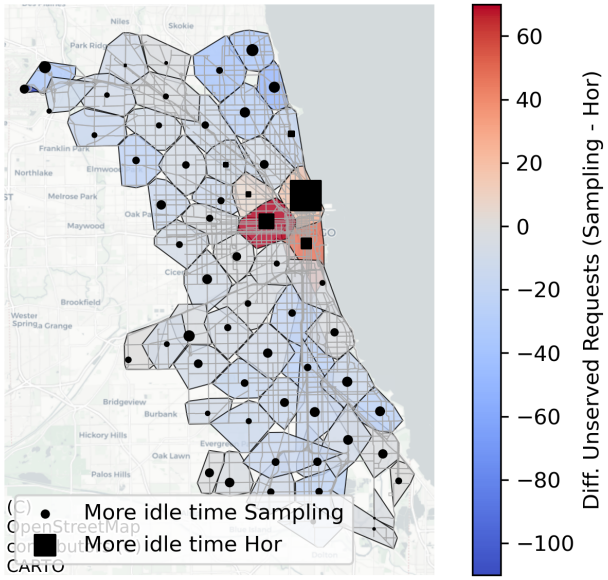


(a) Chicago - Δ Served Rq. (b) Munich - Δ Served Rq. (c) Manhattan - Δ Served Rq.

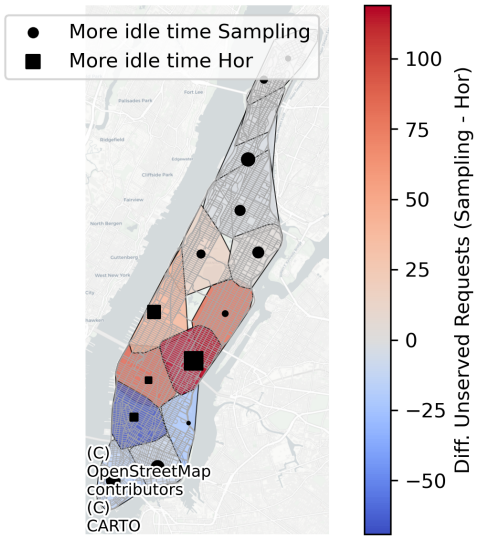
(d) Chicago - Saved Dist. (e) Munich - Saved Dist. (f) Manhattan - Saved Dist.

(g) Chicago - Wait Time. (h) Munich - Wait Time. (i) Manhattan - Wait Time.

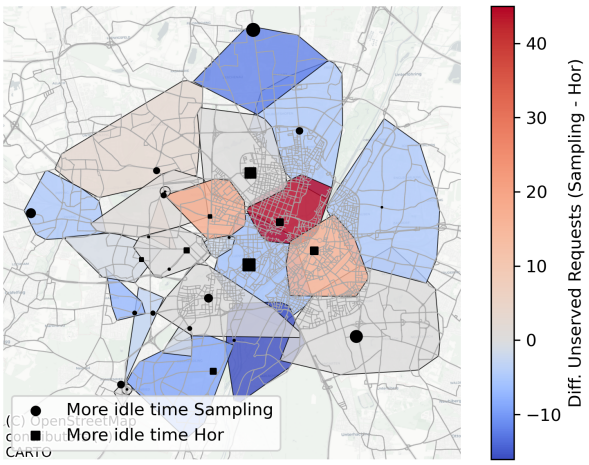
Figure 5.10: Comparison of algorithms and forecast methods. Columns show different Case Studies. Left Column: Chicago, Mid Column: Munich, Right Column: Manhattan. As the *React* algorithm does not use a forecast, only one line is shown in each plot.



(a) Chicago.



(b) Manhattan.



(c) Munich.

Figure 5.11: Spatial differences in unserved requests and vehicle idle times for the sampling and *Hor* algorithm.

served requests, as vehicles can be positioned more accurately to the demand (perfect demand forecast within the zones and temporal intervals is assumed). As repositioning trips are locked, vehicles always end up in the centroid of the corresponding zone, resulting in potentially longer approach times to the actual demand. If trips are not locked, repositioning vehicles passing near incoming demand become a valuable alternative for serving requests.

Figure 5.12b and Figure 5.12c show the saved distance and repositioning VKT for the Chicago case study. It can be observed that the change in saved distance is directly related to the change in repositioning VKT. Unsurprisingly, the shorter the repositioning interval Δ_R, the higher the repositioning VKT and the lower the saved distance. While repositioning VKT is slightly higher for smaller zone sizes, the impact is nearly negligible. The effect of the locking of repositioning trips is more pronounced. Locking of repositioning trips leads to higher repositioning VKT of up to 4% and lower saved distance.

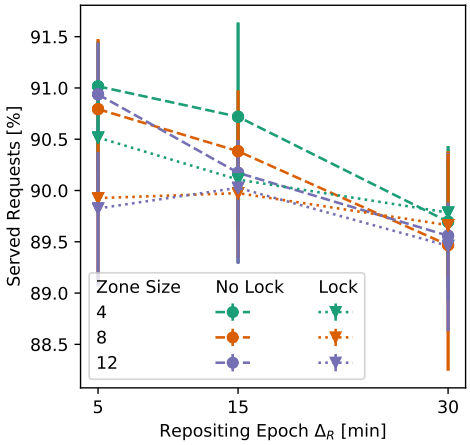
With respect to the average waiting time, Figure 5.12d shows that zone sizes have a stronger influence on this KPI as vehicles can be positioned closer to the customer. As the options for vehicles for new requests increase, the average waiting time decreases if vehicles are allowed to abort their repositioning trips. In this case, a more frequent repositioning interval also shortens the average waiting time. Interestingly, this is not the case if repositioning trips are locked. This might result from too many vehicles being repositioned and unavailable to serve incoming requests.

Finally, Figures 5.12e and 5.12f show the impact of the operational parameters on the served requests for the Munich and Manhattan case studies, respectively. The results are similar to the Chicago case study. Nevertheless, for the Manhattan case study, locking of repositioning trips seems more beneficial in terms of served requests. Due to the high trip density, vehicles might be too frequently unassigned from their initially planned repositioning trip, leading to higher undersupply in profitable regions (midtown). Additionally, as the operating area is relatively small, locking of repositioning trips does not decrease the effectively available vehicle supply too much.

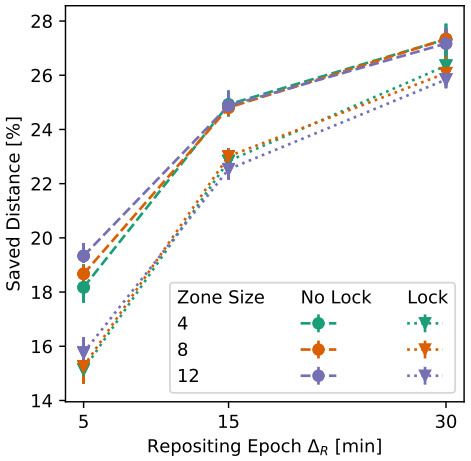
Hyperparameters

Figure 5.13 shows the impact of the hyperparameters forecast horizon $\mathfrak{H}$, number of forecast samples N_S, and the weight of future repositioning trips γ on the performance of the ARP service.

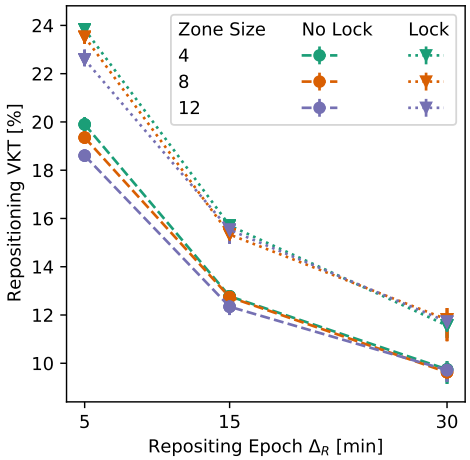
Figures 5.13a to 5.13c compare the effects of different forecast horizons $\mathfrak{H}$ on the served requests, repositioning VKT, and computational time, respectively. It can be observed that the forecast horizon has a significant impact on the performance of the ARP service. For the served requests, the performance increases with the forecast horizon up to $\mathfrak{H} = 45$min, followed by a saturation effect. For Munich, the saturation is already observable for $\mathfrak{H} = 30$min. Due to time constraints in the formulation, the repositioning algorithm can only assign repositioning trips that can be completed within the forecast horizon. Consequently, if the forecast horizon is too short, the algorithm cannot cover the whole operating area, leading to decreased performance. This is especially notable in the Chicago case study, where the operating area is large. For Munich, the operating area is smaller, which shortens the required forecast horizon. Additionally, previous analysis has shown that the impact of repositioning is less significant in the Munich case study, resulting in a less pronounced saturation effect. A



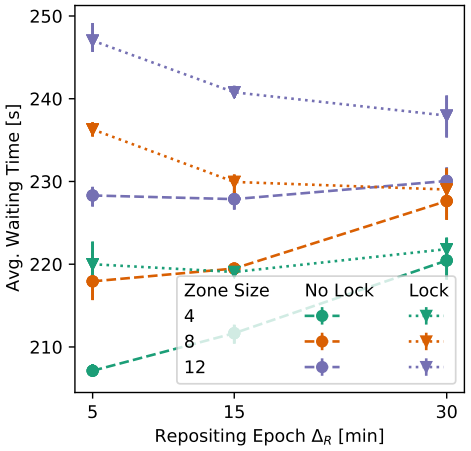
(a) Chicago - Served Requests.



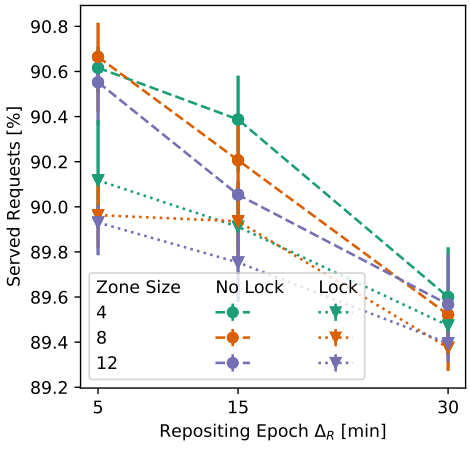
(b) Chicago - Saved Distance.



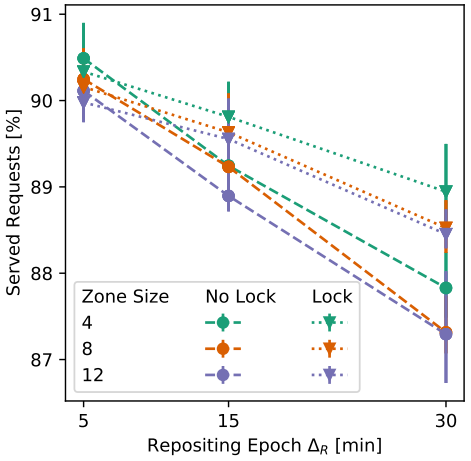
(c) Chicago - Repositioning VKT.



(d) Chicago - Avg. Waiting Time.



(e) Munich - Served Requests.



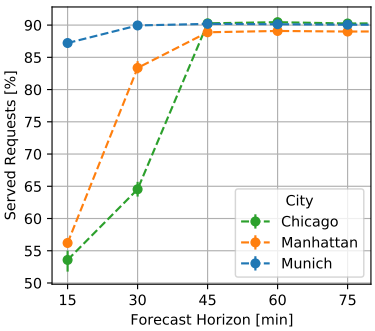
(f) Manhattan - Served Requests.

Figure 5.12: Impact of repositioning frequency Δ_R, repositioning zone size and locking of repositioning trips.

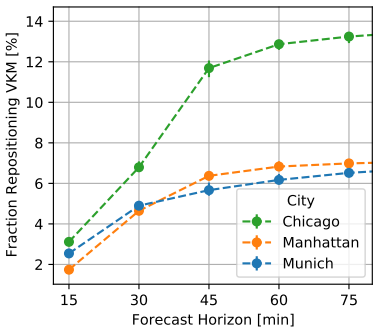
similar trend can be observed for the repositioning VKT. Nevertheless, the saturation effect appears at higher forecast horizons. With longer forecast horizons, longer repositioning trips can be assigned, leading to a higher repositioning VKT. This trend seems to prevail, even if the served requests saturate. Resulting from the large operating area and inhomogeneous demand, the highest fraction of repositioning VKT is observable in the Chicago case study. The computational time increases linearly with the forecast horizon. A more detailed analysis of the computational time reveals that the sampling process is by far the most time-consuming part of the algorithm. Only a small fraction of the computational time is spent on solving the optimization problem of assigning repositioning trips. Therefore, a nearly linear increase in computational time with the forecast horizon is observable because the number of sampled trips increases approximately linearly with the forecast horizon. As the request density is highest for Manhattan, the computational time is also highest for this case study.

Figures 5.13d to 5.13f show the impact of the number of forecast samples N_S on the performance of the ARP service. The impact of N_S on served requests and repositioning VKT is considerably smaller than the forecast horizon. No significant trend can be observed within the error bars for Chicago and Munich. Manhattan shows a slight increase in served requests. Although still small, an increase in the repositioning VKT is observable for the Chicago and Munich case studies. This shows an overall stable performance with respect to stochastic variations in the forecast samples, likely due to the large number of requests sampled (compare Figure 5.14d: In a forecast bin of 15min, up to 3,200 requests are sampled for the Manhattan case study with 20% demand penetration). Similar to the forecast horizon, the computational time increases linearly with the number of forecast samples because the number of sampled trips increases linearly, too. The average computational time varies between around 40s, if only one sample is used, and 220s on average if five samples are used for the Manhattan case study. However, the computational time is still within the 900s repositioning interval, yielding real-time applicability. For further speed-up, the sampling process could easily be parallelized if different samples are calculated in parallel. Additionally, a C++ implementation of the sampling process could significantly reduce computational time further.

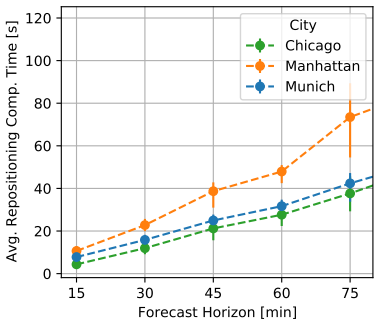
Finally, Figures 5.13g and 5.13h show the impact of the weight of future repositioning trips γ. Again, only minor impacts of this hyperparameter can be observed. No significant trend can be observed for the requests served. For the repositioning VKT, a slight decrease is observable when the parameter approaches $\gamma = 1$. In this case, assigning repositioning trips for future epochs is weighted equally to the current epoch. Nevertheless, as future repositioning trips are not assigned in the current epoch, a decrease in repositioning trips can be observed.



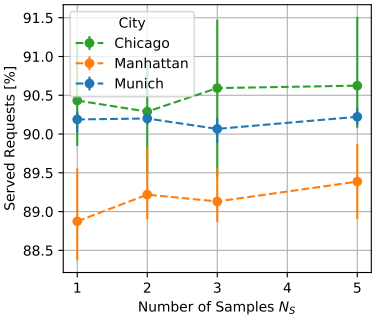
(a) Horizon $\mathfrak{H}$: Served Requests.



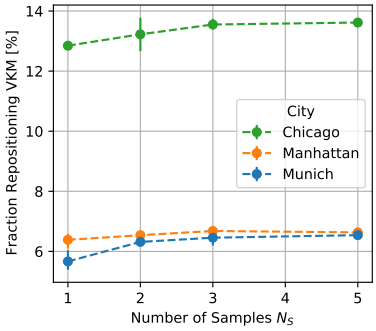
(b) Horizon $\mathfrak{H}$: Repo. VKT.



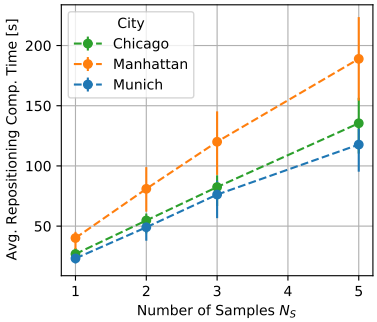
(c) Horizon $\mathfrak{H}$: Comp. Time.



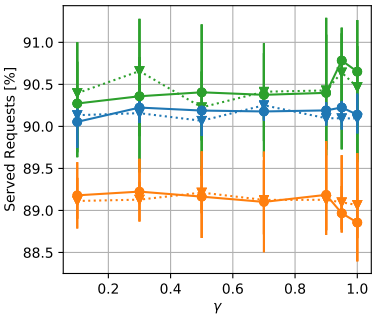
(d) Number Sample N_S: Served Requests.



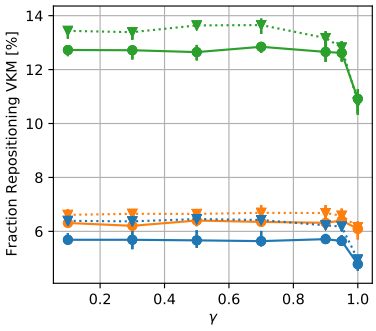
(e) Number Samples N_S: Repo. VKT.



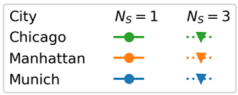
(f) Number Samples N_S: Comp. Time.



(g) γ - Served Requests.



(h) γ - Repo. VKT.



(i) γ - Legend.

Figure 5.13: Impact of Hyperparameters.

5.4 Reservation

This section presents the results of the impact of reservations on the ARP service. First, scenarios and parameters specific to the reservation case study are presented, followed by simulation results.

5.4.1 Scenarios and Parameters

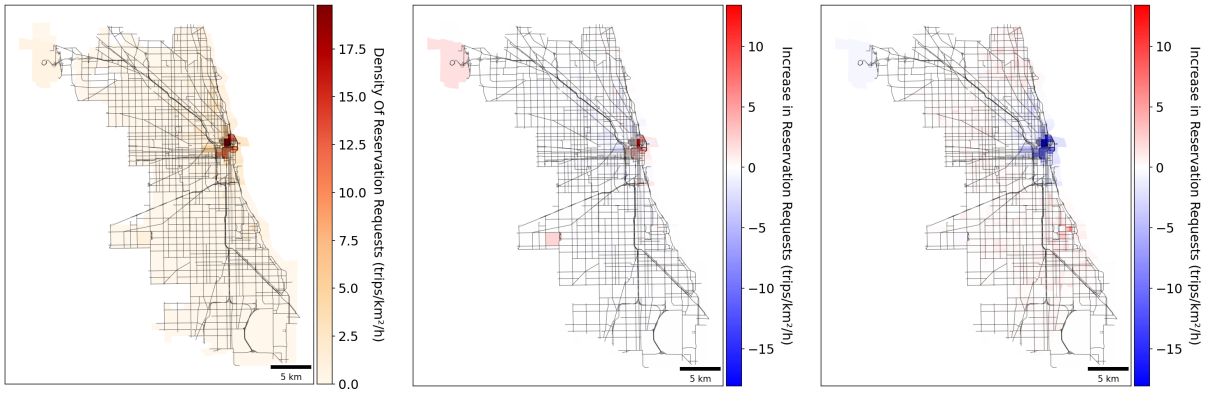
Module	Algorithm	Short Description	Reference Section
Creation of Long-Term Schedules	***NCRH***	Non-Causal Rolling Horizon: This method is used to evaluate long-term-only reservation scenarios.	3.4.3
	CBO	Consecutive Batch Optimization: This method is used to evaluate continuously incoming reservations.	3.4.3
Assignment	***OPT***	Proposed in this thesis. Full re-assignment of reservation and on-demand customers within short-term horizon.	3.2 + 3.4.2
	IH	Insertion Heuristic for on-demand customers. No re-assignment of reservation customers.	3.2.6 + 3.4.2
Repositioning	***Sampling***	Sampling method with adoptions for reservations proposed in this thesis.	3.3 + 3.4.4
	React	Repositioning to locations of unserved requests with adoptions for reservations.	3.3.5 + 3.4.4

Table 5.4: Summary of algorithms to evaluate reservations that are used in this section. Bold algorithms are used as base case.

The same parameters and input data as described in section 4.3 are used in this section. The different algorithms, that have been described in section 3.4 and are evaluated in this section are summarized in Table 5.4.

In contrast to previous simulations, reservation requests have to be defined. Therefore, the set of requests is split into on-demand and pre-booking customers. Different shares S of 0%, 10%, 25%, 50%, 75% and 100% pre-booking customers are tested in the simulations. To evaluate the impact of spatio-temporal variability on the effect of pre-booking customers, pre-booking requests are drawn from three different distributions of the overall set of requests:

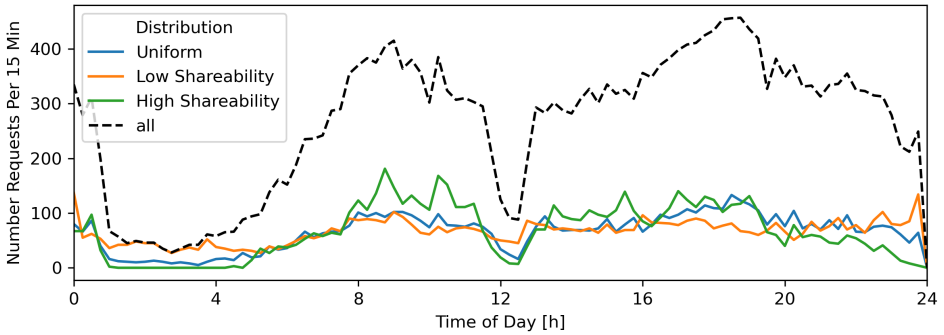
- Uniform: Each request is added to the set of pre-booking customers with a probability of S.
- Low / High Shareability: Using this distribution, those customers pre-book the service that have a low / high probability of finding shareable rides. Therefore, the shareability graph from [Santi et al., 2014] of the whole data set is created first: Two requests are connected if any hypothetical feasible schedule can be found where both requests share the ride for a fraction of the trip. Let R_s be the list of requests selected by the sub-sampling procedure, sorted by the number of connections found in the shareability graph in ascending order. Then, the first / last S requests are added to the set of pre-booking customers.



(a) Uniform Distribution: Reservation Request Density.

(b) High Shareability Distribution. Increase in Reservation Request Density compared to Uniform Distribution.

(c) Low Shareability Distribution. Increase in Reservation Request Density compared to Uniform Distribution.



(d) Temporal distribution of reservation times of pre-booking requests for the different generating distributions in 15 min bins.

Figure 5.14: Spatial and temporal distributions of pre-booking customers when applying uniform, low shareability, and high shareability on the overall request set for the Chicago Case Study. A share $S = 25\%$ reservation requests is used for all cases.

Figure 5.14 shows spatial and temporal differences for the Chicago case study between these distributions and the idea behind selecting these distributions to create pre-booking customers. Figure 5.14a shows the spatial distribution of pre-booking request origins of the uniform distribution revealing high request densities in the city center and the airport as discussed in previous sections. Figure 5.14b shows the spatial increase (and decrease) in pre-booking requests when they are drawn from the high shareability distribution. In this scenario, more customers are pre-booking the service in areas with high request densities due to the correlation between shareability and request density. This is especially visible in the city center and the airport for the Chicago case study. In this scenario, customers tend to pre-book their rides in areas with high competition for vehicles with other customers. On the contrary, Figure 5.14b shows the change in pre-booking request density when the low shareability distribution is used. In this case, pre-booking customers are distributed across the operating area. In this scenario,

customers pre-book rides in areas of low request densities to ensure their trip is in case vehicle supply is concentrated in high-demand areas. Figure 5.14d shows the temporal distribution of pre-booking requests for the different generating distributions. While the uniform pre-booking distribution resembles a scaled version of the overall request distribution, a higher fraction of customers pre-books the services in times of low demand for the low shareability distribution, which is especially visible during nighttimes from 0 to 6 a.m. Contrarily, during high demand, i.e., between 6 and 11 p.m., the fraction of pre-booking customers is higher for the high shareability distribution.

Similar spatial and temporal evaluations for the Manhattan and Munich case study can be found in the appendix in IV.2 and Figures IV.1, respectively.

Next to the spatial distribution, the temporal distribution of pre-booking customers is also of interest. Therefore, two different temporal distributions are tested:

- Long-term only: It is assumed that all pre-booking customers are known from the beginning of the simulation (e.g., by making a reservation the day ahead). Consequently, given an on-demand request from the original data set, that is converted to a pre-booking request, the request time of the pre-booking customer is set to 0s while its earliest pick-up time is set to the original request time.

- Dynamically incoming reservations: In this scenario, pre-booking customers continuously arrive during the simulation. Again, the earliest pick-up time is set to the original request time of the on-demand request to be converted. The pre-booking time, i.e., the difference between the earliest pick-up time and request time, is drawn from an exponential distribution with mean $\tau_{res} = \{900s, 3600s, 7200s\}$.

Three input files with different random seeds of the sub-sampling process are created and used for each simulation presented. A uniform spatial and long-term only temporal distribution is used in the baseline scenario.

This section aims to evaluate the impact of reservations on the ARP service. Therefore, the KPI "Value of Reservation" VoR is defined similarly to the concept of "Value Of Information" introduced by [Wen et al., 2019]. It represents the relative improvement in the overall objective value when reservations are applied compared to an on-demand-only service. The VoR is defined as

$$VoR = \frac{O_{pre} - O_{\overline{pre}}}{O_{\overline{pre}}}, \tag{5.31}$$

with O_{pre} and $O_{\overline{pre}}$ as the overall objective value with and without reservation, respectively. The overall objective $O(sc)$ of scenario sc is defined accordingly to the objective of a schedule (Equation 3.9):

$$O(sc) = -\sum_{r \in R_{sc}} p_r + \omega_{del} \sum_{r \in R_{sc}} t_r^{do} - t_r + \omega_{dis} \sum_{v \in V} d_v(sc) \ , \tag{5.32}$$

with R_{sc} the set of served requests, and $d_v(sc)$ the distance driven by vehicle v in this scenario.

5.4.2 Impact of Long-Term Reservations

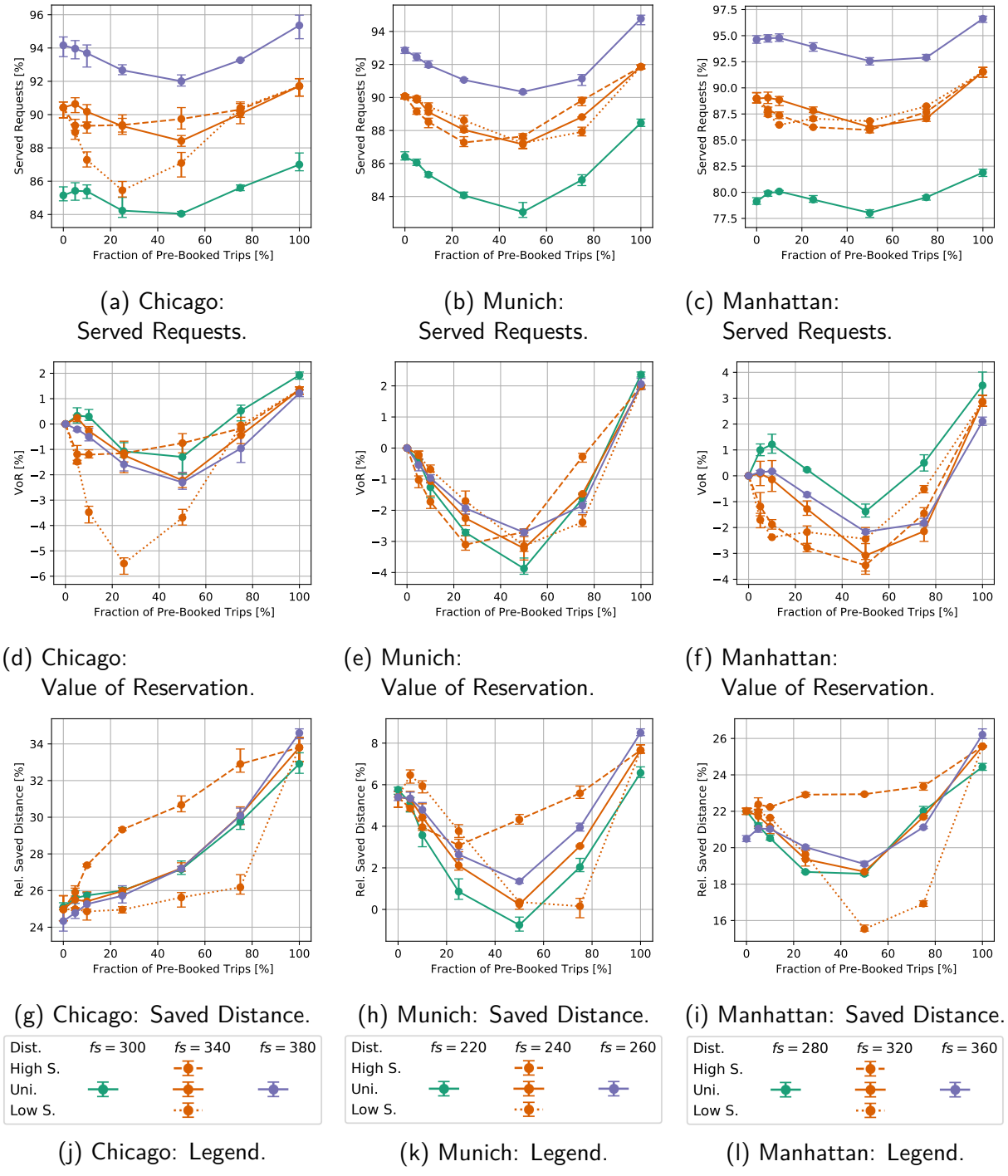
Figure 5.15 shows the impact of long-term reservations on the ARP service for the Chicago, Munich, and Manhattan case studies for different pre-booking fractions and fleet sizes. The different distributions of pre-booking customers are also shown for the medium fleet sizes.

The first row (Figure 5.15a, Figure 5.15b, and Figure 5.15c) shows the overall number of served requests. As a general trend, the number of served requests slightly decreases with the share of pre-booking customers until a fraction of around 50%. This effect is likely due to the increasing number of constraints that must be fulfilled to ensure service for pre-booking customers. For the Chicago and Manhattan case study, even a slight increase in the overall number of served requests can be observed until a pre-booking fraction of 10%. In this regime, the number of constraints for pre-booking customers is relatively low, and the system can benefit from the additional information about future demand. For high pre-booking fractions, the number of served requests recovers and exceeds the number of served requests in the case of an on-demand-only service. In this regime, the service performance becomes dominated by the solution of the long-term schedules, which can exploit all information about pre-booking demand to optimize the vehicle schedules. With respect to different pre-booking distributions, the uniform distribution (i.e., the distribution with the highest correlation between pre-booking and on-demand requests) shows the highest number of served requests for all pre-booking fractions in most cases for pre-booking fractions below 50%. In this scenario, vehicle schedules to serve pre-booking customers also benefit on-demand customers, as those vehicles get distributed proportionally to the on-demand requests. On the contrary, the low shareability distribution leads to a substantial decrease of up to 4% in the overall number of served requests for the Chicago case study. In this huge operating area, the low shareability distribution leads to a concentration of vehicles in areas of low demand, hindering them from efficient service in high-demand areas like the city center or trips from/to the airport.

The second row (Figure 5.15d, Figure 5.15e, and Figure 5.15f) shows the value of reservation VoR. As the overall objective is dominated by the number of served requests, the value of reservation shows a similar trend. In most cases, the value of reservation is negative for low pre-booking fractions, indicating a degraded service performance. Positive values can be observed for either low or high pre-booking fractions. The highest positive values of around 1% can be observed for smaller fleet sizes. Nevertheless, especially for the Chicago case study with the low shareability distribution (i.e., trips made to/from the city's outer areas), too many pre-booked trips can degrade the service by up to 5%.

The third row (Figure 5.15g, Figure 5.15h, and Figure 5.15i) shows the saved distance for the different scenarios. This KPI shows different trends across the case studies. For the Manhattan and Munich case study, the saved distance tends to decrease with the share of pre-booking customers. Compared to an on-demand-only service, the assignment algorithm can be less selective in rejecting those requests that tend to be less efficient to serve. On the contrary, the saved distance increases for the Chicago case study. The main reason is a decrease in empty vehicle kilometers, as vehicle positioning is more accurate in the large operating area compared to repositioning in an on-demand-only service. Unsurprisingly, the highest values for saved distance can be observed for the high shareability distribution as efficient shared routes can be directly planned in the long-term optimization.

Figure 5.16 shows different KPIs for the Chicago case study with a homogeneous reservation



(a) Chicago: Served Requests.

(b) Munich: Served Requests.

(c) Manhattan: Served Requests.

(d) Chicago: Value of Reservation.

(e) Munich: Value of Reservation.

(f) Manhattan: Value of Reservation.

(g) Chicago: Saved Distance.

(h) Munich: Saved Distance.

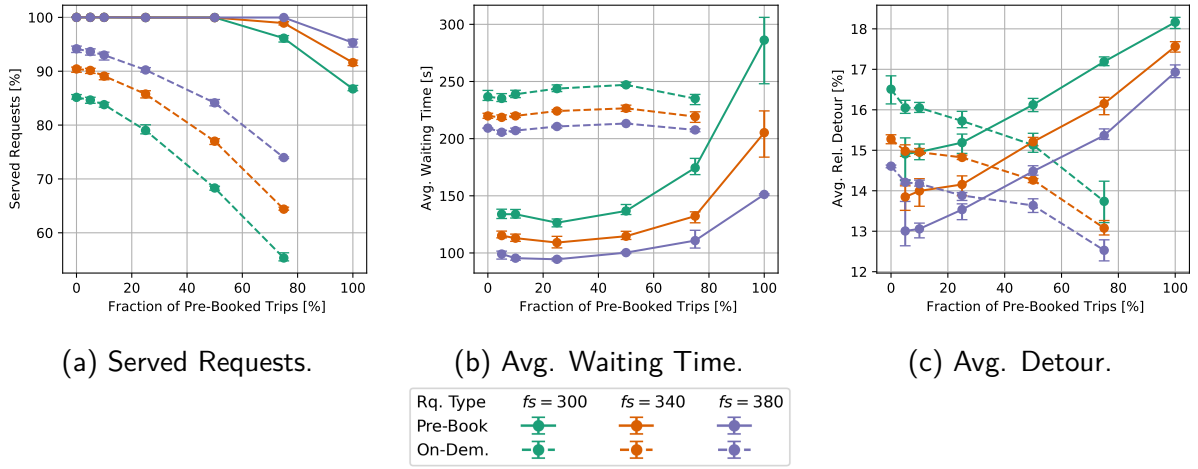
(i) Manhattan: Saved Distance.

(j) Chicago: Legend.

(k) Munich: Legend.

(l) Manhattan: Legend.

Figure 5.15: Effects of Long-Term Reservations and their Distribution. Columns show different Case Studies. Left Column: Chicago, Mid Column: Munich, Right Column: Manhattan.



(a) Served Requests. (b) Avg. Waiting Time. (c) Avg. Detour.

Figure 5.16: Different KPIs for the Chicago Case Study with Homogeneous Reservation Request Distribution.

request distribution for on-demand and pre-booking customers separately (the other case studies can be found in the Appendix IV). Figure 5.16a shows the fractions of served on-demand and pre-booking customers. As pre-booking customers are strongly prioritized by creating the long-term schedule independent of (expected) on-demand requests, all reservations are served for small fractions of pre-booking customers as available vehicle supply does not constrain the service. From fractions above 50%, pre-booking customers are rejected, as the long-term algorithm does not find feasible schedules for all requests. On the other hand, the fraction of served on-demand customers decreases steadily with the share of pre-booking customers, i.e., it becomes less likely for on-demand customers to access the service when many customers are pre-booking their rides (and the fleet size remains the same).

For the average waiting time (Figure 5.16b) and the average detour (Figure 5.16c), two opposite trends can be observed for on-demand and pre-booking customers. While both quantities remain relatively constant with the share of pre-booking customers, the average waiting time for pre-booking customers is considerably lower than for on-demand customers. For the average detour, the opposite trend can be observed. Lower waiting times (the time from the earliest pick-up time to the actual pick-up time) for pre-booking customers result from early planning of their pick-up, and vehicles can also arrive early. In contrast, a vehicle must be dispatched first in almost all cases for on-demand customers. On the contrary, pre-booking allows efficient planning and shared routes, which tends to increase the average detour.

5.4.3 Impact of Repositioning

In the previous sections, the base case to compare reservations with (a pure on-demand service) was rather favored because the repositioning algorithm with perfect forecast was applied, reducing stochasticity present in a real service.

Figure 5.17 therefore shows the impact when other, less accurate, repositioning algorithms are applied. Four different repositioning algorithms, which have been discussed in detail in the last chapter, are compared: The *Sampling*-algorithm with perfect (base case) and myopic

forecast, the *React*-algorithm, and no repositioning.

Similar to the previous section, the first row (Figure 5.17a, Figure 5.17b, and Figure 5.17c) shows the share of overall served requests with an increasing fraction of pre-booking customers for the different case studies. Most prominent is the scenario without repositioning. In this case, it is evident that the service performance strongly increases with the share of pre-booking customers as idle vehicles are actively repositioned to serve pre-booking customers, while in the on-demand-only service, they can only serve customers in their vicinity. Nevertheless, the system still benefits from repositioning until around 75% of customers pre-book their rides (if the pre-booking distribution is uniform).

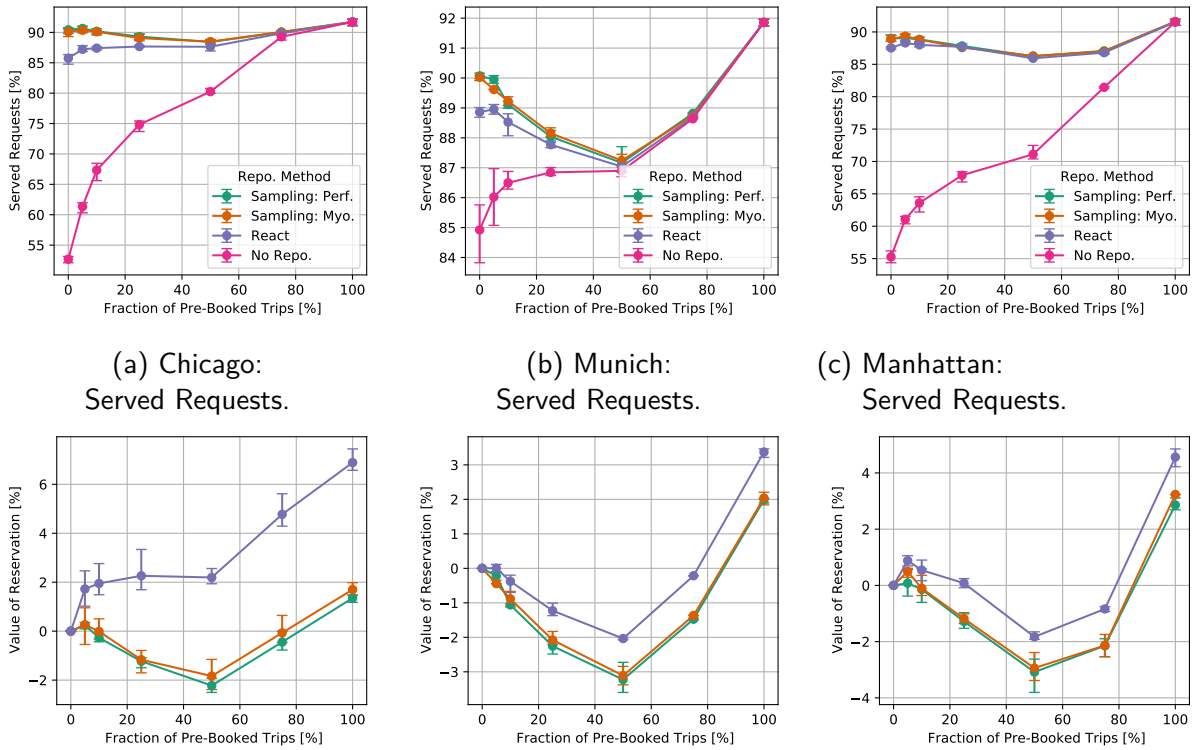
The difference between the repositioning algorithms can be best observed in the Figures of the second row (Figure 5.17d, Figure 5.17e, and Figure 5.17f) showing the value of reservation, omitting the scenarios with repositioning. As the *React*-algorithm serves significantly fewer customers than the *Sampling*-algorithm, in an on-demand-only service, the values of reservation change accordingly. For the Chicago case study, reservations even improve the service when customers pre-book their rides. Nevertheless, for the other case studies, the value of reservation remains negative when pre-booking and on-demand customers have a similar ratio. The same holds for the *Sampling*-algorithm with myopic forecast. While showing a slightly worse performance compared to a perfect forecast, the additional information about future demand from pre-booking does not compensate for the loss from incomplete knowledge in repositioning.

Overall, this evaluation shows the general importance of information about future demand for service performance. Regardless of whether information is available in aggregated form (as in the case of repositioning) or in exact form (as in the case of reservations), the system always benefits from additional information about future demand.

5.4.4 Impact of Re-Assignment

Next to the repositioning algorithm, the assignment algorithm can also significantly impact the service performance. The evaluation in chapter 5.2 showed the benefits of re-assignments in an on-demand-only service. Figure 5.18 shows the importance of re-assignments for the ARP service with reservations. It compares the *OPT*-algorithm that allows re-assignments and batch processing with the insertion heuristic *IH* that does not allow re-assignments for the different case studies and pre-booking penetrations. While the difference in served requests (Figure 5.18a) is in the order of 2% for all case studies for an on-demand-only service (as evaluated in previous sections), the difference increases drastically until a share of 50% pre-booking customers. In this case, a drop of around 7% is observed for the Munich case study, which accounts for unserved on-demand customers. This shows the importance of re-assignments in a service with reservations. The insertion heuristic mainly relies on the long-term schedules, where on-demand customers are inserted. Nevertheless, as long-term schedules are created independently of on-demand requests, no good solutions for serving additional on-demand customers can be expected. With the *OPT*-algorithm, on the other hand, the system can react to dynamically incoming demand and reconfigure short-term schedules to accommodate on-demand customers.

Figure 5.18b shows the value of reservation for the different assignment algorithms. Conse-



(a) Chicago: Served Requests.

(b) Munich: Served Requests.

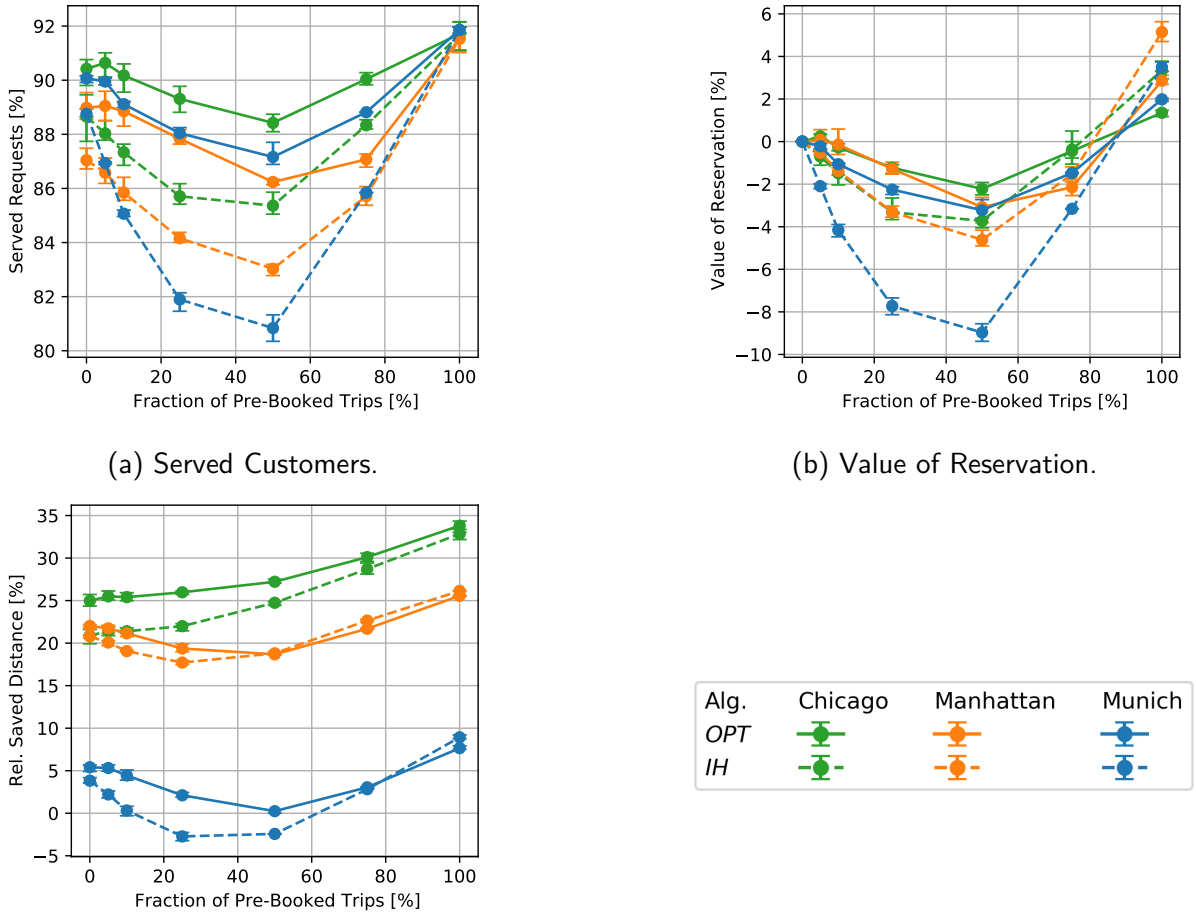
(c) Manhattan: Served Requests.

(d) Chicago: Value of Reservation.

(e) Munich: Value of Reservation.

(f) Manhattan: Value of Reservation.

Figure 5.17: Impact of Repositioning Algorithms with Reservation. Columns show different Case Studies. Left Column: Chicago, Mid Column: Munich, Right Column: Manhattan.



(a) Served Customers.

(b) Value of Reservation.

(c) Saved Distance.

Figure 5.18: Impact of Online Assignment Algorithm for Different Case Studies with Homogeneous Reservation Request Distribution. Fleet Sizes: Chicago: 340, Manhattan: 320, Munich: 240.

quently, the value of reservation drastically decreases with the share of pre-booking customers when the insertion heuristic is applied. For 50% pre-booking, an additional service degrading between 2% and 6% can be observed. For high values, on the other hand, the value of reservation is higher for the insertion heuristic, as the service becomes dominated by the long-term schedules applied in both algorithms. A lower reference value (i.e., the on-demand-only service) leads to a higher relative improvement in the overall objective value for the insertion heuristic.

Unsurprisingly, the saved distance (Figure 5.18c) is also higher for the *OPT*-algorithm. Especially for the Chicago and Munich case study, saved distance is decreased by up to 5% for the insertion heuristic resulting from less efficient vehicle schedules and additional empty vehicle kilometers.

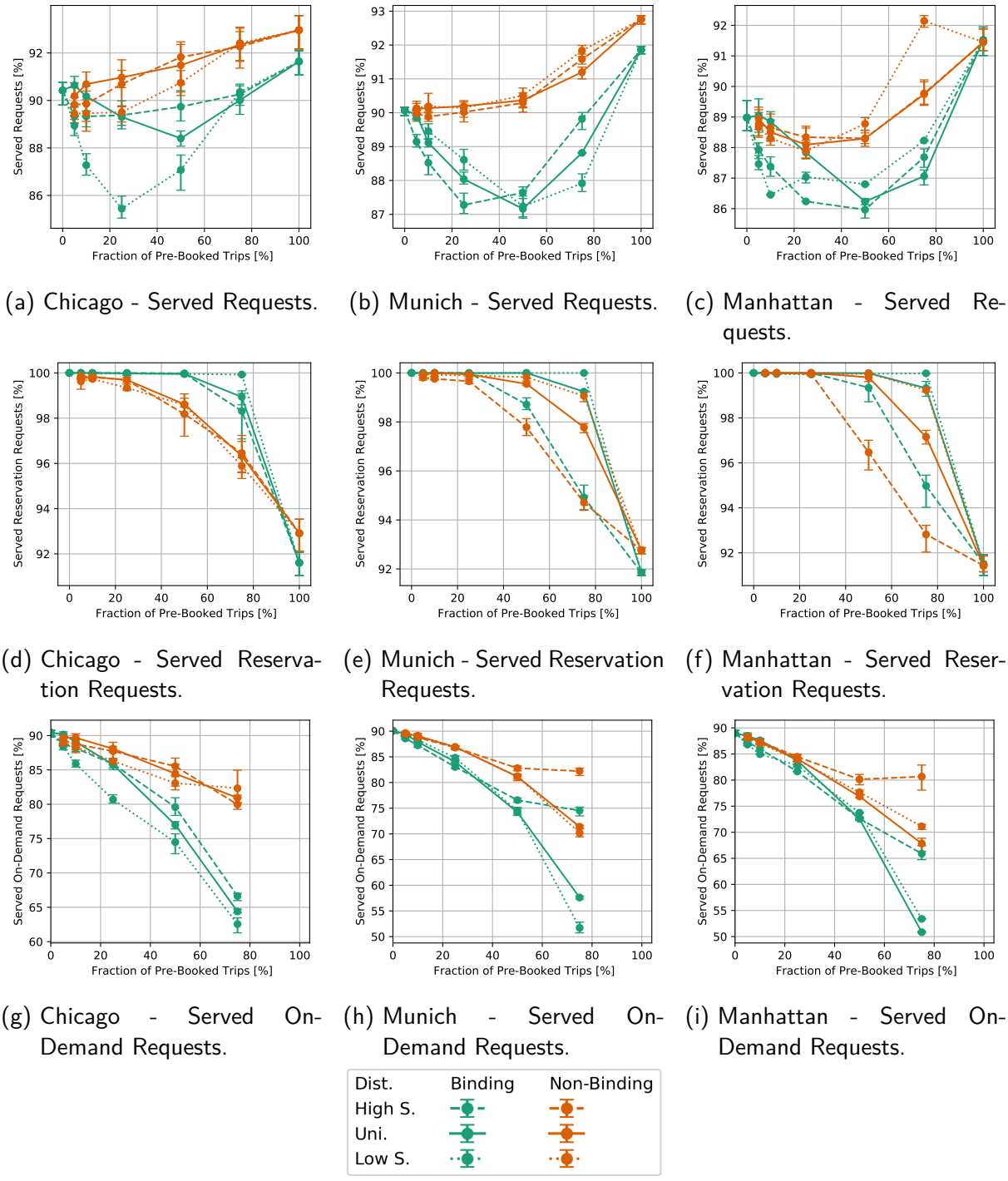
5.4.5 Cost of Service Guarantee

In this section, the alternative treatment of reservation requests is evaluated. Within this methodology, no long-term schedules for reservation requests are created. Instead, reservation requests are revealed to the assignment algorithm within a rolling horizon fashion defined by the short-term horizon T_{short}. In the online assignment algorithm, the assignment of reservation requests is prioritized over on-demand customers by increasing the assignment reward ten-fold compared to on-demand customers in the objective function. Therefore, the assignment algorithm can react more flexibly to incoming demand but cannot ensure that all reservation requests can be served. This can be interpreted either by a service that communicates late rejections to customers (which it tries to minimize) or confirms reservations only at the beginning of the short-term horizon.

Figure 5.19 shows the impact of binding and non-binding reservations on the different request types for the case studies. The first row (Figure 5.19a, Figure 5.19b, and Figure 5.19c) shows the overall fractions of served requests. Compared to binding long-term reservations, the overall number of served requests can generally be increased if reservation requests are treated as non-binding, especially if the share of pre-booking and on-demand customers is similar. In most cases, the number of served requests can even be increased compared to an on-demand-only service. This is not the case only for low fractions of pre-bookings in the Manhattan or Chicago case study with the low shareability distribution. When short-term rejections are possible, the system can react much more flexibly in the presence of incoming on-demand requests. If long-term reservations are binding, the decision to serve a reservation request is purely based on finding a feasible long-term schedule, merely considering any on-demand requests. On the contrary, if this decision is made in the short term, already revealed and assigned on-demand requests constrain the service's availability and allow identifying costly reservation requests that tend to be rejected if no vehicle is available within the pick-up time constraint. Nevertheless, as pre-booking customers are still prioritized, vehicles can still get pulled away from areas of high demand for on-demand trips, possibly reducing the overall number of served requests. This can be observed in the Chicago case study with low shareability distribution.

The second row (Figure 5.19d, Figure 5.19e, and Figure 5.19f) shows the service rate of reservation requests only. As expected, the service rate of reservation requests is higher for binding reservations, although the overall service rate tends to be lower. While creating long-term schedules allows a full-service rate for binding reservations in most scenarios, the service rate for non-binding reservations tends to decrease steadily, with the share of pre-booking customers starting notably at a share of 25%. Nevertheless, until a fraction of 25% pre-booking customers, late rejections of reservation requests happen in less than 0.5% of all cases. Considering the overall increased service rate, an operator might be willing to accept this trade-off. In practice, alternative trip offers (e.g., by offering a later pick-up) to further decrease unserved reservation requests.

For completeness, the third row (Figure 5.19g, Figure 5.19h, and Figure 5.19i) shows the fraction of served on-demand requests. Of course, based on the previous discussion, the fraction of served on-demand requests is higher for non-binding reservations, especially for high fractions of pre-booking customers, when most vehicles are constrained in the service by already assigned reservation requests.



(a) Chicago - Served Requests.

(b) Munich - Served Requests.

(c) Manhattan - Served Requests.

(d) Chicago - Served Reservation Requests.

(e) Munich - Served Reservation Requests.

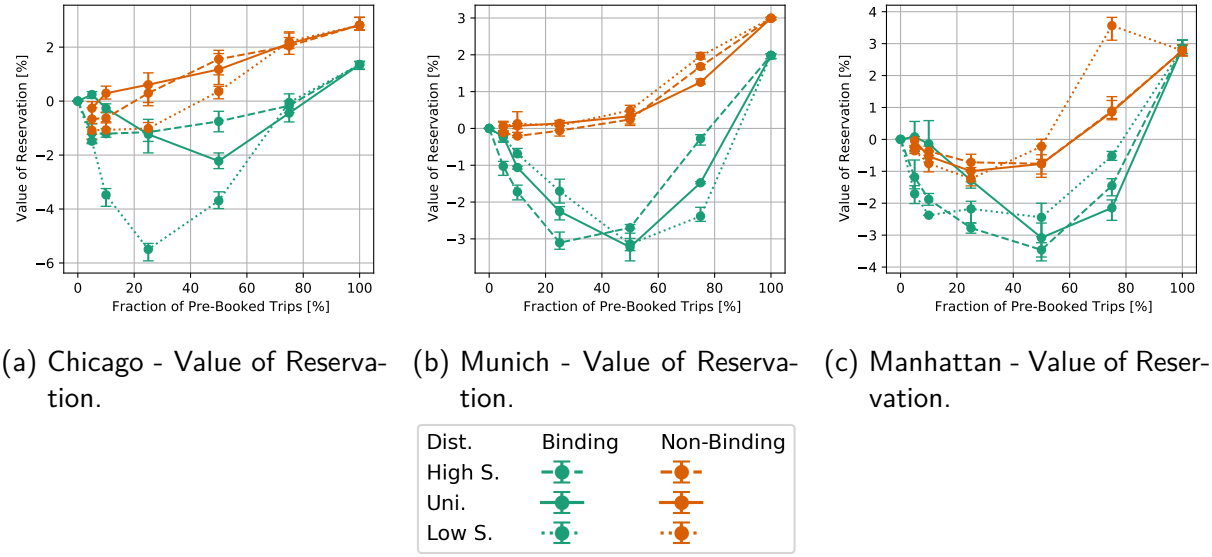
(f) Manhattan - Served Reservation Requests.

(g) Chicago - Served On-Demand Requests.

(h) Munich - Served On-Demand Requests.

(i) Manhattan - Served On-Demand Requests.

Figure 5.19: Impact of Binding and Non-Binding Reservations on Served Requests. Columns show different Case Studies. Left Column: Chicago (340 vehicles), Mid Column: Munich (240 vehicles), Right Column: Manhattan (320 vehicles).



(a) Chicago - Value of Reservation.

(b) Munich - Value of Reservation.

(c) Manhattan - Value of Reservation.

Figure 5.20: Impact of Binding and Non-Binding Reservations on Service Level. Columns show different Case Studies. Left Column: Chicago (340 vehicles), Mid Column: Munich (240 vehicles), Right Column: Manhattan (320 vehicles).

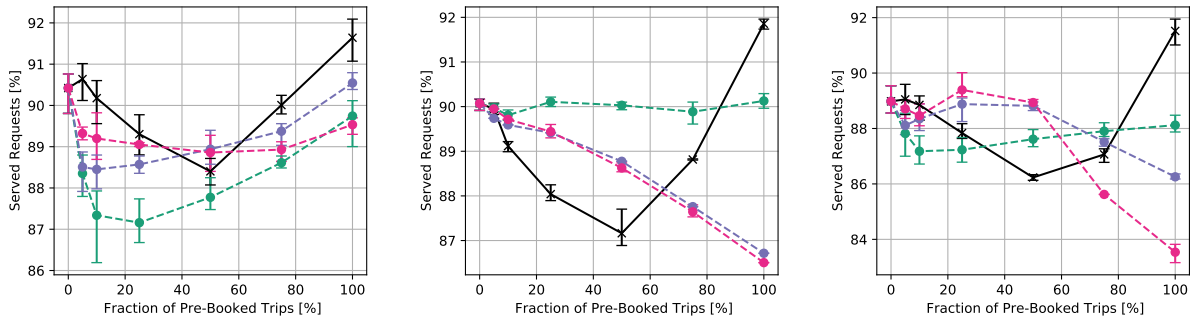
Similarly, Figure 5.20 evaluates the Value of Reservation (first row) for the different case studies. In contrast to binding reservations, the value of reservation is generally higher for non-binding reservations and even positive in most cases, as information on future demand can be used more efficiently when constraints are less binding. The difference in VoR between binding and non-binding reservations can be interpreted as a "cost of service guarantee" of up to about 4%. An operator might, for example, increase fares by this factor to compensate for the additional service quality.

5.4.6 Impact of Reservation Horizon

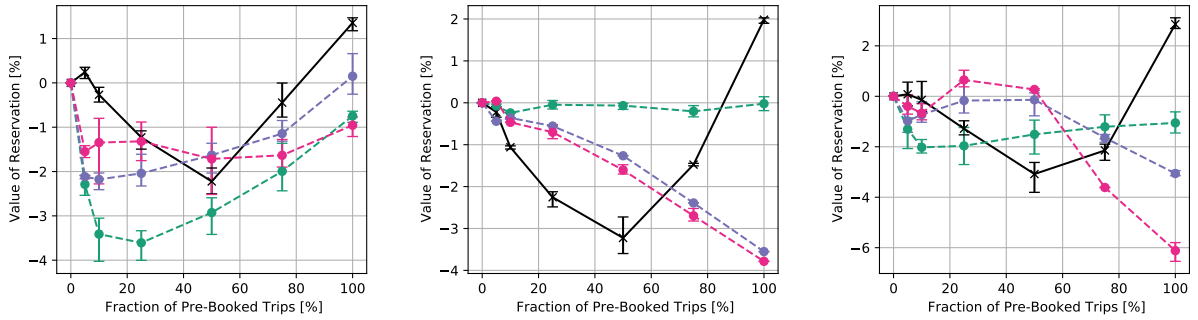
This section evaluates the reservation horizon, i.e., the time between the reservation and the requested pick-up time. Customers continuously make reservations for the service, and long-term schedules must be updated online to assign new reservations. Therefore, the *CBO*-method is used in this section to treat reservations.

Figure 5.21 shows the impact of the reservation horizon on the service level for the different case studies and compares it with the case of long-term-only reservations. The first row (Figure 5.21a, Figure 5.21b, and Figure 5.21c) shows the fractions of served requests for the different case studies. As a general trend, the minimum at 50% pre-booking customers for long-term-only reservations can not be observed when reservations are continuously made. This is likely because reservation requests start to get rejected at a lower penetration of pre-booking customers. (see Figures 5.21g, 5.21h, and 5.21i). In favor of serving on-demand customers, more information about the current system state can be incorporated into the decision to assign reservation requests when the reservation time decreases on average. Nevertheless, the impact of the reservation horizon on the service level differs between the case studies. For Chicago, especially short reservation horizons tend to decrease the service level. In this case, vehicles get pulled away from high-demand areas for on-demand trips to serve reservations. A short planning horizon for short-term reservations can increase this effect. On the contrary, for Munich, the service level is highest for a reservation of 15 minutes on average, keeping the service level relatively constant. For longer reservation horizons, the service level decreases, indicating reduced availability for on-demand customers when long-term commitments are made.

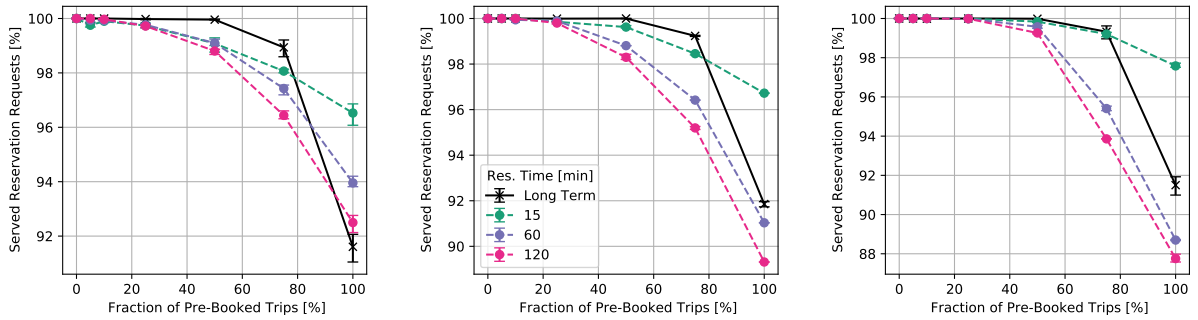
Generally it can be observed (e.g., in Figure 5.21d, Figure 5.21e, and Figure 5.21f) that for very high fractions of pre-booking customers, the value of reservation is lower if no long-term-only pre-bookings are made. In this case, the *CBO*-method provides worse solutions for the long-term schedules than the *NCRH*-method used for long-term-only reservations. In contrast to the *NCRH*-method, the *CBO*-method has to continuously update its long-term schedules while making commitments to incoming reservations, while for long-term-only reservations, all pre-booking customers are known from the beginning.



(a) Chicago - Served Requests. (b) Munich - Served Requests. (c) Manhattan - Served Requests.

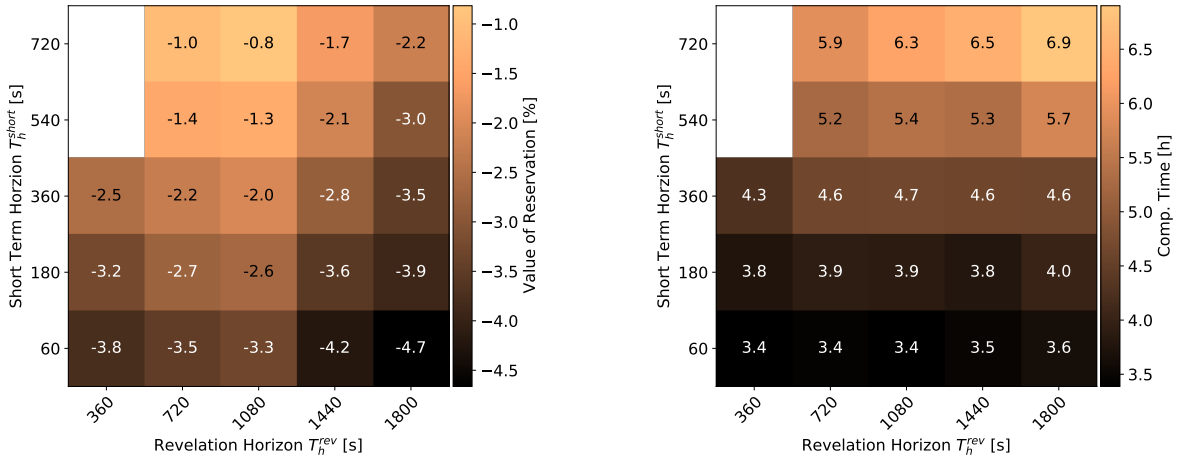


(d) Chicago - Value of Reservation. (e) Munich - Value of Reservation. (f) Manhattan - Value of Reservation.



(g) Chicago - Served Res. Requests. (h) Munich - Served Res. Requests. (i) Manhattan - Served Res. Requests.

Figure 5.21: Impact of Reservation Horizon on Service Level. Columns show different Case Studies. Left Column: Chicago (340 vehicles), Mid Column: Munich (240 vehicles), Right Column: Manhattan (320 vehicles). Legend in Figure 5.21h valid for all subfigures.



(a) Value of Reservation.

(b) Overall Computational Time.

Figure 5.22: Impact of Horizons for Chicago Case Study with 25% Reservation Requests from the Homogeneous Distribution and 340 vehicles.

5.4.7 Evaluation of Rolling Horizons

This section evaluates the impact of the applied rolling horizons.

Therefore, the sensitivity of the horizon parameters T_h^{rev} and T_h^{short} for inserting the long-term solution for pre-booked trips into the online algorithm are assessed. Figure 5.22 compares the impact of various combinations for T_h^{rev} and T_h^{short} on the overall computational time and the improvement in the Value Of Information VoR for the Chicago case study.

Generally, it can be observed that with higher T_h^{short}, the VoR improves while computational time increases. With longer T_h^{short}, pre-booked requests can be reassigned by the online optimization earlier. Therefore, the online optimization can adapt the vehicle routes of pre-booked customers earlier for incoming on-demand requests. As a trade-off, computational time increases due to a growing solution space.

Similarly to T_h^{short}, with higher T_h^{rel} the computational time increases, too. Nevertheless, concerning the VoR, an improvement is observable until $T_h^{rel} = 1080$s, followed by a degrading in solution quality. With longer T_h^{rel}, additional pre-booked stops are revealed to the online optimization, which cannot be assigned to other vehicles, but new on-demand requests can be inserted into less constrained schedules. Therefore, the VoR increases with higher T_h^{rel}, but, again, computational time increases because longer schedules have to be checked for feasibility. While this seems beneficial for the system as long as $T_h^{rel} \leq 1080$s, early revelations also have disadvantages: First, revealed requests that are not yet within T_h^{short} cannot be reassigned, but waypoints after T_h^{rel} can. At $T_h^{rel} = 1080$s, the advantage of re-assigning upcoming waypoints seems to outweigh the advantage of more information for inserting on-demand requests into currently assigned schedules. Second, network travel times are varied every hour. If T_h^{rel} is high, a larger part of the long-term schedule is revealed that has been planned based on different travel times, leading to potentially infeasible schedules forbidding the insertion of on-demand requests.

Chapter 6

Conclusion

The goal of this thesis was to investigate the operational challenges of Autonomous Ride-Pooling (ARP) services in urban environments and to develop methods to address these challenges. Three main aspects of ARP services have been assessed: assignment, repositioning, and reservation. The algorithms to address these aspects of operation in an ARP service were evaluated in agent-based simulations for three distinct urban case studies: Chicago, Munich, and Manhattan.

This chapter concludes the thesis by summarizing the main contributions and answering the research questions. The chapter also discusses the limitations of the work and provides suggestions for future research.

6.1 Answer to Research Questions and Limitations

Based on the results provided in the previous chapter, the research questions of this thesis defined in section 1.3 are answered in the following.

6.1.1 Impacts of Ride-Pooling

Simulation-based evaluation of autonomous ride-pooling services was one of the main contributions of this thesis. Therefore, a detailed agent-based simulation framework was developed with the ability to simulate various aspects of ARP services. Three distinct urban case studies — Chicago, Munich, and Manhattan — were conducted to explore various general questions about ARP services.

The simulation results provided in Section 5.1 allow to answer the research question

RQ I: What are the benefits of pooling rides?

By varying the fleet size in the case studies, it was found that approximately 1,250 of passenger capacity four or higher are enough to serve 90% of the approx. 130,000 TNC trips in Chicago. Comparing a pooling service to a hailing service, that does not allow for shared rides, this translates to approx. 1,000 fewer vehicles that are required to serve the demand. Additionally, the fleet of pooling vehicles produces only half of the VKT compared to the hailing service, proving the high societal, operational, and environmental benefits of pooling rides. Due to the high demand density and rather short taxi trips in Manhattan, a fleet of just 1,500 vehicles with a capacity of four or more is sufficient to serve nearly all of the 210,000

trips. This fleet size corresponds to only around 11% of the current taxi fleet in New York. The case study for Munich involved the replacement of private vehicle trips with the ARP service. Substituting 10% (approx. 102,000) of Munich's inner-city private vehicle trips with an ARP service by 1,250 vehicles could result in an estimated vehicle replacement rate of 97% if those private vehicles would not be in use anymore.

With respect to required vehicle capacity, the simulations of all case studies showed similar results: A huge operational gain is achieved when increasing vehicle capacity from one to two seats, therefore allowing for shared rides between two passengers. This gain can be further increased up to a capacity of four seats. A larger capacity, however, does not provide many additional benefits.

A variation of different demand levels showed that ride-pooling demand is one of the key factors for a successful ARP service. The case studies for Chicago, Munich and Manhattan all showed the same trend: With higher demand, a single vehicle can serve more customers, the pooling rate increases, therefore reducing the VKT per customer and also empty VKT. The reason is clear: With more demand, the probability of finding a matching customer for a vehicle increases, allowing for more efficient vehicle routes. Customers profit from shorter waiting times as also the vehicle density in the city increases, when more customers are to be served. Only the in-vehicle travel time of customers increases with higher demand, as high sharing also leads to higher detour to pick other customer up.

Overall, the results in this thesis agree with general findings in the literature (Table 2.2). Huge benefits in terms of vehicle replacement and VKT reduction can be achieved by pooling rides if the demand is high enough and low occupancy modes, like private cars, taxis, or TNC services, can be substituted. Nevertheless, a maximum saved distance of 42% found in the case studies, indicates that cities should not rely solely on ARP services to solve all traffic problems. If too many travelers switch from public transportation or active modes to ARP services, the benefits of pooling rides can be quickly offset by induced demand. By closing gaps in availability and convenience of the public transport network, ARP services should therefore be seen as a complement to public transportation and active modes.

Limitations

A major limitation of the simulation results is the rather simplistic demand model used in the case studies. By merely replacing trips from private cars, taxis, or TNC services, the simulation allows for a structured evaluation of a multitude of scenarios, but does not capture the full complexity of urban mobility. To evaluate the global impacts of ARP services, a more sophisticated demand model is needed, for giving travelers the possibility to choose between different modes of transportation. Additionally, the simulation framework should be extended to include also other modes of transportation to evaluate trip alternatives. If mode choice is included, the simulation framework also needs to be extended to include a pricing model for the ARP service, which has not been considered in this thesis. Lastly, also the traffic impacts of ARP services are not fully evaluated in this thesis. While the saved distance is a good indicator for the efficiency of the service, the actual traffic impacts can only be evaluated by a more detailed traffic simulation.

6.1.2 Assignment

One key contribution of this thesis is the development of an assignment algorithm that allows for dynamic assignment of customers to vehicles in an ARP service, and therefore answers research question

> **RQ II:** How can the operator of an ARP service assign customers and schedules to fleet vehicles efficiently?

The algorithm developed in this thesis elaborates on a sophisticated search strategy developed by Alonso-Mora et al. [2017a]. By exploiting tight time constraints on customer pick-up and delivery times, that ensure an attractive service to customers, a set of necessary requirements can be formulated to efficiently find a large set of feasible schedules. The algorithm then assigns these schedules to vehicles solving an ILP.

A special contribution of this thesis, is keeping already computed schedules in a database to reuse them in later assignment steps. As creating feasible schedules is the most computationally challenging part of the algorithm, reducing the number of schedules that have to be computed can significantly reduce the computational time of the assignment algorithm, ensuring short response times and real time applicability. The case study confirmed, for example, 76% reduction in computational time for a simulation of the Chicago case study.

Several heuristics and benchmark algorithms were compared to the developed assignment algorithm. While the developed assignment algorithm was able to solve the assignment problem for most instances close to optimality (i.e., by finding all feasible schedules), the heuristics and benchmark algorithms could be sorted by their constraint in explorable solution space to evaluate the trade-off between computational time and solution quality. Based on the order of the algorithms, compared to the optimal solution of the assignment problem, two steps stand out: First, prohibiting re-assignments, and second, treating customer requests sequentially instead of in batch. In the presented Chicago case study, allowing re-assignments resulted in 1.4% more customers being served, a 2.56% increase in saved distance, and a reduction in average customer delay by 35 seconds, compared to a batch assignment that restricts re-assignment. Additionally, when customer requests were treated sequentially, the number of served customers further decreased by 1.5%, the saved distance decreased by 3.1%, and the average customer delay increased by 38s. Therefore, allowing re-assignments tends to have a similar benefit for the service as treating customer requests in batch, compared to treating them sequentially (at least for the parameters chosen in this case study). On the contrary, the computational effort to solve the assignment problem increases with available solution space. In the example mentioned above, an average computational time of 45s computational time was consumed for the optimal solution, while 15s were needed when no re-assignments were allowed, and below 2s when customer requests were treated sequentially. With a batch epoch of 30s, an optimal solution cannot be provided in real-time within this implementation. However, the developed vehicle selection heuristics and time outs allow for a real-time application of the algorithm, while still providing close to optimal solution quality.

Next to the operational benefits of re-assignment, also the impact of re-assignments on customers was investigated. The evaluation of the case studies showed, that 78 to 86% of customers did not experience any re-assignment at all, indicating a rather stable system, where the gain in operational benefits come from the re-assignment of a minority of the requests.

Nevertheless, approximately half of the customer re-assignments occur on a rather short notice, defined as less than 3 minutes before scheduled pick-up. Especially for these customers, re-assignment can be particularly inconvenient, as they may already be waiting for the vehicle when they receive a notification of a delay.

Therefore, different strategies have been proposed to reduce the drawbacks of re-assignments and therefore increase service reliability for customers. These strategies include a guaranteed time-window for pick-up, locking of assigned vehicles shortly before pick-up, or penalizing re-assignments in the objective function of the assignment algorithm. Especially guaranteeing a fixed time-window of one minute around the initial pick-up time, showed to be a promising strategy to increase service reliability for customers, while still maintaining the operational benefits of re-assignments.

Limitations

While the proposed assignment algorithm is able to solve the formulated static DARP even to optimality in many cases, an optimal assignment for the general (long-term) ride-pooling can likely not be found by this approach. The main reason is that this algorithm is myopic, i.e., it does not consider future demand when creating the current assignment. Even though repositioning covers the imbalance of demand and supply in the long-term, better assignments can likely be found if forecasts of future system states are considered. In this case, more informed decisions can be made, e.g., by assigning schedules to guide vehicles towards future demand, or make better choices in which customers should be rejected. Next to schedules, i.e., orders of customers to be served, also the routes that vehicles should take can be optimized. While always taking the fastest path in this study, coordinated routing of vehicles might become necessary especially when large fleet sizes are applied.

The evaluation of re-assignment showed the general benefits of re-assignment for the operator of an ARP service. However, allowing re-assignments might provide technical difficulties not considered in the simulation. In real-world applications, the computational time, even though only in the range of seconds, can result in time lags, possibly leading to conflicts in the assignment if the system state changes unexpectedly. This could be especially problematic with re-assignments, as changes in assigned vehicle routes happens more often than without re-assignments. From a customer perspective, the acceptance of re-assignments is an open research question. If perceived too inconvenient, customers might not use the service anymore, leading to a loss in revenue for the operator, although the service is more efficient.

6.1.3 Repositioning

The second aspect of operation of an ARP service that was investigated in this thesis is repositioning with the goal to answer the research question

> **RQ III:** How does an ARP service benefit from repositioning?

A central contribution of this thesis was the development of a repositioning algorithm that directly incorporates the possibility of sharing rides into its formulation. The algorithm samples trip from a predicted demand distribution. By creating actual vehicle routes from the sampled

trips, an estimation of required vehicles can be made, thereby providing a measure of the imbalance of demand and supply.

An optimization problem was formulated to assign idle vehicles to zones, where they can serve the schedules created from future sampled requests. The objective value of the schedules is used to determine the value of the repositioning to the corresponding zone. The formulation also incorporates a multi-horizon approach, which considers potential future repositioning steps to accurately estimate the imbalance between demand and supply for future time periods. To reduce the impact of stochastic variation within the sampling process, the optimization problem can incorporate multiple samples to provide a more robust solution. Thereby, repositioning trips are assigned that provide good solutions across all samples.

A comparison of the repositioning algorithm to a no-repositioning scenario showed the huge benefits of repositioning for an ARP service. In the Chicago and Manhattan case studies, the fraction of served requests could be increased by up to 40%, while time that vehicles generated revenue for the service for up to 6 additional hours per day. The spatial evaluation showed the repositioning algorithm successfully repositions idle vehicles from zones with low demand to zones with high demand. If no repositioning is conducted, the spatial evaluation reveals that vehicles tend to accumulate in areas of low demand. With not customer requests in the vicinity, these vehicles remain idle even during peak demand times. Even though also the simulated ARP service in the Munich case study benefits from repositioning, the benefits are less pronounced compared to the other case studies, likely due to a more balanced demand distribution.

In the case studies for Chicago, Munich, and Manhattan, the repositioning algorithm was compared to other state-of-the-art repositioning algorithms, that, however, do not consider the possibility of shared rides explicitly. The simulation results showed that the repositioning algorithm developed in this thesis outperforms the other repositioning algorithms. Especially in the case of Chicago, the repositioning algorithm was able to serve by far the most customers, while achieving nearly the highest savings in distance and the lowest average customer waiting times. The same effect is observed when different forecasting methods (perfect and myopic) are used to predict future demand, showing a generally stable performance of the repositioning algorithm. Only for the Manhattan case study, an aggressive hailing-based approach was able to serve more customers, though at the cost of higher empty vehicle kilometers. A spatial evaluation showed, that tends to balance vehicles across the city, while the comparison algorithm tends to concentrate vehicles in the city center, a strategy that seems to be beneficial in the Manhattan case study.

Limitations

While the repositioning algorithm developed in this thesis is able to provide a good solution to the repositioning problem, the applied demand model of serving a fixed set of customer requests is rather simplistic. On the one hand, the simulation model does not take long-term effects of repositioning into account. Depending on service availability, largely influenced by repositioning decisions, customers might change their travel behavior. This might, for example, lead to a lower demand in areas with low service availability, because fewer vehicles are repositioned there. Additionally, full knowledge of future traffic states is assumed in the repositioning algorithm, when evaluating future fleet states.

Also in the formulation of the repositioning algorithm, improvements can be made. If a general undersupply of vehicles is detected, i.e., the number of possible repositioning trips does not suffice to serve sampled requests, the algorithm tends to greedily assign repositioning trips to minimize cost. With respect to stochastic variation, an approach to distribute vehicles proportional to supply shortages might be more beneficial in this case to enable a more balanced system.

6.1.4 Reservation

The last aspect of operation of an ARP service that was investigated in this thesis is reservation with the goal to answer the research question

RQ IV: Does an operator of an ARP service benefit from offering pre-bookings?

A multi rolling horizon approach was developed to incorporate reservation requests into the assignment algorithm. In the multi rolling horizon approach, incoming customer requests are divided into short-term and long-term requests. Because of their upcoming pick-up time, short-term requests are directly assigned to vehicles, by treating them as on-demand requests. For long-term requests, however, the concept of a long-term schedule is introduced. The long-term schedule covers the entire planning horizon of requested long-term pre-bookings. It is used to estimate available capacity for accommodating long-term requests, far in advance of the current planning horizon of the assignment algorithm. The long-term schedule is assigned to a specific fleet vehicle and is used to feed waypoints into the assignment algorithm. The waypoints are used to ensure that the vehicle is available at the right location at the right time to serve the long-term request. The assignment algorithm then assigns the vehicle to on-demand requests, while still ensuring that the long-term requests are served.

As long-term schedules and repositioning act on similar horizon time scales, the repositioning algorithm was extended to also consider long-term schedules and not yet assigned long-term requests. Long-term schedules without immediately upcoming pick-up are detached from the vehicle and used as seed when creating schedules in the sampling process of the repositioning algorithm. Enforcing the assignment of repositioning trips to the respective zones, the repositioning algorithm can assign repositioning trips and re-assign waypoints in the same formulation.

Extensive simulations for the three case studies revealed, that the operator of an ARP service only benefits from pre-bookings under specific conditions. One option is, that the fraction of pre-bookings is high. In this case the operator can benefit from additional knowledge and can assign efficient vehicle schedules. Another option, at least in the case of the Chicago and Manhattan case study, is that a rather small fraction (up to 10%) of pre-bookings is present, and their spatial distribution is highly correlated with the spatial distribution of on-demand customers. In this case, only few long-term schedules of reservation requests constrain the assignment of on-demand requests and the waypoints of long-term schedules guide vehicles to areas of on-demand requests, too. The last option is to provide a non-binding (prioritized) reservation system, that allows cancelling the reservation of a customer a few minutes before a pick-up. Non-binding reservations tend to be beneficial for the operator in most cases, while the number of (late) cancellations of reservations is rather low until a fraction of 50%

pre-bookings. The highest deterioration of service quality due to reservation was observed in the Chicago case study, when 10-50% of the requests were pre-bookings and those customers tend to reserve a trip that are less shareable. As these trips are usually in low demand areas, vehicles are positioned to these areas to commit to the reservation, while profitable on-demand requests in high demand areas remain unserved.

The results for the impact of reservations on the service quality are mixed, especially when service guarantees are assumed. However, this observation is in line with previous findings in the literature (Table 2.5). From an operator perspective, it is therefore essential to evaluate if reservations can attract more customers to the service, and if those additional revenues can compensate for the potential loss in service quality. If these revenues are not sufficient, the operator could consider a pricing strategy for reservations, to ensure that the service quality is not deteriorated by reservations. As the simulation showed, that non-binding reservations can be beneficial for the operator in most cases, another strategy could be to offer non-binding reservations to customers, and compensate unserved reservation by offering discounts. Nevertheless, besides potential negative impacts on the service, reservations can also provide a chance for fair access to the service, especially in low demand areas, where the service might not be available otherwise.

Limitations

Also for reservations, long-term effects of reservations on the usage of the service were not modeled in the simulation. Behavioral data on how customers react to the possibility to reserve a trip is needed for evaluation, either by evaluating data of a real service offering reservations, or surveying potential customers. From an operational perspective, simplified assumptions were made concerning knowledge about traffic states. Perfect knowledge about future traffic states were assumed to plan long-term schedules and therefore ensure the fulfillment of reservations. In reality, traffic always remains a source of uncertainty. Therefore, it can occur that a vehicle is not able to serve a reservation, even though the operator planned for it. Methods have to be developed to handle such cases.

6.2 Future Research Directions

Starting from this thesis, several future research directions can be identified. In general, extended behavioral models are required to evaluate the impact of ARP services on the overall transportation system. But also for the key aspects of re-assignment, repositioning, and reservation, behavioral models need to be developed to estimate their long-term effects on the demand of an ARP services. Additionally, the coupling of the proposed simulation framework with traffic simulation models would allow for a more detailed evaluation of the traffic impacts of ARP services.

To fully exploit the potential of autonomous vehicles, different service designs should be evaluated. This includes the integration into public transportation, e.g., by providing feeder services or provisions for undersupplied origin-destination relations in the public transportation network. Additionally, mixed fleets can be studied to evaluate services with different service levels, e.g., high-capacity and premium services. As the utilization of vehicles is dependent on

the demand, the integration of delivery services can be beneficial to increase the utilization of vehicles in off-peak times. Finally, it should be evaluated if the findings of this thesis can be transferred to rural areas, where the demand is less dense, but traffic efficiency might be less crucial.

For the assignment problem, the developed algorithm can be extended to consider future demand. Identified supply shortages by the proposed repositioning algorithm could be used in the assignment objective function to guide vehicles towards future demand. The impact of re-assignment should be evaluated in a more dynamic and stochastic setting, e.g., by considering stochastic traffic states or customer behavior like late cancellations. It can be assumed that the system can adapt better to unexpected changes in the system state, if re-assignments are allowed. These changes could also include demand surges by, for example, public transport disruptions or special events. A possible solution could be to dynamically relax time constraints for the service to increase the effective capacity of the fleet, and thereby contribute to the general resilience of the transportation system.

With respect to repositioning, forecasting demand for the use-case of an ride-pooling service remains an open questions. To evaluate the shareability of future demand and, therefore, to estimate required vehicle supply forecasts have to be on high quality on spatial as well as on temporal level. Repositioning decisions could also be improved by incorporating long-term effects on demand and fairness considerations. This adaption would implicitly require additional research on customer behavior in case of a regularly unavailable service. Finally, the developed repositioning method requires a forecast of traffic state. In this thesis, a perfect knowledge of this state is assumed leaving further room for evaluating the impact of incomplete knowledge.

Some improvements are also possible for the applied methodology for reservations. Similar to repositioning, incomplete knowledge of future traffic states is a pressing question that needs to be tackled as unexpected traffic delay will result in infeasibilities with respect to commitments for reservations. Methods must be developed to deal with these conflicts. One option could be a multi-step approach for the communication with the customer: Commit to a general service first, but communicate exact pick-up times only on a shorter notice, when more knowledge about the system state is available. This way, more flexibility would remain for the online optimization. As the results showed, that reservation generally do not improve the service, methods could be developed to detect unpleasant reservations that might be harmful for the service. By either rejecting those requests for reservation early, or adjust fares accordingly, could improve the overall service quality. Finally, a general pricing strategy for reservations should be developed with respect to the results of this thesis.

In general, artificial intelligence is a promising candidate for solving operational problems for ARP services. The highly dynamic and stochastic setting makes especially supervised and unsupervised learning a suitable method. This includes the assignment process by reducing the number of required solutions to the vehicle routing problem. Either the vehicle routing problem could be solved directly by a trained model, or a model could pre-select promising candidate vehicles for incoming requests before a vehicle routing problem is solved. The same applies for providing feedback to customer requests, when vehicle availability has to be estimated. Also repositioning is a suitable candidate for the application of artificial intelligence. Instead of providing a demand forecast, a model could be trained to directly predict required supply, rendering the computational expensive methodology developed of progressing fleet

states into the future unnecessary. Unsupervised learning could even be used to directly learn to reposition decisions. For reservations, machine learning could be used to evaluate if a pre-booking should be accepted or not. Overall, machine learning provides a lot of applicability by being able to learn complex environments. Suitable models can especially reduce required computational time significantly, because fewer solutions to a vehicle routing problem are required. Nevertheless, future research must provide solutions on how to train these models, as usually a lot of training data is needed, which can only be required by extensive and computationally costly simulations.

List of Figures

List of Tables

List of Terms and Abbreviations

AMoD Autonomous Mobility-on-Demand 20, 21, 23, 25–27, 29–31, 38, 44, 47, 173

ARP Autonomous Ride-Pooling 3–11, 21–24, 26, 28, 31–35, 43, 49–52, 62, 65, 72, 82, 89, 95, 98, 100–102, 105, 109, 110, 113–115, 118, 119, 122, 124, 125, 129, 136, 137, 140, 142, 144, 146, 147, 150, 159–166, 169, 170

AV Autonomous Vehicle 3, 4, 19, 21, 23

CSV Comma-Separated Values 97, 98

DARP Dial-a-Ride Problem 31–33, 40, 43, 50, 57, 60, 62, 65–67, 70, 88, 124, 125, 162

DRT Demand Responsive Transit 17

EU European Union 1

EV Electric Vehicle 29

FCFS First-Come-First-Served 33

ILP Integer Linear Programming 61, 77, 90, 161

KPI Key Performance Indicator 5, 91, 93, 100, 109, 111–113, 118, 124, 129, 131, 132, 136, 137, 140, 146, 147

LSTM Long Short-Term Memory 39

MATSim Multi-Agent Transport Simulation 39

MFD Macroscopic Fundamental Diagram 27

MILP Mixed-Integer Linear Programming 59

MIP Mixed-Integer Programming 58, 59

MoD Mobility-on-Demand 17, 38, 93, 94, 101

MPC Model Predictive Control 40–42

NP Non-deterministic Polynomial-Time 31, 59, 65

OD Origin-Destination 102, 118, 119

OEM Original Equipment Manufacturer 18

P2P Peer-to-Peer 15
PDPTW Pickup and Delivery Problem with Time Windows 31

SAEV Shared Autonomous Electric Vehicle 2

TNC Transportation Network Company 2, 16, 38, 159, 160

US United States 1, 5, 13, 16, 17, 21, 27, 33

V2RB Vehicle-to-Request-Bundle 66–70, 86, 123–125, 127, 128
VKT Vehicle Kilometers Traveled 16, 21, 22, 29, 37–39, 87, 111, 115, 117, 118, 120, 140–143, 159, 160, 213
VRH Vehicle Revenue Hours 111
VRP Vehicle Routing Problem 6, 31, 43

Own Publications

Directly Related Publications

The following publications directly relate to the context in this thesis.

Assignment (Chapter 3.2)

Roman Engelhardt, Florian Dandl, Aledia Bilali, and Klaus Bogenberger [2019]. "Quantifying the Benefits of Autonomous On-Demand Ride-Pooling: A Simulation Study for Munich, Germany". In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, pp. 2992–2997. ISBN: 978-1-5386-7024-8. DOI: 10.1109/ITSC.2019.8916955

Roman Engelhardt, Florian Dandl, and Klaus Bogenberger [2020]. *Speed-up Heuristic for an On-Demand Ride-Pooling Algorithm*. URL: https://arxiv.org/pdf/2007.14877, Presented at Informs Annual Meeting 2019

Repositioning (Chapter 3.3)

Roman Engelhardt, Hani S. Mahmassani, and Klaus Bogenberger [2023]. *Predictive Vehicle Repositioning for On-Demand Ride-Pooling Services*. URL: http://arxiv.org/pdf/2308.05507v1, Presented at TRB Annual Meeting 2024

Roman Engelhardt, Hani S. Mahmassani, and Klaus Bogenberger [2024]. "Vorhersagebasierte Fahrzeugrepositionierung für On-Demand-Ride-Pooling-Dienste". In: *Straßenverkehrstechnik* 06, pp. 467–474. ISSN: 0039-2219. DOI: 10.53184/SVT6-2024-3

Reservation (Chapter 3.4)

Roman Engelhardt, Florian Dandl, and Klaus Bogenberger [2022a]. *Simulating Ride-Pooling Services with Pre-Booking and On-Demand Customers*. URL: http://arxiv.org/pdf/2210.06972v1, Presented at TRB Annual Meeting 2023

Simulation Framework (Chapter 4)

Roman Engelhardt, Florian Dandl, Arslan-Ali Syed, Yunfei Zhang, Fabian Fehn, Fynn Wolf, and Klaus Bogenberger [2022b]. *FleetPy: A Modular Open-*

Source Simulation Tool for Mobility On-Demand Services. URL: https://arxiv.org/pdf/2207.14246

Other Publications In the Context of the Dissertation

The following publications do not directly relate to the context of this thesis, but as they all relate to the context of autonomous mobility on-demand services they shaped my understanding of the topic presented in this thesis.

Journal Articles

Roman Engelhardt, Patrick Malcolm, Florian Dandl, and Klaus Bogenberger [2022c]. "Competition and Cooperation of Autonomous Ridepooling Services: Game-Based Simulation of a Broker Concept". In: *Frontiers in Future Transportation* 3. DOI: 10.3389/ffutr.2022.915219

Aledia Bilali, Roman Engelhardt, Florian Dandl, Ulrich Fastenrath, and Klaus Bogenberger [2020]. "Analytical and Agent-Based Model to Evaluate Ride-Pooling Impact Factors". In: *Transportation Research Record: Journal of the Transportation Research Board* 2674.6, pp. 1–12. ISSN: 0361-1981. DOI: 10.1177/0361198120917666

Florian Dandl, Roman Engelhardt, Michael Hyland, Gabriel Tilg, Klaus Bogenberger, and Hani S. Mahmassani [2021b]. "Regulating mobility-on-demand services: Tri-level model and Bayesian optimization solution approach". In: *Transportation Research Part C: Emerging Technologies* 125, p. 103075. ISSN: 0968-090X. DOI: 10.1016/j.trc.2021.103075. URL: https://www.sciencedirect.com/science/article/pii/S0968090X21000991

Gabriel Wilkes, Roman Engelhardt, Lars Briem, Florian Dandl, Peter Vortisch, Klaus Bogenberger, and Martin Kagerbauer [2021]. "Self-Regulating Demand and Supply Equilibrium in Joint Simulation of Travel Demand and a Ride-Pooling Service". In: *Transportation Research Record: Journal of the Transportation Research Board* 2675.8, pp. 226–239. ISSN: 0361-1981. DOI: 10.1177/0361198121997140

Fabian Fehn, Roman Engelhardt, Florian Dandl, Klaus Bogenberger, and Fritz Busch [2023]. "Integrating parcel deliveries into a ride-pooling service—An agent-based simulation study". In: *Transportation Research Part A: Policy and Practice* 169, p. 103580. ISSN: 0965-8564. DOI: 10.1016/j.tra.2022.103580. URL: https://www.sciencedirect.com/science/article/pii/S0965856422003317

Max T.M. Ng, Hani S. Mahmassani, Ömer Verbas, Taner Cokyasar, and Roman Engelhardt [2024]. "Redesigning large-scale multimodal transit networks with shared autonomous mobility services". In: *Transportation Research Part C: Emerging Technologies*, p. 104575. ISSN: 0968-090X. DOI: 10.1016/j.trc.2024.104575. URL: https:

//www.sciencedirect.com/science/article/pii/S0968090X24000962

Conference Proceedings

Roman Engelhardt and Klaus Bogenberger [2021]. "Benefits of Flexible Boarding Locations in On-Demand Ride-Pooling Systems". In: *2021 7th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*. IEEE, pp. 1–6. ISBN: 978-1-7281-8995-6. DOI: 10.1109/MT-ITS49943.2021.9529284

Florian Dandl, Roman Engelhardt, and Klaus Bogenberger [2021a]. "On the Dynamism of User Rejections in Mobility-on-Demand Systems". In: *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*. [Piscataway, NJ]: IEEE, pp. 3399–3404. ISBN: 978-1-7281-9142-3. DOI: 10.1109/ITSC48978.2021.9564918

Fabian Fehn, Roman Engelhardt, and Klaus Bogenberger [2021]. "Ride-Parcel-Pooling - Assessment of the Potential in Combining On-Demand Mobility and City Logistics". In: *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*. [Piscataway, NJ]: IEEE, pp. 3366–3372. ISBN: 978-1-7281-9142-3. DOI: 10.1109/ITSC48978.2021.9564630

Yunfei Zhang, Roman Engelhardt, Arslan-Ali Syed, Florian Dandl, Cornelius Hardt, and Klaus Bogenberger [2022]. "Simulating Charging Processes of Mobility-On-Demand Services at Public Infrastructure: Can Operators Complement Each Other?" In: *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, pp. 2200–2205. ISBN: 978-1-6654-6880-0. DOI: 10.1109/ITSC55140.2022.9922449

Felix Zwick, Gabriel Wilkes, Roman Engelhardt, Steffen Axer, Florian Dandl, Hannes Rewald, Nadine Kostorz, Eva Fraedrich, Martin Kagerbauer, and Kay W. Axhausen [2022]. "Mode choice and ride-pooling simulation: A comparison of mobiTopp, Fleetpy, and MATSim". In: *Procedia Computer Science* 201, pp. 608–613. ISSN: 1877-0509. DOI: 10.1016/j.procs.2022.03.079. URL: https://www.sciencedirect.com/science/article/pii/S1877050922004926

Fynn Wolf, Roman Engelhardt, Yunfei Zhang, Florian Dandl, and Klaus Bogenberger [2023]. "Effects of Dynamic and Stochastic Travel Times on the Operation of Mobility-on-Demand Services". In: *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, pp. 5476–5481. ISBN: 979-8-3503-9946-2. DOI: 10.1109/ITSC57777.2023.10422554

References

Abkarian, Hoseb; Ying Chen; Hani S. Mahmassani (2022a). "Understanding Ridesplitting Behavior with Interpretable Machine Learning Models Using Chicago Transportation Network Company Data". In: *Transportation Research Record: Journal of the Transportation Research Board* 2676.2, pp. 83–99. ISSN: 0361-1981. DOI: 10.1177/03611981211036363.

Abkarian, Hoseb; Hani S. Mahmassani; Michael Hyland (2022b). "Modeling the Mixed-Service Fleet Problem of Shared-Use Autonomous Mobility Systems for On-Demand Ridesourcing and Carsharing With Reservations". In: *Transportation Research Record: Journal of the Transportation Research Board*, p. 036119812210836. ISSN: 0361-1981. DOI: 10.1177/03611981221083617.

Ackermann, Christian; Julia Rieck (2023). "A novel repositioning approach and analysis for dynamic ride-hailing problems". In: *EURO Journal on Transportation and Logistics* 12, p. 100109. ISSN: 2192-4376. DOI: 10.1016/j.ejtl.2023.100109. URL: https://www.sciencedirect.com/science/article/pii/S2192437623000067.

Agatz, Niels; Alan Erera; Martin Savelsbergh; Xing Wang (2012). "Optimization for dynamic ride-sharing: A review". In: *European Journal of Operational Research* 223.2, pp. 295–303. ISSN: 03772217. DOI: 10.1016/j.ejor.2012.05.028. URL: https://www.sciencedirect.com/science/article/pii/S0377221712003864.

Agatz, Niels; Alan L. Erera; Martin W.P. Savelsbergh; Xing Wang (2011). "Dynamic Ride-Sharing: a Simulation Study in Metro Atlanta". In: *Procedia - Social and Behavioral Sciences* 17, pp. 532–550. ISSN: 18770428. DOI: 10.1016/j.sbspro.2011.04.530.

Ahadi, Ramin; Wolfgang Ketter; John Collins; Nicolò Daina (2022). "Cooperative Learning for Smart Charging of Shared Autonomous Vehicle Fleets". In: *Transportation Science*. ISSN: 0041-1655. DOI: 10.1287/trsc.2022.1187.

Alonso-González, María J.; Niels van Oort; Oded Cats; Sascha Hoogendoorn-Lanser; Serge Hoogendoorn (2020). "Value of time and reliability for urban pooled on-demand services". In: *Transportation Research Part C: Emerging Technologies* 115, p. 102621. ISSN: 0968-090X. DOI: 10.1016/j.trc.2020.102621. URL: https://www.sciencedirect.com/science/article/pii/S0968090X1931589X.

Alonso-Mora, Javier; Samitha Samaranayake; Alex Wallar; Emilio Frazzoli; Daniela Rus (2017a). "On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment". In: *Proceedings of the National Academy of Sciences of the United States of America* 114.3, pp. 462–467. DOI: 10.1073/pnas.1611675114.

Alonso-Mora, Javier; Alex Wallar; Daniela Rus (2017b). "Predictive routing for autonomous mobility-on-demand systems with ride-sharing". In: *IROS Vancouver 2017*. Ed. by IEEE/RSJ International Conference on Intelligent Robots and Sys-

TEMS. [Piscataway, NJ]: IEEE, pp. 3583–3590. ISBN: 978-1-5386-2682-5. DOI: 10.1109/IROS.2017.8206203.

ARBIB, JAMES; TONY SEBA (2017). *Rethinking transportation 2020-2030: The disruption of transportation and the collapse of the internal-combustion vehicle and oil industries*. RethinkX Sector Disruption.

ATASOY, BILGE; TAKURO IKEDA; XIANG SONG; MOSHE E. BEN-AKIVA (2015). "The concept and impact analysis of a flexible mobility on demand system". In: *Transportation Research Part C: Emerging Technologies* 56, pp. 373–392. ISSN: 0968-090X. DOI: 10.1016/j.trc.2015.04.009. URL: https://www.sciencedirect.com/science/article/pii/S0968090X15001503.

AUAD-PEREZ, RAMON; PASCAL VAN HENTENRYCK (2022). "Ridesharing and fleet sizing for On-Demand Multimodal Transit Systems". In: *Transportation Research Part C: Emerging Technologies* 138, p. 103594. ISSN: 0968-090X. DOI: 10.1016/j.trc.2022.103594. URL: https://www.sciencedirect.com/science/article/pii/S0968090X22000407.

AULD, JOSHUA; MICHAEL HOPE; HUBERT LEY; VADIM SOKOLOV; BO XU; KUILIN ZHANG (2016). "POLARIS: Agent-based modeling framework development and implementation for integrated travel demand and network and operations simulations". In: *Transportation Research Part C: Emerging Technologies* 64, pp. 101–116. ISSN: 0968-090X. DOI: 10.1016/j.trc.2015.07.017. URL: https://www.sciencedirect.com/science/article/pii/S0968090X15002703.

AUTO MOTOR UND SPORT (2023). "Mercedes mit Level-3-Zulassung". In: URL: https://www.auto-motor-und-sport.de/tech-zukunft/mercedes-autonom-level-3-drive-pilot-haftung-unfall/.

AUTOX (2024). *RoboTaxi – AutoX*. URL: https://www.autox.ai/en/mobility.html.

BCS (2023). *Aktuelle Zahlen und Fakten zum CarSharing in Deutschland*. URL: https://carsharing.de/alles-ueber-carsharing/carsharing-zahlen/aktuelle-zahlen-fakten-zum-carsharing-deutschland.

BECKER, HENRIK et al. (2020). "Impact of vehicle automation and electric propulsion on production costs for mobility services worldwide". In: *Transportation Research Part A: Policy and Practice* 138, pp. 105–126. ISSN: 0965-8564. DOI: 10.1016/j.tra.2020.04.021. URL: https://www.sciencedirect.com/science/article/pii/S0965856420305772.

BELLMAN, RICHARD (1957). *Dynamic programming*. Rand Corporation research study. Princeton: Princeton University Press. ISBN: 9780691079516. URL: https://books.google.de/books?id=wdtoPwAACAAJ.

BENT, RUSSELL W.; PASCAL VAN HENTENRYCK (2004). "Scenario-Based Planning for Partially Dynamic Vehicle Routing with Stochastic Customers". In: *Operations Research* 52.6, pp. 977–987. ISSN: 0030-364X. DOI: 10.1287/opre.1040.0124.

BILALI, ALEDIA; FLORIAN DANDL; ULRICH FASTENRATH; KLAUS BOGENBERGER (2019a). "An Analytical Model for On-Demand Ride Sharing to Evaluate the Impact of Reservation, Detour and Maximum Waiting Time". In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, pp. 1715–1720. ISBN: 978-1-5386-7024-8. DOI: 10.1109/ITSC.2019.8917280.

BILALI, ALEDIA; FLORIAN DANDL; ULRICH FASTENRATH; KLAUS BOGENBERGER (2019b). "Impact of service quality factors on ride sharing in urban areas". In: *2019 6th International*

Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS). IEEE, pp. 1–8. ISBN: 978-1-5386-9484-8. DOI: 10.1109/MTITS.2019.8883364.

Bilali, Aledia; Roman Engelhardt; Florian Dandl; Ulrich Fastenrath; Klaus Bogenberger (2020). "Analytical and Agent-Based Model to Evaluate Ride-Pooling Impact Factors". In: *Transportation Research Record: Journal of the Transportation Research Board* 2674.6, pp. 1–12. ISSN: 0361-1981. DOI: 10.1177/0361198120917666.

Bischoff, Joschka; Michal Maciejewski (2016). "Simulation of City-wide Replacement of Private Cars with Autonomous Taxis in Berlin". In: *Procedia Computer Science* 83, pp. 237–244. ISSN: 1877-0509. DOI: 10.1016/j.procs.2016.04.121. URL: https://www.sciencedirect.com/science/article/pii/S1877050916301442.

Bischoff, Joschka; Michal Maciejewski (2020). "Proactive empty vehicle rebalancing for Demand Responsive Transport services". In: *Procedia Computer Science* 170, pp. 739–744. ISSN: 1877-0509. DOI: 10.1016/j.procs.2020.03.162. URL: https://www.sciencedirect.com/science/article/pii/S1877050920306220.

Bischoff, Joschka; Michal Maciejewski; Kai Nagel (2017). "City-wide shared taxis: A simulation study in Berlin". In: *IEEE ITSC 2017*. Piscataway, NJ: IEEE, pp. 275–280. ISBN: 978-1-5386-1526-3. DOI: 10.1109/ITSC.2017.8317926.

Boeing, Geoff (2017). "OSMnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks". In: *Computers, Environment and Urban Systems* 65, pp. 126–139. ISSN: 0198-9715. DOI: 10.1016/j.compenvurbsys.2017.05.004. URL: https://www.sciencedirect.com/science/article/pii/S0198971516303970.

Boesch, Patrick M.; Francesco Ciari; Kay W. Axhausen (2016). "Autonomous Vehicle Fleet Sizes Required to Serve Different Levels of Demand". In: *Transportation Research Record: Journal of the Transportation Research Board* 2542.1, pp. 111–119. ISSN: 0361-1981. DOI: 10.3141/2542-13.

Bösch, Patrick M.; Felix Becker; Henrik Becker; Kay W. Axhausen (2018). "Cost-based analysis of autonomous mobility services". In: *Transport Policy* 64, pp. 76–91. ISSN: 0967-070X. DOI: 10.1016/j.tranpol.2017.09.005. URL: https://www.sciencedirect.com/science/article/pii/S0967070X17300811.

Bracher, Benedikt Andreas (2019). "Intelligente verkehrsabhängige Steuerung einer Citymaut". PhD Thesis. Universität der Bundeswehr München. URL: https://athene-forschung.unibw.de/92528?query=Bracher&show_id=129898.

Brar, Avalpreet Singh; Rong Su (2021). "Dynamic Supply-Demand Balancing Policy for CMoD Fleet". In: *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*. [Piscataway, NJ]: IEEE, pp. 2435–2440. ISBN: 978-1-7281-9142-3. DOI: 10.1109/ITSC48978.2021.9564845.

Brotcorne, Luce; Gilbert Laporte; Frédéric Semet (2003). "Ambulance location and relocation models". In: *European Journal of Operational Research* 147.3, pp. 451–463. ISSN: 03772217. DOI: 10.1016/S0377-2217(02)00364-8. URL: https://www.sciencedirect.com/science/article/pii/S0377221702003648.

Bujak, Michal; Rafal Kucharski (2023). "Network structures of urban ride-pooling problems and their properties". In: *Social Network Analysis and Mining* 13.1, pp. 1–13. ISSN: 1869-5469. DOI: 10.1007/s13278-023-01094-9. URL: https://link.springer.com/article/10.1007/s13278-023-01094-9.

Bundesverband CarSharing e.V. (2023). *Carsharing legt kräftig zu*. URL: https://carsharing.de/carsharing-legt-kraeftig-zu.

Calvert, S. C.; W. J. Schakel; J. W. C. van Lint (2017). "Will Automated Vehicles Negatively Impact Traffic Flow?" In: *Journal of Advanced Transportation* 2017, pp. 1–17. ISSN: 0197-6729. DOI: 10.1155/2017/3082781.

Castillo, Juan Camilo; Dan Knoepfle; Glen Weyl (2017). "Surge Pricing Solves the Wild Goose Chase". In: *Proceedings of the 2017 ACM Conference on Economics and Computation*. Ed. by Constantinos Daskalakis. New York NY: ACM, pp. 241–242. ISBN: 9781450345279. DOI: 10.1145/3033274.3085098.

Chen, M. Keith; Judith Chevalier; Peter Rossi; Emily Oehlsen (2017). *The Value of Flexible Work: Evidence from Uber Drivers*. Cambridge, MA. DOI: 10.3386/w23296.

Chen, T. Donna; Kara M. Kockelman; Josiah P. Hanna (2016). "Operations of a shared, autonomous, electric vehicle fleet: Implications of vehicle & charging infrastructure decisions". In: *Transportation Research Part A: Policy and Practice* 94, pp. 243–254. ISSN: 0965-8564. DOI: 10.1016/j.tra.2016.08.020. URL: https://www.sciencedirect.com/science/article/pii/S096585641630756X.

Cheng Li; David Parker; Qi Hao (2022). "A value-based dynamic learning approach for vehicle dispatch in ride-sharing". In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2022)*. URL: https://research.birmingham.ac.uk/en/publications/a-value-based-dynamic-learning-approach-for-vehicle-dispatch-in-r.

Chicago Department of Business Affairs & Consumer Protection (2022). *Chicago TNC Data*. URL: https://data.cityofchicago.org/Transportation/Transportation-Network-Providers-Trips-2022/2tdj-ffvb.

Chouaki, Tarek; Sebastian Hörl; Jakob Puchinger (2022). "Implementing reinforcement learning for on-demand vehicle rebalancing in MATSim". In: *Procedia Computer Science* 201, pp. 134–141. ISSN: 1877-0509. DOI: 10.1016/j.procs.2022.03.020. URL: https://www.sciencedirect.com/science/article/pii/S187705092200432X.

City of New York (2024). *TLC Trip Record Data - TLC*. URL: https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page.

Compostella, Junia; Lewis M. Fulton; Robert de Kleine; Hyung Chul Kim; Timothy J. Wallington (2020). "Near- (2020) and long-term (2030–2035) costs of automated, electrified, and shared mobility in the United States". In: *Transport Policy* 85, pp. 54–66. ISSN: 0967-070X. DOI: 10.1016/j.tranpol.2019.10.001. URL: https://www.sciencedirect.com/science/article/pii/S0967070X18309119.

Cordeau, Jean-François (2006). "A Branch-and-Cut Algorithm for the Dial-a-Ride Problem". In: *Operations Research* 54.3, pp. 573–586. ISSN: 0030-364X. DOI: 10.1287/opre.1060.0283.

Cordeau, Jean-François; Gilbert Laporte (2003). "A tabu search heuristic for the static multi-vehicle dial-a-ride problem". In: *Transportation Research Part B: Methodological* 37.6, pp. 579–594. ISSN: 0191-2615. DOI: 10.1016/S0191-2615(02)00045-0. URL: https://www.sciencedirect.com/science/article/pii/S0191261502000450.

Cortina, Mélanie; Nicolas Chiabaut; Ludovic Leclercq (2023). "Fostering synergy between transit and Autonomous Mobility-on-Demand systems: A dynamic modeling approach for the morning commute problem". In: *Transportation Research Part A: Policy and Practice* 170, p. 103638. ISSN: 0965-8564. DOI: 10.1016/j.tra.2023.103638. URL: https://www.sciencedirect.com/science/article/pii/S0965856423000587.

Cruise (2024). *Cruise Driverless Rides | Autonomous Vehicles | Self-Driving Cars*. URL: https://www.getcruise.com/.

Cui, Hongjun; Yizhe Yang; Minqing Zhu; Xinwei Ma; Xiuyong Chen; Binghui Qie (2023). "The scheduling methods with different demand priorities for shared autonomous vehicle system in hybrid demands mode considering dynamic travel time". In: *Physica A: Statistical Mechanics and its Applications* 632, p. 129325. ISSN: 0378-4371. DOI: 10.1016/j.physa.2023.129325. URL: https://www.sciencedirect.com/science/article/pii/S0378437123008804.

Dandl, Florian (2022). "Operation and Regulation of Autonomous Mobility-on-Demand Systems". PhD thesis. Technische Universität München. URL: https://mediatum.ub.tum.de/1639389.

Dandl, Florian; Klaus Bogenberger (2019). "Comparing Future Autonomous Electric Taxis With an Existing Free-Floating Carsharing System". In: *IEEE Transactions on Intelligent Transportation Systems* 20.6, pp. 2037–2047. ISSN: 1524-9050. DOI: 10.1109/TITS.2018.2857208.

Dandl, Florian; Benedikt Bracher; Klaus Bogenberger (2017). "Microsimulation of an autonomous taxi-system in Munich". In: *2017 5th IEEE International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*. IEEE, pp. 833–838. ISBN: 978-1-5090-6484-7. DOI: 10.1109/MTITS.2017.8005628.

Dandl, Florian; Roman Engelhardt; Klaus Bogenberger (2021a). "On the Dynamism of User Rejections in Mobility-on-Demand Systems". In: *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*. [Piscataway, NJ]: IEEE, pp. 3399–3404. ISBN: 978-1-7281-9142-3. DOI: 10.1109/ITSC48978.2021.9564918.

Dandl, Florian; Roman Engelhardt; Michael Hyland; Gabriel Tilg; Klaus Bogenberger; Hani S. Mahmassani (2021b). "Regulating mobility-on-demand services: Tri-level model and Bayesian optimization solution approach". In: *Transportation Research Part C: Emerging Technologies* 125, p. 103075. ISSN: 0968-090X. DOI: 10.1016/j.trc.2021.103075. URL: https://www.sciencedirect.com/science/article/pii/S0968090X21000991.

Dandl, Florian; Fabian Fehn; Klaus Bogenberger; Fritz Busch (2020a). "Pre-Day Scheduling of Charging Processes in Mobility-on-Demand Systems Considering Electricity Price and Vehicle Utilization Forecasts". In: *2020 Forum on Integrated and Sustainable Transportation Systems (FISTS)*. IEEE, pp. 127–134. ISBN: 978-1-7281-9503-2. DOI: 10.1109/FISTS46898.2020.9264862.

Dandl, Florian; Michael Hyland; Klaus Bogenberger; Hani S. Mahmassani (2019). "Evaluating the impact of spatio-temporal demand forecast aggregation on the operational performance of shared autonomous mobility fleets". In: *Transportation* 46.6, pp. 1975–1996. ISSN: 1572-9435. DOI: 10.1007/s11116-019-10007-9. URL: https://link.springer.com/article/10.1007/s11116-019-10007-9.

Dandl, Florian; Michael Hyland; Klaus Bogenberger; Hani S. Mahmassani (2020b). "Dual-horizon forecasts and repositioning strategies for operating shared autonomous mobility fleets". In: *99th Annual Meeting of the Transportation Research Board (TRB 2020)*.

Dantzig, G. B.; J. H. Ramser (1959). "The Truck Dispatching Problem". In: *Management Science* 6.1, pp. 80–91. ISSN: 0025-1909. DOI: 10.1287/mnsc.6.1.80.

Dean, Matthew D.; Krishna Murthy Gurumurthy; Felipe de Souza; Joshua Auld; Kara M. Kockelman (2022). "Synergies between repositioning and charging strategies for shared autonomous electric vehicle fleets". In: *Transportation Research Part D: Transport and Environment* 108, p. 103314. ISSN: 1361-9209. DOI: 10.1016/j.trd.2022.103314. URL: https://www.sciencedirect.com/science/article/pii/S1361920922001420.

Dell'Amico, Mauro; Eleni Hadjicostantinou; Manuel Iori; Stefano Novellani (2014). "The bike sharing rebalancing problem: Mathematical formulations and benchmark instances". In: *Omega* 45, pp. 7–19. ISSN: 0305-0483. DOI: 10.1016/j.omega.2013.12.001. URL: https://www.sciencedirect.com/science/article/pii/S0305048313001187.

Duan, Leyi; Yuguang Wei; Jinchuan Zhang; Yang Xia (2020). "Centralized and decentralized autonomous dispatching strategy for dynamic autonomous taxi operation in hybrid request mode". In: *Transportation Research Part C: Emerging Technologies* 111, pp. 397–420. ISSN: 0968-090X. DOI: 10.1016/j.trc.2019.12.020. URL: https://www.sciencedirect.com/science/article/pii/S0968090X19306710.

Duan, Leyi; Yuguang Wei; Jinchuan Zhang; Yang Xia (2023). "Addressing the urban-scale vehicle assignment and rebalancing problems in shared autonomous vehicle system while simultaneously considering immediate, reservation, shareable, and unshareable requests". In: *Computers & Industrial Engineering* 177, p. 109025. ISSN: 0360-8352. DOI: 10.1016/j.cie.2023.109025. URL: https://www.sciencedirect.com/science/article/pii/S0360835223000499.

Engelhardt, Roman; Klaus Bogenberger (2021). "Benefits of Flexible Boarding Locations in On-Demand Ride-Pooling Systems". In: *2021 7th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*. IEEE, pp. 1–6. ISBN: 978-1-7281-8995-6. DOI: 10.1109/MT-ITS49943.2021.9529284.

Engelhardt, Roman; Florian Dandl; Aledia Bilali; Klaus Bogenberger (2019). "Quantifying the Benefits of Autonomous On-Demand Ride-Pooling: A Simulation Study for Munich, Germany". In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, pp. 2992–2997. ISBN: 978-1-5386-7024-8. DOI: 10.1109/ITSC.2019.8916955.

Engelhardt, Roman; Florian Dandl; Klaus Bogenberger (2020). *Speed-up Heuristic for an On-Demand Ride-Pooling Algorithm*. URL: https://arxiv.org/pdf/2007.14877.

Engelhardt, Roman; Florian Dandl; Klaus Bogenberger (2022a). *Simulating Ride-Pooling Services with Pre-Booking and On-Demand Customers*. URL: http://arxiv.org/pdf/2210.06972v1.

Engelhardt, Roman; Florian Dandl; Arslan-Ali Syed; Yunfei Zhang; Fabian Fehn; Fynn Wolf; Klaus Bogenberger (2022b). *FleetPy: A Modular Open-Source*

Simulation Tool for Mobility On-Demand Services. URL: https://arxiv.org/pdf/2207.14246.

Engelhardt, Roman; Hani S. Mahmassani; Klaus Bogenberger (2023). *Predictive Vehicle Repositioning for On-Demand Ride-Pooling Services*. URL: http://arxiv.org/pdf/2308.05507v1.

Engelhardt, Roman; Hani S. Mahmassani; Klaus Bogenberger (2024). "Vorhersagebasierte Fahrzeugrepositionierung für On-Demand-Ride-Pooling-Dienste". In: *Straßenverkehrstechnik* 06, pp. 467–474. ISSN: 0039-2219. DOI: 10.53184/SVT6-2024-3.

Engelhardt, Roman; Patrick Malcolm; Florian Dandl; Klaus Bogenberger (2022c). "Competition and Cooperation of Autonomous Ridepooling Services: Game-Based Simulation of a Broker Concept". In: *Frontiers in Future Transportation* 3. DOI: 10.3389/ffutr.2022.915219.

Erdmann, Marvin; Florian Dandl; Klaus Bogenberger (2021). "Combining immediate customer responses and car–passenger reassignments in on-demand mobility services". In: *Transportation Research Part C: Emerging Technologies* 126, p. 103104. ISSN: 0968-090X. DOI: 10.1016/j.trc.2021.103104. URL: https://www.sciencedirect.com/science/article/pii/S0968090X21001248.

Erhardt, Gregory D.; Sneha Roy; Drew Cooper; Bhargava Sana; Mei Chen; Joe Castiglione (2019). "Do transportation network companies decrease or increase congestion?" In: *Science advances* 5.5, eaau2670. DOI: 10.1126/sciadv.aau2670.

Estandia, Alvaro; Maximilian Schiffer; Federico Rossi; Justin Luke; Emre Can Kara; Ram Rajagopal; Marco Pavone (2021). "On the Interaction Between Autonomous Mobility on Demand Systems and Power Distribution Networks—An Optimal Power Flow Approach". In: *IEEE Transactions on Control of Network Systems* 8.3, pp. 1163–1176. DOI: 10.1109/TCNS.2021.3059225.

European Commission (2023). *Delivering the European Green Deal*. URL: https://commission.europa.eu/strategy-and-policy/priorities-2019-2024/european-green-deal/delivering-european-green-deal_en.

Fageda, Xavier (2021). "Measuring the impact of ride–hailing firms on urban congestion: The case of Uber in Europe". In: *Papers in Regional Science* 100.5, pp. 1230–1254. ISSN: 1056-8190. DOI: 10.1111/pirs.12607. URL: https://www.sciencedirect.com/science/article/pii/S1056819023001410.

Fagnant, Daniel J.; Kara M. Kockelman (2014). "The travel and environmental implications of shared autonomous vehicles, using agent-based model scenarios". In: *Transportation Research Part C: Emerging Technologies* 40, pp. 1–13. ISSN: 0968-090X. DOI: 10.1016/j.trc.2013.12.001. URL: https://www.sciencedirect.com/science/article/pii/S0968090X13002581.

Fagnant, Daniel J.; Kara M. Kockelman (2018). "Dynamic ride-sharing and fleet sizing for a system of shared autonomous vehicles in Austin, Texas". In: *Transportation* 45.1, pp. 143–158. ISSN: 1572-9435. DOI: 10.1007/s11116-016-9729-z. URL: https://link.springer.com/article/10.1007/s11116-016-9729-z.

Fehn, Fabian; Roman Engelhardt; Klaus Bogenberger (2021). "Ride-Parcel-Pooling - Assessment of the Potential in Combining On-Demand Mobility and City Logistics". In: *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*.

[Piscataway, NJ]: IEEE, pp. 3366–3372. ISBN: 978-1-7281-9142-3. DOI: 10.1109/ITSC48978.2021.9564630.

Fehn, Fabian; Roman Engelhardt; Florian Dandl; Klaus Bogenberger; Fritz Busch (2023). "Integrating parcel deliveries into a ride-pooling service—An agent-based simulation study". In: *Transportation Research Part A: Policy and Practice* 169, p. 103580. ISSN: 0965-8564. DOI: 10.1016/j.tra.2022.103580. URL: https://www.sciencedirect.com/science/article/pii/S0965856422003317.

Fiedler, David; Michal Čertický; Javier Alonso-Mora; Michal Čáp (2018). "The Impact of Ridesharing in Mobility-on-Demand Systems: Simulation Case Study in Prague". In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1173–1178. ISBN: 2153-0017. DOI: 10.1109/ITSC.2018.8569451.

Fiedler, David; Michal Čertický; Javier Alonso-Mora; Michal Pěchouček; Michal Čáp (2022). "Large-scale online ridesharing: the effect of assignment optimality on system performance". In: *Journal of Intelligent Transportation Systems*, pp. 1–22. ISSN: 1547-2450. DOI: 10.1080/15472450.2022.2121651.

Fielbaum, Andres; Javier Alonso-Mora (2024). "Design of mixed fixed-flexible bus public transport networks by tracking the paths of on-demand vehicles". In: *Transportation Research Part C: Emerging Technologies*, p. 104580. ISSN: 0968-090X. DOI: 10.1016/j.trc.2024.104580. URL: https://www.sciencedirect.com/science/article/pii/S0968090X24001013.

Fielbaum, Andres; Xiaoshan Bai; Javier Alonso-Mora (2021). "On-demand ridesharing with optimized pick-up and drop-off walking locations". In: *Transportation Research Part C: Emerging Technologies* 126, p. 103061. ISSN: 0968-090X. DOI: 10.1016/j.trc.2021.103061. URL: https://www.sciencedirect.com/science/article/pii/S0968090X21000887.

Foljanty, Lukas (2022). "The On-Demand Ridepooling Market in 2022 – further growth or signs of saturation?" In: URL: https://www.linkedin.com/pulse/on-demand-ridepooling-market-2022-further-growth-signs-lukas-foljanty/?trackingId=eBlCzqljSM6nADFpAy%2FzzQ%3D%3D.

Fortune (2024). "Exclusive: Mercedes becomes the first automaker to sell autonomous cars in the U.S. that don't come with a requirement that drivers watch the road". In: *Fortune*. URL: https://fortune.com/2024/04/18/mercedes-self-driving-autonomous-cars-california-nevada-level-3-drive-pilot/.

Frei, Charlotte; Michael Hyland; Hani S. Mahmassani (2017). "Flexing service schedules: Assessing the potential for demand-adaptive hybrid transit via a stated preference approach". In: *Transportation Research Part C: Emerging Technologies* 76, pp. 71–89. ISSN: 0968-090X. DOI: 10.1016/j.trc.2016.12.017. URL: https://www.sciencedirect.com/science/article/pii/S0968090X1630273X.

Friedrich, Bernhard (2016). "The Effect of Autonomous Vehicles on Traffic". In: *Autonomous Driving*. Ed. by Markus Maurer; J. Christian Gerdes; Barbara Lenz; Hermann Winner. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 317–334. ISBN: 978-3-662-48847-8. DOI: 10.1007/978-3-662-48847-8{\textunderscore}16. URL: https://link.springer.com/chapter/10.1007/978-3-662-48847-8_16.

Friedrich, Markus; Maximilian Hartl (2016). *MEGAFON - Modellergebnisse geteilter autonomer Fahrzeugflotten des oeffentlichen Nahverkehrs*.

Furuhata, Masabumi; Maged Dessouky; Fernando Ordóñez; Marc-Etienne Brunet; Xiaoqing Wang; Sven Koenig (2013). "Ridesharing: The state-of-the-art and future directions". In: *Transportation Research Part B: Methodological* 57, pp. 28–46. ISSN: 0191-2615. DOI: 10.1016/j.trb.2013.08.012. URL: https://www.sciencedirect.com/science/article/pii/S0191261513001483.

Gao, Xuehong (2022). "A bi-level stochastic optimization model for multi-commodity rebalancing under uncertainty in disaster response". In: *Annals of Operations Research* 319.1, pp. 115–148. ISSN: 1572-9338. DOI: 10.1007/s10479-019-03506-6. URL: https://link.springer.com/article/10.1007/s10479-019-03506-6.

Ghandeharioun, Zahra; Anastasios Kouvelas (2023). "Real-time ridesharing operations for on-demand capacitated systems considering dynamic travel time information". In: *Transportation Research Part C: Emerging Technologies* 151, p. 104115. ISSN: 0968-090X. DOI: 10.1016/j.trc.2023.104115. URL: https://www.sciencedirect.com/science/article/pii/S0968090X23001043.

Goel, Preeti; Lars Kulik; Kotagiri Ramamohanarao (2017). "Optimal Pick up Point Selection for Effective Ride Sharing". In: *IEEE Transactions on Big Data* 3.2, pp. 154–168. DOI: 10.1109/TBDATA.2016.2599936.

Gomez, Juan; Álvaro Aguilera-García; Felipe F. Dias; Chandra R. Bhat; José Manuel Vassallo (2021). "Adoption and frequency of use of ride-hailing services in a European city: The case of Madrid". In: *Transportation Research Part C: Emerging Technologies* 131, p. 103359. ISSN: 0968-090X. DOI: 10.1016/j.trc.2021.103359. URL: https://www.sciencedirect.com/science/article/pii/S0968090X21003612.

Göransson, Jessica; Henrik Andersson (2023). "Factors that make public transport systems attractive: a review of travel preferences and travel mode choices". In: *European Transport Research Review* 15.1, pp. 1–14. ISSN: 1866-8887. DOI: 10.1186/s12544-023-00609-x. URL: https://link.springer.com/article/10.1186/s12544-023-00609-x.

Gueriau, Maxime; Federico Cugurullo; Ransford A. Acheampong; Ivana Dusparic (2020). "Shared Autonomous Mobility on Demand: A Learning-Based Approach and Its Performance in the Presence of Traffic Congestion". In: *IEEE Intelligent Transportation Systems Magazine* 12.4, pp. 208–218. ISSN: 1939-1390. DOI: 10.1109/MITS.2020.3014417.

Guo, Xiaotong; Qingyi Wang; Jinhua Zhao (2022). "Data-Driven Vehicle Rebalancing With Predictive Prescriptions in the Ride-Hailing System". In: *IEEE Open Journal of Intelligent Transportation Systems* 3, pp. 251–266. DOI: 10.1109/OJITS.2022.3163180.

Guo, Yuhan; Yu Zhang; Youssef Boulaksil (2021). "Real-time ride-sharing framework with dynamic timeframe and anticipation-based migration". In: *European Journal of Operational Research* 288.3, pp. 810–828. ISSN: 03772217. DOI: 10.1016/j.ejor.2020.06.038. URL: https://www.sciencedirect.com/science/article/pii/S0377221720305816.

Gurumurthy, Krishna Murthy; Felipe de Souza; Annesha Enam; Joshua Auld (2020). "Integrating Supply and Demand Perspectives for a Large-Scale Simulation of Shared

Autonomous Vehicles". In: *Transportation Research Record: Journal of the Transportation Research Board* 2674.7, pp. 181–192. ISSN: 0361-1981. DOI: 10.1177/0361198120921157.

Hall, Jonathan D.; Craig Palsson; Joseph Price (2018). "Is Uber a substitute or complement for public transit?" In: *Journal of Urban Economics* 108, pp. 36–50. ISSN: 0094-1190. DOI: 10.1016/j.jue.2018.09.003. URL: https://www.sciencedirect.com/science/article/pii/S0094119018300731.

Harmann, Dennis; Sefa Yilmaz-Niewerth; Christina Jacob (2022). "Methodological Distribution of Virtual Stops for Ridepooling". In: *Transportation Research Procedia* 62, pp. 442–449. ISSN: 2352-1465. DOI: 10.1016/j.trpro.2022.02.055. URL: https://www.sciencedirect.com/science/article/pii/S235214652200182X.

Henao, Alejandro; Wesley E. Marshall (2019). "The impact of ride-hailing on vehicle miles traveled". In: *Transportation* 46.6, pp. 2173–2194. ISSN: 1572-9435. DOI: 10.1007/s11116-018-9923-2. URL: https://link.springer.com/article/10.1007/s11116-018-9923-2.

Hörl, S.; C. Ruch; F. Becker; E. Frazzoli; K. W. Axhausen (2019). "Fleet operational policies for automated mobility: A simulation assessment for Zurich". In: *Transportation Research Part C: Emerging Technologies* 102, pp. 20–31. ISSN: 0968-090X. DOI: 10.1016/j.trc.2019.02.020. URL: https://www.sciencedirect.com/science/article/pii/S0968090X18304248.

Horni, Andreas; Kai Nagel; Kay W. Axhausen (2016). *The Multi-Agent Transport Simulation MATSim*. Ubiquity Press. DOI: 10.5334/baw. URL: https://library.oapen.org/handle/20.500.12657/32162.

Hosni, Hadi; Joe Naoum-Sawaya; Hassan Artail (2014). "The shared-taxi problem: Formulation and solution methods". In: *Transportation Research Part B: Methodological* 70, pp. 303–318. ISSN: 0191-2615. DOI: 10.1016/j.trb.2014.09.011. URL: https://www.sciencedirect.com/science/article/pii/S0191261514001659.

Huang, Chenn-Jung; Kai-Wen Hu; Cheng-Yang Hsieh (2022). "Congestion-Aware Rideshare Dispatch for Shared Autonomous Electric Vehicle Fleets". In: *Electronics* 11.16, p. 2591. DOI: 10.3390/electronics11162591.

Huang, Yan; Ruoming Jin; Favyen Bastani; Xiaoyang Sean Wang (2013). *Large Scale Real-time Ridesharing with Service Guarantee on Road Networks*. URL: https://arxiv.org/pdf/1302.6666.

Hyland, Michael; Hani S. Mahmassani (2018). "Dynamic autonomous vehicle fleet operations: Optimization-based strategies to assign AVs to immediate traveler demand requests". In: *Transportation Research Part C: Emerging Technologies* 92, pp. 278–297. ISSN: 0968-090X. DOI: 10.1016/j.trc.2018.05.003. URL: https://www.sciencedirect.com/science/article/pii/S0968090X18306028.

Hyland, Michael; Hani S. Mahmassani (2020). "Operational benefits and challenges of shared-ride automated mobility-on-demand services". In: *Transportation Research Part A: Policy and Practice* 134, pp. 251–270. ISSN: 0965-8564. DOI: 10.1016/j.tra.2020.02.017. URL: https://www.sciencedirect.com/science/article/pii/S0965856419307888.

Hyland, Michael F.; Hani S. Mahmassani (2017). "Taxonomy of Shared Autonomous Vehicle Fleet Management Problems to Inform Future Transportation Mobility". In: *Trans-*

portation Research Record: Journal of the Transportation Research Board 2653.1, pp. 26–34. ISSN: 0361-1981. DOI: 10.3141/2653-04.

Hyytiä, Esa; Aleksi Penttinen; Reijo Sulonen (2012). "Non-myopic vehicle and route selection in dynamic DARP with travel time and workload objectives". In: *Computers & Operations Research* 39.12, pp. 3021–3030. ISSN: 0305-0548. DOI: 10.1016/j.cor.2012.03.002. URL: https://www.sciencedirect.com/science/article/pii/S030505481200055X.

IEA (2024). *Trends in electric cars – Global EV Outlook 2024 – Analysis - IEA*. URL: https://www.iea.org/reports/global-ev-outlook-2024/trends-in-electric-cars.

Iglesias, Ramon; Federico Rossi; Kevin Wang; David Hallac; Jure Leskovec; Marco Pavone (2018). "Data-Driven Model Predictive Control of Autonomous Mobility-on-Demand Systems". In: *ICRA*. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, pp. 1–7. ISBN: 978-1-5386-3081-5. DOI: 10.1109/ICRA.2018.8460966.

Illgen, Stefan; Michael Höck (2019). "Literature review of the vehicle relocation problem in one-way car sharing networks". In: *Transportation Research Part B: Methodological* 120, pp. 193–204. ISSN: 0191-2615. DOI: 10.1016/j.trb.2018.12.006. URL: https://www.sciencedirect.com/science/article/pii/S0191261518300547.

ITF (2015). *Urban Mobility System Upgrade: How shared self-driving cars could change city traffic*. International Transport Forum Policy Papers. DOI: 10.1787/5jlwvzdk29g5-en.

Jain, Siddhartha; Pascal van Hentenryck (2011). "Large Neighborhood Search for Dial-a-Ride Problems". In: Springer, Berlin, Heidelberg, pp. 400–413. DOI: 10.1007/978-3-642-23786-7{\textunderscore}31. URL: https://link.springer.com/chapter/10.1007/978-3-642-23786-7_31.

Jaw, Jang-Jei; Amedeo R. Odoni; Harilaos N. Psaraftis; Nigel H.M. Wilson (1986). "A heuristic algorithm for the multi-vehicle advance request dial-a-ride problem with time windows". In: *Transportation Research Part B: Methodological* 20.3, pp. 243–257. ISSN: 0191-2615. DOI: 10.1016/0191-2615(86)90020-2. URL: https://www.sciencedirect.com/science/article/pii/0191261586900202.

Joel Jaeger (2023). *These Countries Are Adopting Electric Vehicles the Fastest*. URL: https://www.wri.org/insights/countries-adopting-electric-vehicles-fastest.

Jung, Jaeyoung; R. Jayakrishnan; Ji Young Park (2016). "Dynamic Shared-Taxi Dispatch Algorithm with Hybrid-Simulated Annealing". In: *Computer-Aided Civil and Infrastructure Engineering* 31.4, pp. 275–291. ISSN: 1467-8667. DOI: 10.1111/mice.12157.

Kagerbauer, Martin; Nadine Kostorz; Gabriel Wilkes; Florian Dandl; Roman Engelhardt; Ulrich Glöckl; Eva Fraedrich; Felix Zwick (2021). *Ridepooling in der Modellierung des Gesamtverkehrs - Methodenbericht zur MOIA Begleitforschung*. Karlsruher Institut für Technologie (KIT). DOI: 10.5445/IR/1000141282. URL: https://publikationen.bibliothek.kit.edu/1000141282.

Karaenke, Paul; Maximilian Schiffer; Stefan Waldherr (2023). "On the benefits of ex-post pricing for ride-pooling". In: *Transportation Research Part C: Emerging Technologies* 155, p. 104290. ISSN: 0968-090X. DOI: 10.1016/j.trc.2023.104290. URL: https://www.sciencedirect.com/science/article/pii/S0968090X23002796.

Kesting, Arne; Martin Treiber; Dirk Helbing (2010). "Enhanced intelligent driver model to access the impact of driving strategies on traffic capacity". In: *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences* 368.1928, pp. 4585–4605. ISSN: 1364-503X. DOI: 10.1098/rsta.2010.0084.

Kolodny, Lora (2022). "Cruise gets green light for commercial robotaxi service in San Francisco". In: *CNBC*. URL: https://www.cnbc.com/2022/06/02/cruise-gets-green-light-for-commercial-robotaxis-in-san-francisco.html.

Kondor, Dániel; Iva Bojic; Giovanni Resta; Fábio Duarte; Paolo Santi; Carlo Ratti (2022). "The cost of non-coordination in urban on-demand mobility". In: *Scientific Reports* 12.1, p. 4669. ISSN: 2045-2322. DOI: 10.1038/s41598-022-08427-2. URL: https://www.nature.com/articles/s41598-022-08427-2.

König, Alexandra; Jan Grippenkoven (2020). "Modelling travelers' appraisal of ridepooling service characteristics with a discrete choice experiment". In: *European Transport Research Review* 12.1, pp. 1–11. ISSN: 1866-8887. DOI: 10.1186/s12544-019-0391-3. URL: https://etrr.springeropen.com/articles/10.1186/s12544-019-0391-3.

Kucharski, Rafal; Oded Cats (2020). "Exact matching of attractive shared rides (ExMAS) for system-wide strategic evaluations". In: *Transportation Research Part B: Methodological* 139, pp. 285–310. ISSN: 0191-2615. DOI: 10.1016/j.trb.2020.06.006. URL: https://www.sciencedirect.com/science/article/pii/S0191261520303465.

Kumar, Pramesh; Alireza Khani (2022). "Planning of integrated mobility-on-demand and urban transit networks". In: *Transportation Research Part A: Policy and Practice* 166, pp. 499–521. ISSN: 0965-8564. DOI: 10.1016/j.tra.2022.11.001. URL: https://www.sciencedirect.com/science/article/pii/S0965856422002841.

Lee, Gaeun; Jun Soo Lee; Kun Soo Park (2024). "Battery swapping, vehicle rebalancing, and staff routing for electric scooter sharing systems". In: *Transportation Research Part E: Logistics and Transportation Review* 186, p. 103540. ISSN: 1366-5545. DOI: 10.1016/j.tre.2024.103540. URL: https://www.sciencedirect.com/science/article/pii/S1366554524001315.

Levin, Michael W.; Kara M. Kockelman; Stephen D. Boyles; Tianxin Li (2017). "A general framework for modeling shared autonomous vehicles with dynamic network-loading and dynamic ride-sharing application". In: *Computers, Environment and Urban Systems* 64, pp. 373–383. ISSN: 0198-9715. DOI: 10.1016/j.compenvurbsys.2017.04.006. URL: https://www.sciencedirect.com/science/article/pii/S019897151630237X.

Li, Cheng; David Parker; Qi Hao (2021). "Optimal Online Dispatch for High-Capacity Shared Autonomous Mobility-on-Demand Systems". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 779–785. ISBN: 2577-087X. DOI: 10.1109/ICRA48506.2021.9561281.

Li, Donghui; Constantinos Antoniou; Hai Jiang; Qianyan Xie; Wei Shen; Weijian Han (2019a). "The Value of Prepositioning in Smartphone-Based Vanpool Services under Stochastic Requests and Time-Dependent Travel Times". In: *Transportation Research Record: Journal of the Transportation Research Board* 2673.2, pp. 26–37. ISSN: 0361-1981. DOI: 10.1177/0361198118822815.

Li, Sen; Hamidreza Tavafoghi; Kameshwar Poolla; Pravin Varaiya (2019b). "Regulating TNCs: Should Uber and Lyft set their own rules?" In: *Transportation Research Part B: Methodological* 129, pp. 193–225. ISSN: 0191-2615. DOI: 10.1016/j.trb.2019.09.008. URL: https://www.sciencedirect.com/science/article/pii/S0191261519300669.

Lim, Linda; Aly M. Tawfik (2018). "Estimating Future Travel Costs for Autonomous Vehicles (AVs) and Shared Autonomous Vehicles (SAVs)". In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, pp. 1702–1707. ISBN: 978-1-7281-0321-1. DOI: 10.1109/ITSC.2018.8569715.

Liu, Jun; Kara M. Kockelman; Patrick M. Boesch; Francesco Ciari (2017). "Tracking a system of shared autonomous vehicles across the Austin, Texas network using agent-based simulation". In: *Transportation* 44.6, pp. 1261–1278. ISSN: 1572-9435. DOI: 10.1007/s11116-017-9811-1. URL: https://link.springer.com/article/10.1007/s11116-017-9811-1.

Liu, Yang; Samitha Samaranayake (2022). "Proactive Rebalancing and Speed-Up Techniques for On-Demand High Capacity Ridesourcing Services". In: *IEEE Transactions on Intelligent Transportation Systems* 23.2, pp. 819–826. ISSN: 1524-9050. DOI: 10.1109/TITS.2020.3016128.

Loeb, Benjamin; Kara M. Kockelman (2019). "Fleet performance and cost evaluation of a shared autonomous electric vehicle (SAEV) fleet: A case study for Austin, Texas". In: *Transportation Research Part A: Policy and Practice* 121, pp. 374–385. ISSN: 0965-8564. DOI: 10.1016/j.tra.2019.01.025. URL: https://www.sciencedirect.com/science/article/pii/S096585641730112X.

Lowalekar, Meghna; Pradeep Varakantham; Patrick Jaillet (2018). "Online spatio-temporal matching in stochastic and dynamic domains". In: *Artificial Intelligence* 261, pp. 71–112. ISSN: 0004-3702. DOI: 10.1016/j.artint.2018.04.005. URL: https://www.sciencedirect.com/science/article/pii/S0004370218302030.

Lu, Chengqi; Michal Maciejewski; Hao Wu; Kai Nagel (2023). "Optimization of demand-responsive transport: The rolling horizon approach". In: *Procedia Computer Science* 220, pp. 145–153. ISSN: 1877-0509. DOI: 10.1016/j.procs.2023.03.021. URL: https://www.sciencedirect.com/science/article/pii/S1877050923005562.

Lukasiewicz, Agnieszka; Venere Stefania Sanna; Vera Lúcia Alves Pereira Diogo; Anikó Bernát (2022). "Shared Mobility: A Reflection on Sharing Economy Initiatives in European Transportation Sectors". In: *The Sharing Economy in Europe*. Ed. by Česnuitytė. Cham: Springer International Publishing, pp. 89–114. ISBN: 978-3-030-86896-3. DOI: 10.1007/978-3-030-86897-0{\textunderscore}5.

Ma, Shuo; Yu Zheng; Ouri Wolfson (2013). "T-share: A large-scale dynamic taxi ridesharing service". In: *2013 IEEE 29th International Conference on Data Engineering (ICDE)*, pp. 410–421. ISBN: 1063-6382. DOI: 10.1109/ICDE.2013.6544843.

Ma, Zhenliang; Haris N. Koutsopoulos (2022). "Near-on-demand mobility. The benefits of user flexibility for ride-pooling services". In: *Transportation Research Part C: Emerging Technologies* 135, p. 103530. ISSN: 0968-090X. DOI: 10.1016/j.trc.2021.103530. URL: https://www.sciencedirect.com/science/article/pii/S0968090X2100512X.

Machado, Cláudia; Nicolas de Salles Hue; Fernando Berssaneti; José Quintanilha (2018). "An Overview of Shared Mobility". In: *Sustainability* 10.12, p. 4342. doi: 10.3390/su10124342.

Magramo, Kathleen; Hassan Tayir; Joyce Jiang (2024). "Super cheap robotaxi rides spark widespread anxiety in China". In: *CNN*. url: https://edition.cnn.com/2024/07/18/cars/china-baidu-apollo-go-robotaxi-anxiety-intl-hnk/index.html.

Mallig, Nicolai; Martin Kagerbauer; Peter Vortisch (2013). "mobiTopp – A Modular Agent-based Travel Demand Modelling Framework". In: *Procedia Computer Science* 19, pp. 854–859. issn: 1877-0509. doi: 10.1016/j.procs.2013.06.114. url: https://www.sciencedirect.com/science/article/pii/S1877050913007229.

Marczuk, Katarzyna Anna; Harold Soh Soon Hong; Carlos Miguel Lima Azevedo; Muhammad Adnan; Scott Drew Pendleton; Emilio Frazzoli; Der Horng Lee (2015). "Autonomous mobility on demand in SimMobility: Case study of the central business district in Singapore". In: *2015 IEEE 7th International Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM)*. IEEE / Institute of Electrical and Electronics Engineers Incorporated, pp. 167–172. isbn: 978-1-4673-7337-1. doi: 10.1109/ICCIS.2015.7274567.

Markov, Iliya; Rafael Guglielmetti; Marco Laumanns; Anna Fernández-Antolín; Ravin de Souza (2021). "Simulation-based design and analysis of on-demand mobility services". In: *Transportation Research Part A: Policy and Practice* 149, pp. 170–205. issn: 0965-8564. doi: 10.1016/j.tra.2021.01.004. url: https://www.sciencedirect.com/science/article/pii/S0965856421000045.

Masoud, Neda; R. Jayakrishnan (2017). "A decomposition algorithm to solve the multi-hop Peer-to-Peer ride-matching problem". In: *Transportation Research Part B: Methodological* 99, pp. 1–29. issn: 0191-2615. doi: 10.1016/j.trb.2017.01.004. url: https://www.sciencedirect.com/science/article/pii/S019126151730019X.

Massobrio, Renzo; Gabriel Fagúndez; Sergio Nesmachnow (2016). "Multiobjective evolutionary algorithms for the taxi sharing problem". In: *International Journal of Metaheuristics* 5.1, p. 67. issn: 1755-2176. doi: 10.1504/IJMHEUR.2016.079103.

McKinsey (2022). *Snapshot of the European car-sharing market*. url: https://www.mckinsey.com/features/mckinsey-center-for-future-mobility/mckinsey-on-urban-mobility/snapshot-of-the-european-car-sharing-market.

Miguel, Cristina; Gabriela Avram; Andrzej Klimczuk; Bori Simonovits; Bálint Balázs; Vida Česnuitytė (2022). "The Sharing Economy in Europe: From Idea to Reality". In: *The Sharing Economy in Europe*. Ed. by Česnuitytė. Cham: Springer International Publishing, pp. 3–18. isbn: 978-3-030-86896-3. doi: 10.1007/978-3-030-86897-0{\textunderscore}1.

Mo, Baichuan; Zhejing Cao; Hongmou Zhang; Yu Shen; Jinhua Zhao (2021). "Competition between shared autonomous vehicles and public transit: A case study in Singapore". In: *Transportation Research Part C: Emerging Technologies* 127, p. 103058. issn: 0968-090X. doi: 10.1016/j.trc.2021.103058. url: https://www.sciencedirect.com/science/article/pii/S0968090X21000863.

Molenbruch, Yves; Kris Braekers; Caris (2017). "Typology and literature review for dial-a-ride problems". In: *Annals of Operations Research* 259.1-2, pp. 295–325. ISSN: 1572-9338. DOI: 10.1007/s10479-017-2525-0. URL: https://link.springer.com/article/10.1007/s10479-017-2525-0.

Morsche, Wietse te; Lissy La Paix Puello; Karst T. Geurs (2019). "Potential uptake of adaptive transport services: An exploration of service attributes and attitudes". In: *Transport Policy* 84, pp. 1–11. ISSN: 0967-070X. DOI: 10.1016/j.tranpol.2019.09.001. URL: https://www.sciencedirect.com/science/article/pii/S0967070X18304645.

Motamedidehkordi, Nassim; Thomas Benz; Martin Margreiter (2016). "Shockwave Analysis on Motorways and Possibility of Damping by Autonomous Vehicles". In: *Advanced Microsystems for Automotive Applications 2015*, pp. 37–52. ISSN: 2196-5552. DOI: 10.1007/978-3-319-20855-8{\textunderscore}4. URL: https://link.springer.com/chapter/10.1007/978-3-319-20855-8_4.

Muelas, Santiago; Antonio LaTorre; José-María Peña (2013). "A variable neighborhood search algorithm for the optimization of a dial-a-ride problem in a large city". In: *Expert Systems with Applications* 40.14, pp. 5516–5531. ISSN: 0957-4174. DOI: 10.1016/j.eswa.2013.04.015. URL: https://www.sciencedirect.com/science/article/pii/S0957417413002522.

Muelas, Santiago; Antonio LaTorre; José-María Peña (2015). "A distributed VNS algorithm for optimizing dial-a-ride problems in large-scale scenarios". In: *Transportation Research Part C: Emerging Technologies* 54, pp. 110–130. ISSN: 0968-090X. DOI: 10.1016/j.trc.2015.02.024. URL: https://www.sciencedirect.com/science/article/pii/S0968090X15000790.

Namdarpour, Farnoosh; Bingqing Liu; Nico Kuehnel; Felix Zwick; Joseph Y.J. Chow (2024). "On non-myopic internal transfers in large-scale ride-pooling systems". In: *Transportation Research Part C: Emerging Technologies* 162, p. 104597. ISSN: 0968-090X. DOI: 10.1016/j.trc.2024.104597. URL: https://www.sciencedirect.com/science/article/pii/S0968090X24001189.

Narayan, Jishnu; Oded Cats; Niels van Oort; Serge Paul Hoogendoorn (2021). "Fleet size determination for a mixed private and pooled on-demand system with elastic demand". In: *Transportmetrica A: Transport Science* 17.4, pp. 897–920. ISSN: 2324-9935. DOI: 10.1080/23249935.2020.1819910.

Narayanan, Santhanakrishnan; Emmanouil Chaniotakis; Constantinos Antoniou (2020). "Shared autonomous vehicle services: A comprehensive review". In: *Transportation Research Part C: Emerging Technologies* 111, pp. 255–293. ISSN: 0968-090X. DOI: 10.1016/j.trc.2019.12.008. URL: https://www.sciencedirect.com/science/article/pii/S0968090X19303493.

Negro, Pia; Dorian Ridderskamp; Moritz Paul; Fabian Fehn; Heidrun Belzner; Klaus Bogenberger (2021). "Cost Structures of Ride-Hailing Providers in the Context of Vehicle Electrification and Automation". In: *2021 7th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*. IEEE, pp. 1–8. ISBN: 978-1-7281-8995-6. DOI: 10.1109/MT-ITS49943.2021.9529308.

New York Post (2022). "Uber brings back 'Uber Pool' under a new name in NYC and other cities". In: *New York Post*. URL: https://nypost.com/2022/06/21/uber-brings-back-uber-pool-as-uberx-share/.

Ng, Max T.M.; Hani S. Mahmassani; Ömer Verbas; Taner Cokyasar; Roman Engelhardt (2024). "Redesigning large-scale multimodal transit networks with shared autonomous mobility services". In: *Transportation Research Part C: Emerging Technologies*, p. 104575. ISSN: 0968-090X. DOI: 10.1016/j.trc.2024.104575. URL: https://www.sciencedirect.com/science/article/pii/S0968090X24000962.

Niels, Tanja; Nikola Mitrovic; Nemanja Dobrota; Klaus Bogenberger; Aleksandar Stevanovic; Robert Bertini (2020). "Simulation-Based Evaluation of a New Integrated Intersection Control Scheme for Connected Automated Vehicles and Pedestrians". In: *Transportation Research Record: Journal of the Transportation Research Board* 2674.11, pp. 779–793. ISSN: 0361-1981. DOI: 10.1177/0361198120949531.

Oke, Jimi B.; Arun Prakash Akkinepally; Siyu Chen; Yifei Xie; Youssef M. Aboutaleb; Carlos Lima Azevedo; P. Christopher Zegras; Joseph Ferreira; Moshe Ben-Akiva (2020). "Evaluating the systemic effects of automated mobility-on-demand services via large-scale agent-based simulation of auto-dependent prototype cities". In: *Transportation Research Part A: Policy and Practice* 140, pp. 98–126. ISSN: 0965-8564. DOI: 10.1016/j.tra.2020.06.013. URL: https://www.sciencedirect.com/science/article/pii/S0965856420306327.

Osorio, Jesus; Chao Lei; Yanfeng Ouyang (2021). "Optimal rebalancing and on-board charging of shared electric scooters". In: *Transportation Research Part B: Methodological* 147, pp. 197–219. ISSN: 0191-2615. DOI: 10.1016/j.trb.2021.03.009. URL: https://www.sciencedirect.com/science/article/pii/S0191261521000552.

Ouyang, Yanfeng; Haolin Yang; Carlos F. Daganzo (2021). "Performance of reservation-based carpooling services under detour and waiting time restrictions". In: *Transportation Research Part B: Methodological* 150, pp. 370–385. ISSN: 0191-2615. DOI: 10.1016/j.trb.2021.06.007. URL: https://www.sciencedirect.com/science/article/pii/S0191261521001181.

Pandey, Venktesh; Julien Monteil; Claudio Gambella; Andrea Simonetto (2019). "On the needs for MaaS platforms to handle competition in ridesharing mobility". In: *Transportation Research Part C: Emerging Technologies* 108, pp. 269–288. ISSN: 0968-090X. DOI: 10.1016/j.trc.2019.09.021. URL: https://www.sciencedirect.com/science/article/pii/S0968090X19303353.

Parragh, Sophie N.; Karl F. Doerner; Richard F. Hartl (2010). "Variable neighborhood search for the dial-a-ride problem". In: *Computers & Operations Research* 37.6, pp. 1129–1138. ISSN: 0305-0548. DOI: 10.1016/j.cor.2009.10.003. URL: https://www.sciencedirect.com/science/article/pii/S030505480900241X.

Parragh, Sophie N.; Verena Schmid (2013). "Hybrid column generation and large neighborhood search for the dial-a-ride problem". In: *Computers & Operations Research* 40.1, pp. 490–497. ISSN: 0305-0548. DOI: 10.1016/j.cor.2012.08.004. URL: https://www.sciencedirect.com/science/article/pii/S0305054812001694.

Pinto, Helen K.R.F.; Michael F. Hyland; Hani S. Mahmassani; I. Ömer Verbas (2020). "Joint design of multimodal transit networks and shared autonomous mobility

fleets". In: *Transportation Research Part C: Emerging Technologies* 113, pp. 2–20. ISSN: 0968-090X. DOI: 10.1016/j.trc.2019.06.010. URL: https://www.sciencedirect.com/science/article/pii/S0968090X18317728.

Psaraftis, Harilaos N. (1980). "A Dynamic Programming Solution to the Single Vehicle Many-to-Many Immediate Request Dial-a-Ride Problem". In: *Transportation Science* 14.2, pp. 130–154. ISSN: 0041-1655. DOI: 10.1287/trsc.14.2.130.

Psaraftis, Harilaos N.; Min Wen; Christos A. Kontovas (2016). "Dynamic vehicle routing problems: Three decades and counting". In: *Networks* 67.1, pp. 3–31. ISSN: 1097-0037. DOI: 10.1002/net.21628. URL: https://onlinelibrary.wiley.com/doi/10.1002/net.21628.

Reiss, Svenja; Klaus Bogenberger (2017). "A Relocation Strategy for Munich's Bike Sharing System: Combining an operator-based and a user-based Scheme". In: *Transportation Research Procedia* 22, pp. 105–114. ISSN: 2352-1465. DOI: 10.1016/j.trpro.2017.03.016. URL: https://www.sciencedirect.com/science/article/pii/S2352146517301515.

Riley, Connor; Antoine Legrain; Pascal van Hentenryck (2019). "Column Generation for Real-Time Ride-Sharing Operations". In: *Integration of Constraint Programming, Artificial Intelligence, and Operations Research*. Ed. by Rousseau; Hofmann. Vol. 11494. Lecture Notes in Computer Science. Springer International Publishing, pp. 472–487. ISBN: 978-3-030-19211-2. DOI: 10.1007/978-3-030-19212-9$\backslash${\textunderscore}31.

Ruch, Claudio; Sebastian Horl; Emilio Frazzoli (2018). "AMoDeus, a Simulation-Based Testbed for Autonomous Mobility-on-Demand Systems". In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, pp. 3639–3644. ISBN: 978-1-7281-0321-1. DOI: 10.1109/ITSC.2018.8569961.

Ruijter, Arjan de; Oded Cats; Javier Alonso-Mora; Serge Hoogendoorn (2023). "Ride-pooling adoption, efficiency and level of service under alternative demand, behavioural and pricing settings". In: *Transportation Planning and Technology* 46.4, pp. 407–436. ISSN: 0308-1060. DOI: 10.1080/03081060.2023.2194874.

SAE (2021). *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*. URL: https://www.sae.org/standards/content/j3016_201806/?src=j3016_201609.

Salazar, Mauro; Nicolas Lanzetti; Federico Rossi; Maximilian Schiffer; Marco Pavone (2020). "Intermodal Autonomous Mobility-on-Demand". In: *IEEE Transactions on Intelligent Transportation Systems* 21.9, pp. 3946–3960. ISSN: 1524-9050. DOI: 10.1109/TITS.2019.2950720.

Salazar, Mauro; Matthew Tsao; Izabel Aguiar; Maximilian Schiffer; Marco Pavone (2019). "A Congestion-aware Routing Scheme for Autonomous Mobility-on-Demand Systems". In: *2019 18th European Control Conference (ECC)*. IEEE, pp. 3040–3046. ISBN: 978-3-907144-00-8. DOI: 10.23919/ECC.2019.8795897.

Santi, Paolo; Giovanni Resta; Michael Szell; Stanislav Sobolevsky; Steven H. Strogatz; Carlo Ratti (2014). "Quantifying the benefits of vehicle pooling with shareability networks". In: *Proceedings of the National Academy of Sciences of the United*

States of America 111.37, pp. 13290–13294. DOI: 10.1073/pnas.1403657111. URL: https://www.pnas.org/doi/10.1073/pnas.1403657111.

SANTOS, DOUGLAS O.; EDUARDO C. XAVIER (2013). "Dynamic Taxi and Ridesharing: A Framework and Heuristics for the Optimization Problem". In: *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*. IJCAI '13. AAAI Press, pp. 2885–2891. ISBN: 9781577356332.

SANTOS, DOUGLAS O.; EDUARDO C. XAVIER (2015). "Taxi and Ride Sharing: A Dynamic Dial-a-Ride Problem with Money as an Incentive". In: *Expert Systems with Applications* 42.19, pp. 6728–6737. ISSN: 0957-4174. DOI: 10.1016/j.eswa.2015.04.060. URL: https://www.sciencedirect.com/science/article/pii/S0957417415003024.

SARMA, NAVJYOTH J.S.; MICHAEL HYLAND (2024). "Metrics for Quantifying Shareability in Transportation Networks: The Maximum Network Flow Overlap Problem". In: *Transportation Research Part C: Emerging Technologies* 158, p. 104420. ISSN: 0968-090X. DOI: 10.1016/j.trc.2023.104420. URL: https://www.sciencedirect.com/science/article/pii/S0968090X23004102.

SAYARSHAD, HAMID R.; JOSEPH Y.J. CHOW (2017). "Non-myopic relocation of idle mobility-on-demand vehicles as a dynamic location-allocation-queueing problem". In: *Transportation Research Part E: Logistics and Transportation Review* 106, pp. 60–77. ISSN: 1366-5545. DOI: 10.1016/j.tre.2017.08.003. URL: https://www.sciencedirect.com/science/article/pii/S1366554517300121.

SCHLENTHER, TILMANN; GREGOR LEICH; MICHAL MACIEJEWSKI; KAI NAGEL (2023). "Addressing spatial service provision equity for pooled ride–hailing services through rebalancing". In: *IET Intelligent Transport Systems* 17.3, pp. 547–556. ISSN: 1751-956X. DOI: 10.1049/itr2.12279.

SCHRANK, DAVID; BILL EISELE; TIM LOMAX; TEXAS TRANSPORTATION INSTITUTE (2019). *Urban Mobility Report 2019*. Texas Transportation Institute and INRIX, Inc. URL: https://rosap.ntl.bts.gov/view/dot/61408.

SCHULLER, PIETER; ANDRES FIELBAUM; JAVIER ALONSO-MORA (2021). "Towards a geographically even level of service in on-demand ridepooling". In: *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*. [Piscataway, NJ]: IEEE, pp. 2429–2434. ISBN: 978-1-7281-9142-3. DOI: 10.1109/ITSC48978.2021.9564910.

SÉJOURNÈ, THIBAULT; SAMITHA SAMARANAYAKE; SIDDHARTHA BANERJEE (2018). "The Price of Fragmentation in Mobility-on-Demand Services". In: *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 2.2, pp. 1–26. DOI: 10.1145/3224425.

SHAHEEN, SUSAN (2018). "Shared Mobility: The Potential of Ridehailing and Pooling". In: *Three Revolutions*. Ed. by DANIEL SPERLING. Washington, DC: Island Press/Center for Resource Economics, pp. 55–76. ISBN: 978-1-61091-983-8. DOI: 10.5822/978-1-61091-906-7{\textunderscore}3.

SHAHEEN, SUSAN; ADAM COHEN (2019). "Shared ride services in North America: definitions, impacts, and the future of pooling". In: *Transport Reviews* 39.4, pp. 427–442. ISSN: 0144-1647. DOI: 10.1080/01441647.2018.1497728.

SHETTY, AKHIL; MENGQIAO YU; ALEX KURZHANSKIY; OFFER GREMBEK; HAMIDREZA TAVAFOGHI; PRAVIN VARAIYA (2021). "Safety challenges for autonomous vehicles in the

absence of connectivity". In: *Transportation Research Part C: Emerging Technologies* 128, p. 103133. ISSN: 0968-090X. DOI: 10.1016/j.trc.2021.103133.

Shoup, D. C.; American Planning Association (2005). *The High Cost of Free Parking*. A Planners Press book. Planners Press, American Planning Association. ISBN: 9781884829987. URL: https://books.google.de/books?id=WBe3AAAAIAAJ.

Sieber, L.; C. Ruch; S. Hörl; K. W. Axhausen; E. Frazzoli (2020). "Improved public transportation in rural areas with self-driving cars: A study on the operation of Swiss train lines". In: *Transportation Research Part A: Policy and Practice* 134, pp. 35–51. ISSN: 0965-8564. DOI: 10.1016/j.tra.2020.01.020. URL: https://www.sciencedirect.com/science/article/pii/S0965856418314083.

Simonetto, Andrea; Julien Monteil; Claudio Gambella (2019). "Real-time city-scale ridesharing via linear assignment problems". In: *Transportation Research Part C: Emerging Technologies* 101, pp. 208–232. ISSN: 0968-090X. DOI: 10.1016/j.trc.2019.01.019. URL: https://www.sciencedirect.com/science/article/pii/S0968090X18302882.

Soza-Parra, Jaime; Rafal Kucharski; Oded Cats (2022). "The shareability potential of ride-pooling under alternative spatial demand patterns". In: *Transportmetrica A: Transport Science*, pp. 1–23. ISSN: 2324-9935. DOI: 10.1080/23249935.2022.2140022.

Sperling, Daniel, ed. (2018). *Three Revolutions*. Washington, DC: Island Press/Center for Resource Economics. ISBN: 978-1-61091-983-8. DOI: 10.5822/978-1-61091-906-7.

Statista (2024). *Ride-hailing - North America | Statista Market Forecast*. URL: https://www.statista.com/outlook/mmo/shared-mobility/ride-hailing/north-america#global-comparison.

Stiglic, Mitja; Niels Agatz; Martin Savelsbergh; Mirko Gradisar (2016). "Making dynamic ride-sharing work: The impact of driver and rider flexibility". In: *Transportation Research Part E: Logistics and Transportation Review* 91, pp. 190–207. ISSN: 1366-5545. DOI: 10.1016/j.tre.2016.04.010. URL: https://www.sciencedirect.com/science/article/pii/S1366554515303033.

Stueger, Philipp N.; Fabian Fehn; Klaus Bogenberger (2023). "Minimizing the Effects of Urban Mobility-on-Demand Pick-Up and Drop-Off Stops: A Microscopic Simulation Approach". In: *Transportation Research Record: Journal of the Transportation Research Board* 2677.1, pp. 814–828. ISSN: 0361-1981. DOI: 10.1177/03611981221101894.

Su, Shun; Emmanouil Chaniotakis; Santhanakrishnan Narayanan; Hai Jiang; Constantinos Antoniou (2022). "Clustered tabu search optimization for reservation-based shared autonomous vehicles". In: *Transportation Letters* 14.2, pp. 124–128. ISSN: 1942-7867. DOI: 10.1080/19427867.2020.1824309.

Syed, Arslan Ali; Karim Akhnoukh; Bernd Kaltenhaeuser; Klaus Bogenberger (2019). "Neural Network Based Large Neighborhood Search Algorithm for Ride Hailing Services". In: *Progress in artificial intelligence*. Cham, Switzerland: Springer, pp. 584–595. ISBN: 978-3-030-30241-2. DOI: 10.1007/978-3-030-30241-2{\textunderscore}49. URL: https://link.springer.com/chapter/10.1007/978-3-030-30241-2_49.

Syed, Arslan Ali; Florian Dandl; Bernd Kaltenhäuser; Klaus Bogenberger (2021). "Density Based Distribution Model for Repositioning Strategies of Ride Hailing Services". In: *Frontiers in Future Transportation* 2. DOI: 10.3389/ffutr.2021.681451.

Syed, Arslan Ali; Yunfei Zhang; Klaus Bogenberger (2023). *Data-driven Spatio-Temporal Scaling of Travel Times for AMoD Simulations*. URL: http://arxiv.org/pdf/2311.06291v1.

Tachet, R.; O. Sagarra; P. Santi; G. Resta; M. Szell; S. H. Strogatz; C. Ratti (2017). "Scaling Law of Urban Ride Sharing". In: *Scientific Reports* 7.1, p. 42868. ISSN: 2045-2322. DOI: 10.1038/srep42868. URL: https://www.nature.com/articles/srep42868.

Tafreshian, Amirmahdi; Mojtaba Abdolmaleki; Neda Masoud; Huizhu Wang (2021). "Proactive shuttle dispatching in large-scale dynamic dial-a-ride systems". In: *Transportation Research Part B: Methodological* 150, pp. 227–259. ISSN: 0191-2615. DOI: 10.1016/j.trb.2021.06.002. URL: https://www.sciencedirect.com/science/article/pii/S0191261521001090.

Talebpour, Alireza; Hani S. Mahmassani (2016). "Influence of connected and autonomous vehicles on traffic flow stability and throughput". In: *Transportation Research Part C: Emerging Technologies* 71, pp. 143–163. ISSN: 0968-090X. DOI: 10.1016/j.trc.2016.07.007. URL: https://www.sciencedirect.com/science/article/pii/S0968090X16301140.

Tavor, Shir; Tal Raviv (2023). "Anticipatory rebalancing of RoboTaxi systems". In: *Transportation Research Part C: Emerging Technologies* 153, p. 104196. ISSN: 0968-090X. DOI: 10.1016/j.trc.2023.104196. URL: https://www.sciencedirect.com/science/article/pii/S0968090X23001857.

Taxi-München eG (2024). *Preise - Taxi-München eG*. URL: https://www.taxi-muenchen.de/preise/.

The White House (2021). *Executive Order on Catalyzing Clean Energy Industries and Jobs Through Federal Sustainability*. URL: https://www.whitehouse.gov/briefing-room/presidential-actions/2021/12/08/executive-order-on-catalyzing-clean-energy-industries-and-jobs-through-federal-sustainability/.

Tirachini, Alejandro; Constantinos Antoniou (2020). "The economics of automated public transport: Effects on operator cost, travel time, fare and subsidy". In: *Economics of Transportation* 21, p. 100151. ISSN: 2212-0122. DOI: 10.1016/j.ecotra.2019.100151. URL: https://www.sciencedirect.com/science/article/pii/S2212012219300802.

Tobias Enders; James Harrison; Marco Pavone; Maximilian Schiffer (2023). "Hybrid Multi-agent Deep Reinforcement Learning for Autonomous Mobility on Demand Systems". In: *Learning for Dynamics and Control Conference*, pp. 1284–1296. ISSN: 2640-3498. URL: https://proceedings.mlr.press/v211/enders23a.html.

TomTom (2022). *How the Pandemic Changed How We Move in Our Cities in 2021*. URL: https://www.tomtom.com/newsroom/explainers-and-insights/how-covid-19-changed-the-way-we-move-in-2021/.

Treat, John R.; N. S. Tumbas; Stephen T. McDonald; David Shinar; R. D. Hume; R. E. Mayer; R. L. Stansifer; N. J. Castellan (1979). *Tri-level study of the causes of traffic accidents: final report. Executive summary*. URL: https://deepblue.lib.umich.edu/handle/2027.42/64993.

Tsao, Matthew; Dejan Milojevic; Claudio Ruch; Mauro Salazar; Emilio Frazzoli; Marco Pavone (2019). "Model Predictive Control of Ride-sharing Autonomous Mobility-on-Demand Systems". In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 6665–6671. ISBN: 978-1-5386-6027-0. DOI: 10.1109/ICRA.2019.8794194.

TUM-VT (2022). *FleetPy*. URL: https://github.com/TUM-VT/FleetPy.

Tuncel, Kerem; Haris N. Koutsopoulos; Zhenliang Ma (2023). "An integrated ride-matching and vehicle-rebalancing model for shared mobility on-demand services". In: *Computers & Operations Research* 159, p. 106317. ISSN: 0305-0548. DOI: 10.1016/j.cor.2023.106317. URL: https://www.sciencedirect.com/science/article/pii/S0305054823001818.

TÜV Süd (2023). *Robotaxis auf der Überholspur: Wie fahrerlose Fahrdienste eine neue Mobilität ermöglichen*. URL: https://abouttrust.tuvsud.com/robotaxis-auf-der-ueberholspur-wie-fahrerlose-fahrdienste-eine-neue-mobilitaet-ermoeglichen.

Umweltbundesamt (2022). *Indicator: Greenhouse gas emissions*. URL: https://www.umweltbundesamt.de/en/data/environmental-indicators/indicator-greenhouse-gas-emissions#at-a-glance.

Umweltbundesamt (2024). *Fahrgemeinschaften*. URL: https://www.umweltbundesamt.de/umwelttipps-fuer-den-alltag/mobilitaet/fahrgemeinschaften#hintergrund.

United Nations (2018). *68% of the world population projected to live in urban areas by 2050, says UN*. URL: https://www.un.org/development/desa/en/news/population/2018-revision-of-world-urbanization-prospects.html.

US EPA (2021). *Sources of Greenhouse Gas Emissions | US EPA*. URL: https://www.epa.gov/ghgemissions/sources-greenhouse-gas-emissions.

Valadkhani, Amir Hosein; Mohsen Ramezani (2023). "Dynamic ride-sourcing systems for city-scale networks, Part II: Proactive vehicle repositioning". In: *Transportation Research Part C: Emerging Technologies* 152, p. 104159. ISSN: 0968-090X. DOI: 10.1016/j.trc.2023.104159. URL: https://www.sciencedirect.com/science/article/pii/S0968090X23001481.

van Engelen, Matti; Oded Cats; Henk Post; Karen Aardal (2018). "Enhancing flexible transport services with demand-anticipatory insertion heuristics". In: *Transportation Research Part E: Logistics and Transportation Review* 110, pp. 110–121. ISSN: 1366-5545. DOI: 10.1016/j.tre.2017.12.015.

Vazifeh, M. M.; P. Santi; G. Resta; S. H. Strogatz; C. Ratti (2018). "Addressing the minimum fleet problem in on-demand urban mobility". In: *Nature* 557.7706, pp. 534–538. ISSN: 1476-4687. DOI: 10.1038/s41586-018-0095-1. URL: https://www.nature.com/articles/s41586-018-0095-1.

vercel (2024). *C++ VS Python benchmarks, Which programming language or compiler is faster*. URL: https://programming-language-benchmarks.vercel.app/cpp-vs-python.

Vosooghi, Reza; Jakob Puchinger; Joschka Bischoff; Marija Jankovic; Anthony Vouillon (2020). "Shared autonomous electric vehicle service performance: As-

sessing the impact of charging infrastructure". In: *Transportation Research Part D: Transport and Environment* 81, p. 102283. ISSN: 1361-9209. DOI: 10.1016/j.trd.2020.102283. URL: https://www.sciencedirect.com/science/article/pii/S1361920919307114.

Vosooghi, Reza; Jakob Puchinger; Marija Jankovic; Anthony Vouillon (2019). "Shared autonomous vehicle simulation and service design". In: *Transportation Research Part C: Emerging Technologies* 107, pp. 15–33. ISSN: 0968-090X. DOI: 10.1016/j.trc.2019.08.006. URL: https://www.sciencedirect.com/science/article/pii/S0968090X19304449.

Wallar, Alex; Javier Alonso-Mora; Daniela Rus (2021). "Optimizing Vehicle Distributions and Fleet Sizes for Shared Mobility-on-Demand". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3853–3859. ISBN: 2577-087X. DOI: 10.1109/ICRA.2019.8793685.

Wallar, Alex; Menno van der Zee; Javier Alonso-Mora; Daniela Rus (2018). "Vehicle Rebalancing for Mobility-on-Demand Systems with Ride-Sharing". In: *IROS Madrid 2018*. [Piscataway, New Jersey]: IEEE, pp. 4539–4546. ISBN: 978-1-5386-8094-0. DOI: 10.1109/IROS.2018.8593743.

WallStreetZen (2023). *Uber Statistics - Uber Facts, Stats, Trends & Data (2023) | WallStreetZen*.

Wang, Hai; Hai Yang (2019). "Ridesourcing systems: A framework and review". In: *Transportation Research Part B: Methodological* 129, pp. 122–155. ISSN: 0191-2615. DOI: 10.1016/j.trb.2019.07.009. URL: https://www.sciencedirect.com/science/article/pii/S019126151831172X.

Wang, Jun; Li Zhang; Yanjun Huang; Jian Zhao; Francesco Bella (2020). "Safety of Autonomous Vehicles". In: *Journal of Advanced Transportation* 2020, pp. 1–13. ISSN: 0197-6729. DOI: 10.1155/2020/8867757.

Wang, Ning; Yelin Lyu; Shengling Jia; Chaojun Zheng; Zhiquan Meng; Jingyun Chen (2023). "A dynamic graph-based many-to-one ride-matching approach for shared autonomous electric vehicles". In: *Transportation*, pp. 1–27. ISSN: 1572-9435. DOI: 10.1007/s11116-023-10391-3. URL: https://link.springer.com/article/10.1007/s11116-023-10391-3.

Waymo (2024). *Waymo - Self-Driving Cars - Autonomous Vehicles - Ride-Hail*. URL: https://waymo.com/.

Weikl, Simone; Klaus Bogenberger (2013). "Relocation Strategies and Algorithms for Free-Floating Car Sharing Systems". In: *IEEE Intelligent Transportation Systems Magazine* 5.4, pp. 100–111. ISSN: 1939-1390. DOI: 10.1109/MITS.2013.2267810.

Wen, Jian; Neema Nassir; Jinhua Zhao (2019). "Value of demand information in autonomous mobility-on-demand systems". In: *Transportation Research Part A: Policy and Practice* 121, pp. 346–359. ISSN: 0965-8564. DOI: 10.1016/j.tra.2019.01.018. URL: https://www.sciencedirect.com/science/article/pii/S0965856418306785.

Wen, Jian; Jinhua Zhao; Patrick Jaillet (2017). "Rebalancing shared mobility-on-demand systems: A reinforcement learning approach". In: *IEEE ITSC 2017*. Piscataway, NJ: IEEE, pp. 220–225. ISBN: 978-1-5386-1526-3. DOI: 10.1109/ITSC.2017.8317908.

Wenzel, Tom; Clement Rames; Eleftheria Kontou; Alejandro Henao (2019). "Travel and energy implications of ridesourcing service in Austin, Texas". In: *Transportation*

Research Part D: Transport and Environment 70, pp. 18–34. ISSN: 1361-9209. DOI: 10.1016/j.trd.2019.03.005. URL: https://www.sciencedirect.com/science/article/pii/S1361920918309878.

Wilkes, Gabriel; Roman Engelhardt; Lars Briem; Florian Dandl; Peter Vortisch; Klaus Bogenberger; Martin Kagerbauer (2021). "Self-Regulating Demand and Supply Equilibrium in Joint Simulation of Travel Demand and a Ride-Pooling Service". In: *Transportation Research Record: Journal of the Transportation Research Board* 2675.8, pp. 226–239. ISSN: 0361-1981. DOI: 10.1177/0361198121997140.

Winkle, Thomas (2016). "Safety Benefits of Automated Vehicles: Extended Findings from Accident Research for Development, Validation and Testing". In: *Etiology and Morphogenesis of Congenital Heart Disease: From Gene Function and Cellular Interaction to Morphology*. Ed. by Toshio Nakanishi; Roger R. Markwald; H.Scott Baldwin; Bradley B. Keller; Deepack Srivastava; Hiroyuki Yamagishi. Berlin, Heidelberg: Springer, pp. 335–364. ISBN: 978-3-662-48845-4. DOI: 10.1007/978-3-662-48847-8{\textunderscore}17.

Winter, Konstanze; Oded Cats; Karel Martens; Bart van Arem (2021). "Relocating shared automated vehicles under parking constraints: assessing the impact of different strategies for on-street parking". In: *Transportation* 48.4, pp. 1931–1965. ISSN: 1572-9435. DOI: 10.1007/s11116-020-10116-w. URL: https://link.springer.com/article/10.1007/s11116-020-10116-w.

Wolf, Fynn; Roman Engelhardt; Yunfei Zhang; Florian Dandl; Klaus Bogenberger (2023). "Effects of Dynamic and Stochastic Travel Times on the Operation of Mobility-on-Demand Services". In: *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, pp. 5476–5481. ISBN: 979-8-3503-9946-2. DOI: 10.1109/ITSC57777.2023.10422554.

Yang, Lu; Muhammad Adnan; Kakali Basak; Francisco Pereira; Carlos C.; Saber V.H.; Lognathan H.; Moshe Ben-Akiva (2015). "SimMobility Mid-Term Simulator: A State of the Art Integrated Agent Based Demand and Supply Model". In.

Yang, Yizhe; Hongjun Cui; Xinwei Ma; Wei Fan; Minqing Zhu; Sheng Yao (2022). "Evaluating the Impacts of Optimization Horizon on the Shared Autonomous Vehicle Reservation Request System". In: *Journal of Advanced Transportation* 2022, pp. 1–19. ISSN: 0197-6729. DOI: 10.1155/2022/7304148. URL: https://www.hindawi.com/journals/jat/2022/7304148/.

Ye, Lanhang; Toshiyuki Yamamoto (2019). "Evaluating the impact of connected and autonomous vehicles on traffic safety". In: *Physica A: Statistical Mechanics and its Applications* 526, p. 121009. ISSN: 0378-4371. DOI: 10.1016/j.physa.2019.04.245. URL: https://www.sciencedirect.com/science/article/pii/S0378437119306181.

Yeo, Jiho; Sujin Lee; Kitae Jang; Jinwoo Lee (2023). "Real-Time Operations of Autonomous Mobility-on-Demand Services with Inter- and Intra-Zonal Relocation". In: *IEEE Transactions on Intelligent Vehicles*, pp. 1–14. ISSN: 2379-8858. DOI: 10.1109/TIV.2023.3299692.

Young, Mischa; Steven Farber; Matthew Palm (2020). "The true cost of sharing: A detour penalty analysis between UberPool and UberX trips in Toronto". In: *Transportation Research Part D: Transport and Environment* 87, p. 102540. ISSN: 1361-9209. DOI: 10.

1016/j.trd.2020.102540. URL: https://www.sciencedirect.com/science/article/pii/S1361920920307276.

Yu, Xinlian; Zihao Zhu; Haijun Mao; Mingzhuang Hua; Dawei Li; Jingxu Chen; Hongli Xu (2023). "Coordinating matching, rebalancing and charging of electric ride-hailing fleet under hybrid requests". In: *Transportation Research Part D: Transport and Environment* 123, p. 103903. ISSN: 1361-9209. DOI: 10.1016/j.trd.2023.103903. URL: https://www.sciencedirect.com/science/article/pii/S1361920923003000.

Zalesak, Matthew; Samitha Samaranayake (2021). "Real time operation of high-capacity electric vehicle ridesharing fleets". In: *Transportation Research Part C: Emerging Technologies* 133, p. 103413. ISSN: 0968-090X. DOI: 10.1016/j.trc.2021.103413. URL: https://www.sciencedirect.com/science/article/pii/S0968090X21004071.

Zhan, Xianyuan; Xinwu Qian; Satish V. Ukkusuri (2016). "A Graph-Based Approach to Measuring the Efficiency of an Urban Taxi Service System". In: *IEEE Transactions on Intelligent Transportation Systems* 17.9, pp. 2479–2489. ISSN: 1524-9050. DOI: 10.1109/TITS.2016.2521862.

Zhan, Xingbin; W. Y. Szeto; C. S. Shui; Xiqun Chen (2021). "A modified artificial bee colony algorithm for the dynamic ride-hailing sharing problem". In: *Transportation Research Part E: Logistics and Transportation Review* 150, p. 102124. ISSN: 1366-5545. DOI: 10.1016/j.tre.2020.102124. URL: https://www.sciencedirect.com/science/article/pii/S1366554520307729.

Zhang, Kenan; Yu Nie (2021). "To pool or not to pool: Equilibrium, pricing and regulation". In: *Transportation Research Part B: Methodological* 151, pp. 59–90. ISSN: 0191-2615. DOI: 10.1016/j.trb.2021.07.001. URL: https://www.sciencedirect.com/science/article/pii/S0191261521001338.

Zhang, Rick; Marco Pavone (2016). "Control of robotic mobility-on-demand systems: A queueing-theoretical perspective". In: *The International Journal of Robotics Research* 35.1-3, pp. 186–203. ISSN: 0278-3649. DOI: 10.1177/0278364915581863.

Zhang, Wenwen; Subhrajit Guhathakurta; Jinqi Fang; Ge Zhang (2015). "Exploring the impact of shared autonomous vehicles on urban parking demand: An agent-based simulation approach". In: *Sustainable Cities and Society* 19, pp. 34–45. ISSN: 2210-6707. DOI: 10.1016/j.scs.2015.07.006. URL: https://www.sciencedirect.com/science/article/pii/S221067071530010X.

Zhang, Yunfei; Roman Engelhardt; Arslan-Ali Syed; Florian Dandl; Cornelius Hardt; Klaus Bogenberger (2022). "Simulating Charging Processes of Mobility-On-Demand Services at Public Infrastructure: Can Operators Complement Each Other?" In: *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, pp. 2200–2205. ISBN: 978-1-6654-6880-0. DOI: 10.1109/ITSC55140.2022.9922449.

Zhu, Pengbo; Isik Ilber Sirmatel; Giancarlo Ferrari Trecate; Nikolas Geroliminis (2022). "Idle-vehicle Rebalancing Coverage Control for Ride-sourcing systems". In: *2022 European Control Conference (ECC)*. IEEE, pp. 1970–1975. ISBN: 978-3-9071-4407-7. DOI: 10.23919/ECC55457.2022.9838069.

Zuo, Haojia; Bo Cao; Ying Zhao; Bilong Shen; Weimin Zheng; Yan Huang (2021). "High-capacity ride-sharing via shortest path clustering on large road networks". In: *The Journal of Supercomputing* 77.4, pp. 4081–4106. ISSN: 1573-0484. DOI: 10.1007/

s11227-020-03424-6. URL: https://link.springer.com/article/10.1007/s11227-020-03424-6.

Zwick, Felix; Nico Kuehnel; Rolf Moeckel; Kay W. Axhausen (2021a). "Agent-based simulation of city-wide autonomous ride-pooling and the impact on traffic noise". In: *Transportation Research Part D: Transport and Environment* 90, p. 102673. ISSN: 1361-9209. DOI: 10.1016/j.trd.2020.102673. URL: https://www.sciencedirect.com/science/article/pii/S1361920920308580.

Zwick, Felix; Nico Kuehnel; Rolf Moeckel; Kay W. Axhausen (2021b). "Ride-Pooling Efficiency in Large, Medium-Sized and Small Towns -Simulation Assessment in the Munich Metropolitan Region". In: *Procedia Computer Science* 184, pp. 662–667. ISSN: 1877-0509. DOI: 10.1016/j.procs.2021.03.083. URL: https://www.sciencedirect.com/science/article/pii/S1877050921007195.

Zwick, Felix; Gabriel Wilkes; Roman Engelhardt; Steffen Axer; Florian Dandl; Hannes Rewald; Nadine Kostorz; Eva Fraedrich; Martin Kagerbauer; Kay W. Axhausen (2022). "Mode choice and ride-pooling simulation: A comparison of mobiTopp, Fleetpy, and MATSim". In: *Procedia Computer Science* 201, pp. 608–613. ISSN: 1877-0509. DOI: 10.1016/j.procs.2022.03.079. URL: https://www.sciencedirect.com/science/article/pii/S1877050922004926.

Statement on the Use of Generative AI

The author acknowledges that he used generative AI (ChatGPT 3.5 and 4, Grammarly, and GitHub Copilot) exclusively for spell-checking and revising of his own text.

Appendix

I Simulation Framework

Creation of Zone System

For creating zones and corresponding centroids, a maximum coverage problem based on Wallar et al. [2018] is solved. Let K_n be the set of access nodes reachable from node n within a maximum driving time of t^Z_{max}. The minimum set of zone centroid nodes that guarantee that each access node is reachable by at least one centroid node within a maximum driving time of t^Z_{max} is determined by solving the following ILP:

$$\text{Minimize:} \quad \sum_{n \in N} x_n \tag{33a}$$

$$\text{s.t.:} \quad \sum_{\hat{n} \in K_n} x_n \geq 1 \qquad \forall n \in N \tag{33b}$$

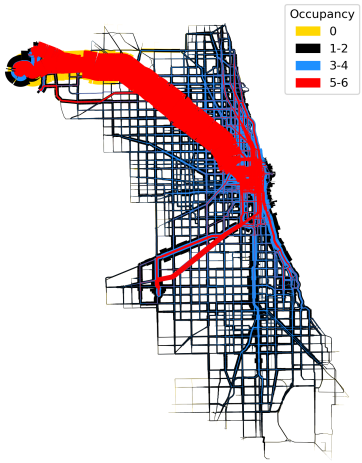
$$x_n \in \{0, 1\} \qquad \forall n \in N \tag{33c}$$

Constraints 33b ensure that each access node is reachable by at least one centroid node.

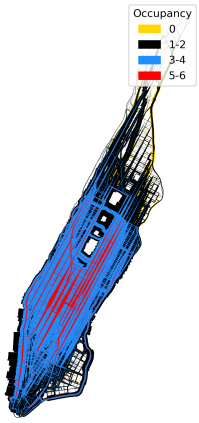
II Further Results - Assignment

Impacts of Ride-Pooling

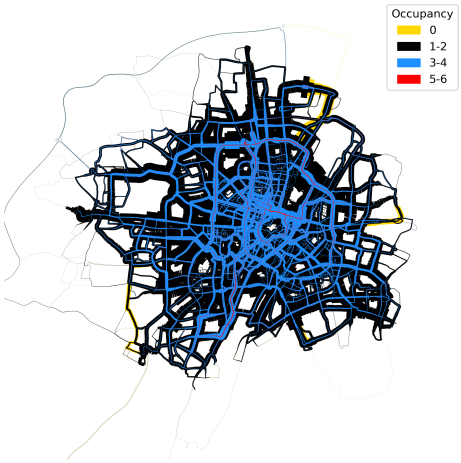
Figure II.1 shows the vehicle occupancy counts on different network sections for different scenarios. High occupancy trips (5-6 passengers) are especially notable in the Chicago case study on highways between the city center and the airports. While high occupancy levels are also observed in central Manhattan, occupancy counts of 5-6 are hardly observed in Munich.



(a) Chicago. 100% Demand Penetration. 1700 vehicles with capacity 6.



(b) Manhattan. 100% Demand Penetration. 1600 vehicles with capacity 6.



(c) Munich. 10% Demand Penetration. 1200 vehicles with capacity 6.

Figure II.1: Vehicle occupancy counts on different network sections. Thickness indicates the number of vehicles passing through the section with a given occupancy. Higher occupancy are plotted on top of lower occupancy levels.

Assignment Reliability

In extension to Figure 5.8, Figure II.2 shows the impact of different assignment reliability strategies for the Munich case study. Overall, similar behavior is observable in the Munich case study compared to the discussion of the Chicago case study in section 5.2.3.

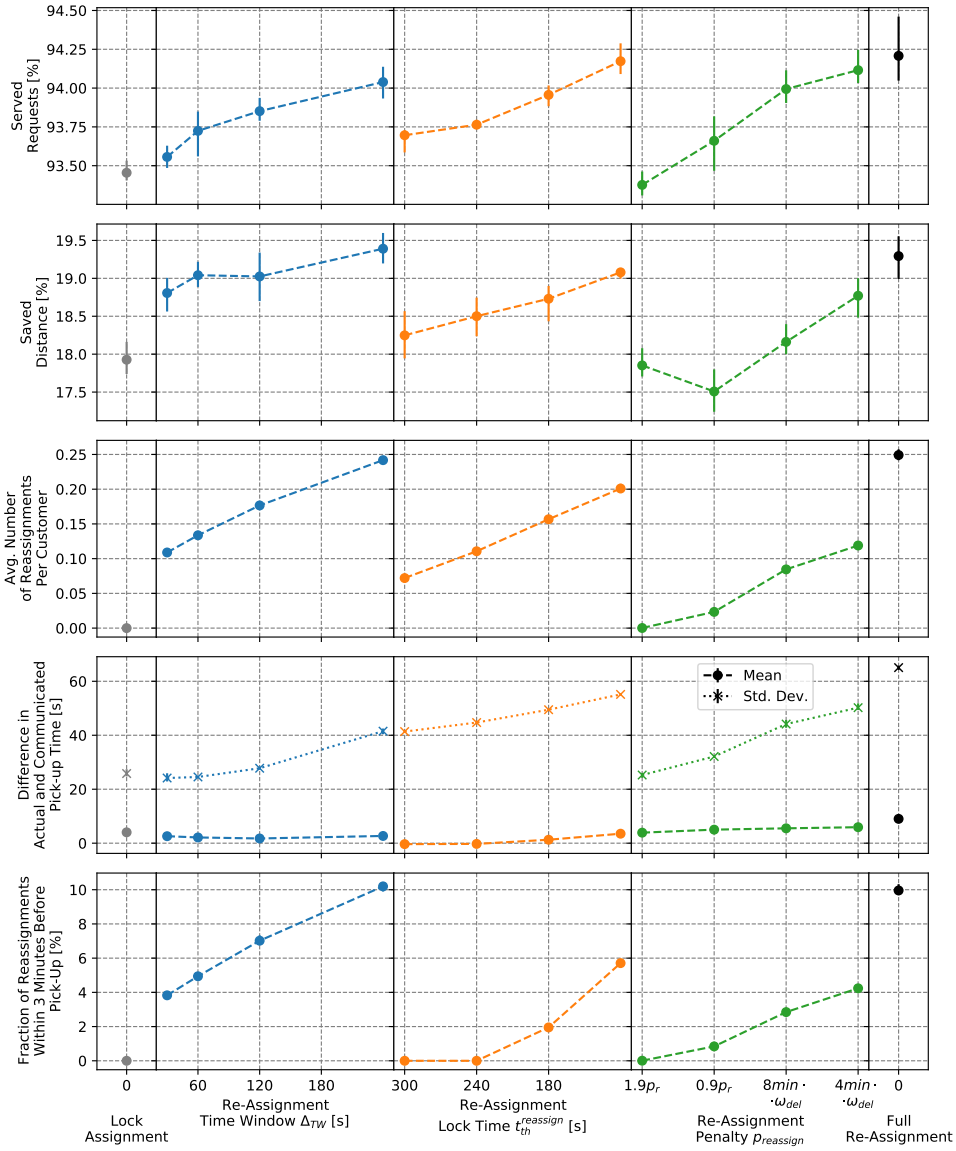


Figure II.2: Comparison of methods to limit re-assignment for improved customer convenience. Munich - 5% Demand Penetration - Fleetsize: 600.

III Further Results - Repositioning

Calibration of Parameters

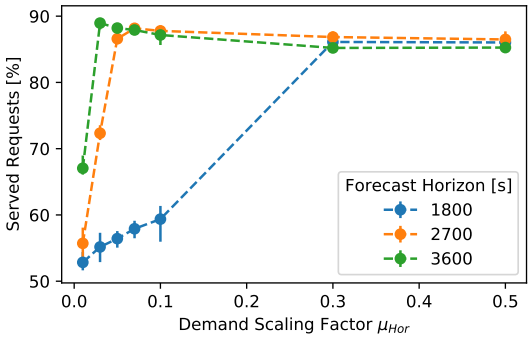
Figures III.1 and III.2 show the calibration of the demand scaling factor and repositioning forecast horizon parameters for the *Hor*-Method and the *QT*-Method, respectively.

Figure III.1 shows the calibration of the demand scaling factor μ_{Hor} and the forecast horizon $\mathfrak{H}_{Hor}$ for the *Hor*-Method. A strong dependency on the service performance is notable, especially with respect to μ_{Hor}. As this method repositions idle vehicles proportional to expected demand scaled by μ_{Hor}, a sharp drop in served requests is observed for low values of μ_{Hor}. On the contrary, if μ_{Hor} is chosen too high, the number of served requests settles, but VKT increases as too many repositioning trips are assigned. Concerning the forecast horizon $\mathfrak{H}_{Hor}$, a sharp increase in served requests is observed in the Chicago case study when increasing the forecast horizon from 1800s to 2700s (and a low value for μ_{Hor}). In this regime, a forecast horizon of 1800s is insufficient for repositioning to cover the large operating area of Chicago.

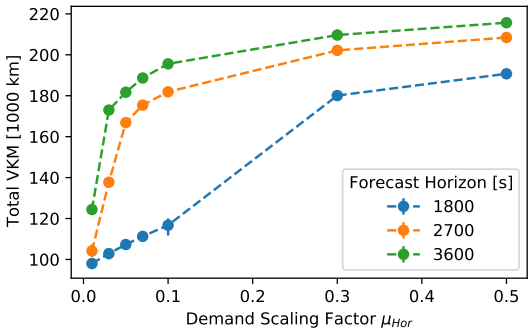
As a trade-off between served requests and VKT, the values $\mu_{Hor} = 0.05$ and $\mathfrak{H}_{Hor} = 2700s$ are chosen for the calibration of the *Hor*-Method.

Similarly, Figure III.2 shows the calibration of the demand scaling factor μ_{QT} and the forecast horizon $\mathfrak{H}_{QT}$ for the *QT*-Method. As this method tries to balance idle vehicles on a relative measure of expected demand, this method is less sensitive to the demand scaling factor μ_{QT}. It is also less sensitive to the forecast horizon $\mathfrak{H}_{QT}$, as the method does not constrain repositioning trips with trip durations exceeding the forecast horizon.

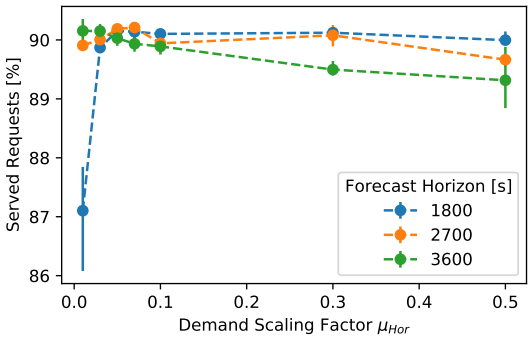
The values $\mu_{QT} = 0.3$ and $\mathfrak{H}_{QT} = 2700s$ are chosen for the calibration of the *QT*-Method as they show a good trade-off between served requests and VKT in all case studies.



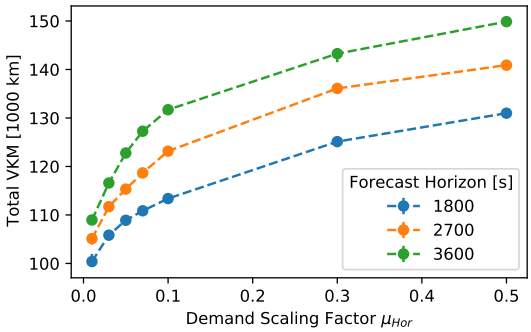
(a) Chicago - 20% Dem. Pen. - 340 veh



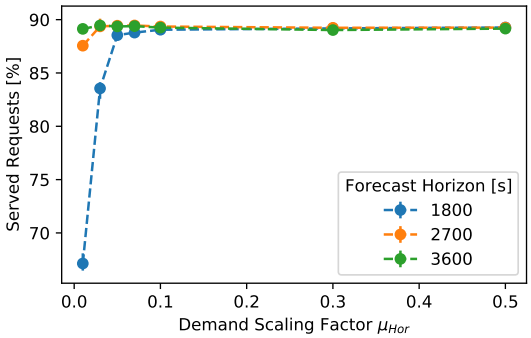
(b) Chicago - 20% Dem. Pen. - 340 veh



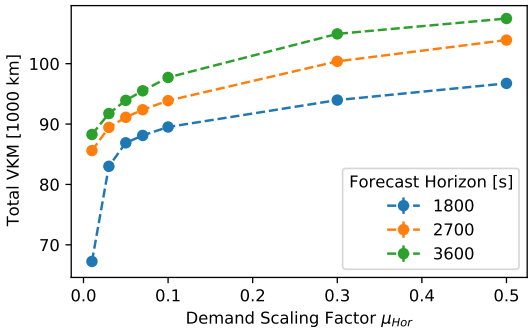
(c) Munich - 2% Dem. Pen. - 240 veh



(d) Munich - 2% Dem. Pen. - 240 veh

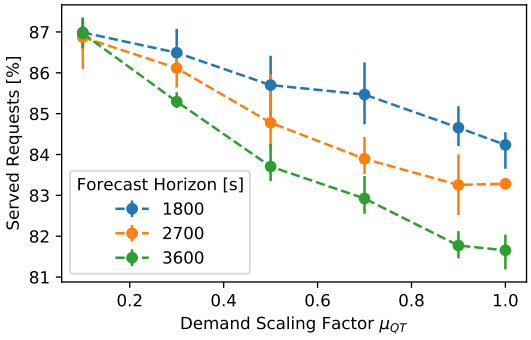


(e) Manhattan - 20% Dem. Pen. - 320 veh

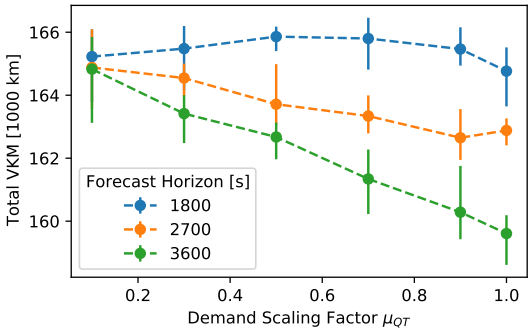


(f) Manhattan - 20% Dem. Pen. - 320 veh

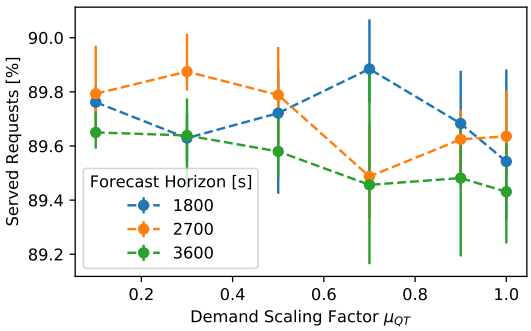
Figure III.1: Calibrating Factors for *Hor*-Method.



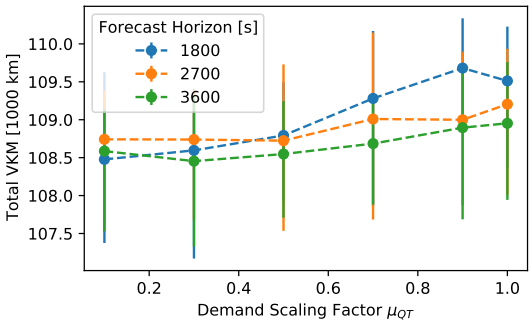
(a) Chicago - 20% Dem. Pen. - 340 veh



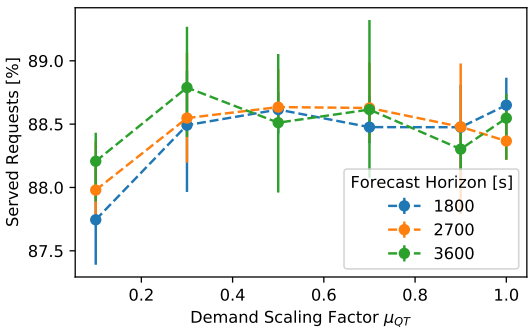
(b) Chicago - 20% Dem. Pen. - 340 veh



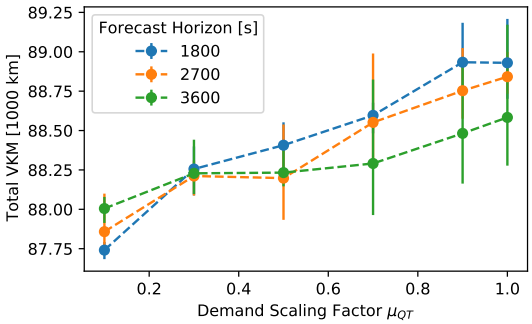
(c) Munich - 2% Dem. Pen. - 240 veh



(d) Munich - 2% Dem. Pen. - 240 veh



(e) Manhattan - 20% Dem. Pen. - 320 veh



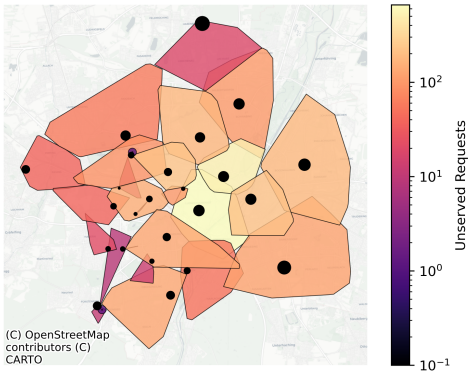
(f) Manhattan - 20% Dem. Pen. - 320 veh

Figure III.2: Calibrating Factors for QT-Method.

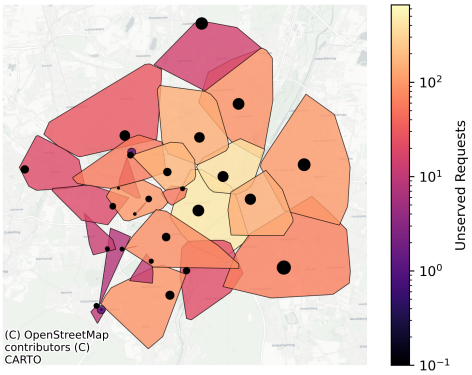
Impacts of Repositioning

As extension to the evaluation of the impact of repositioning on the Chicago case study in section 5.3.2, Figures III.3 and III.4 show the spatial and temporal impact of repositioning for Munich and Manhattan, respectively.

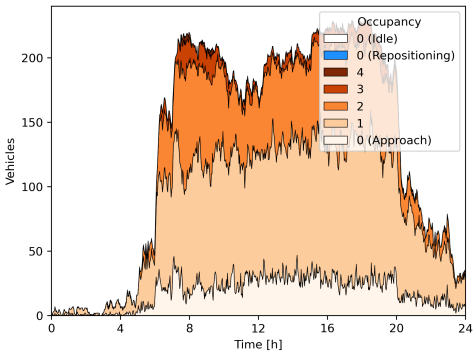
For Munich (Figure III.3), it has been evaluated that repositioning is less vital. Nevertheless, repositioning improves the service rate, especially in the outskirts of the city. The temporal distribution of vehicle occupancy shows the typical morning and evening peaks from the underlying private vehicle trip demand pattern. Also without repositioning, the fleet is highly utilized during these times resulting in high vehicle revenue hours. Nevertheless, idle vehicles still remain during these times and are utilized once repositioning is applied.



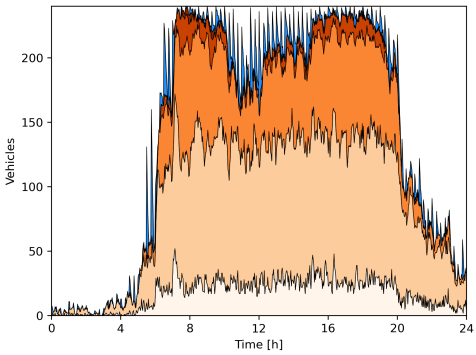
(a) No Repo.



(b) With Repo.



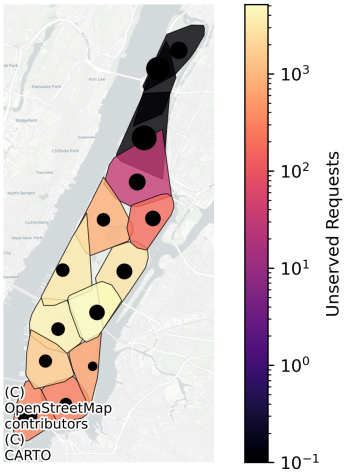
(c) No Repo.



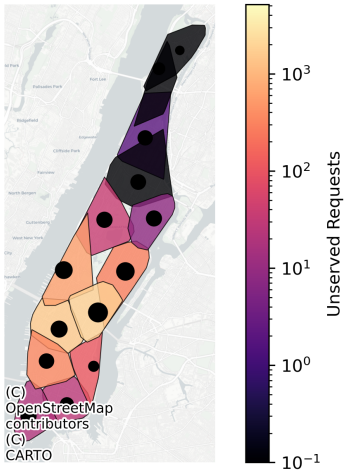
(d) With Repo.

Figure III.3: Spatial and Temporal Impact of Repositioning for Munich with 240 Vehicles. (Extension of Figure 5.9)

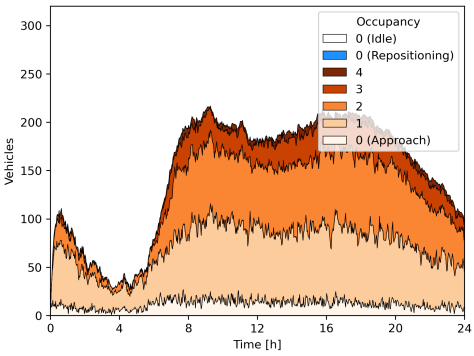
For Manhattan (Figure III.4), the impact of repositioning is more significant. Without repositioning, the number of unserved requests is high, especially in the center of Manhattan. At the same time, idle vehicles (indicated by the size of black circles) are observed in the north of Manhattan. The temporal distribution shows that only up to 240 of 320 vehicles are utilized during peak times without repositioning. If repositioning is applied, vehicle utilization increases to close to 100%, allowing to serve more requests, especially in the center of Manhattan.



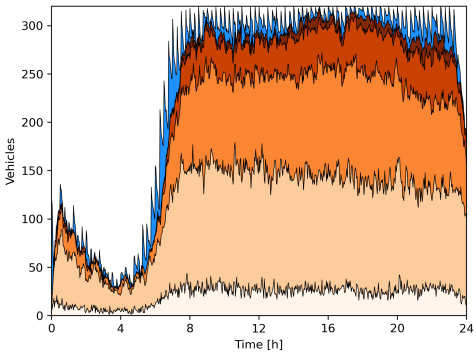
(a) No Repo.



(b) With Repo.



(c) No Repo.



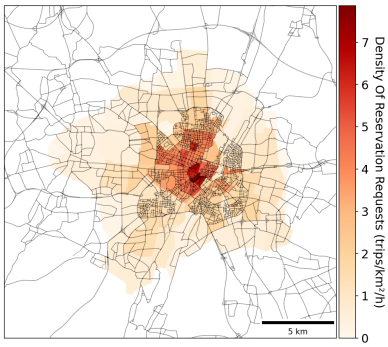
(d) With Repo.

Figure III.4: Spatial and Temporal Impact of Repositioning for Manhattan with 320 Vehicles. (Extension of Figure 5.9)

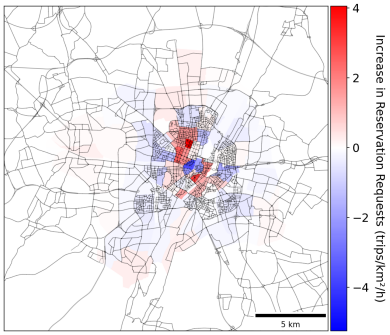
IV Further Results - Reservation

Scenarios and Parameters

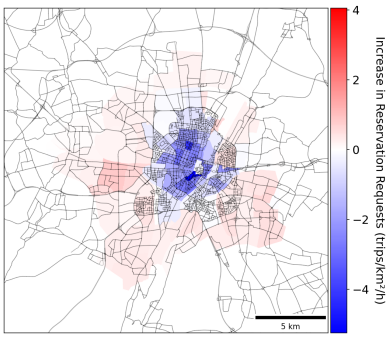
In extension to the discussion in section 5.4.1, Figure IV.1 and IV.2 show the spatial and temporal distributions of pre-booking customers when applying uniform, low shareability, and high shareability distributions to create reservation requests for the Munich and Manhattan case studies, respectively. Similar to the discussion provided in section 5.4.1, the number of reservation requests tends to be higher in high-demand areas and peak times when applying the high shareability distribution. In Munich, this is temporally notable in the morning and evening, while spatially more pre-booking customers are created in the city center. In Manhattan, the high shareability distribution creates more reservation requests in the late evening and in the center of Manhattan.



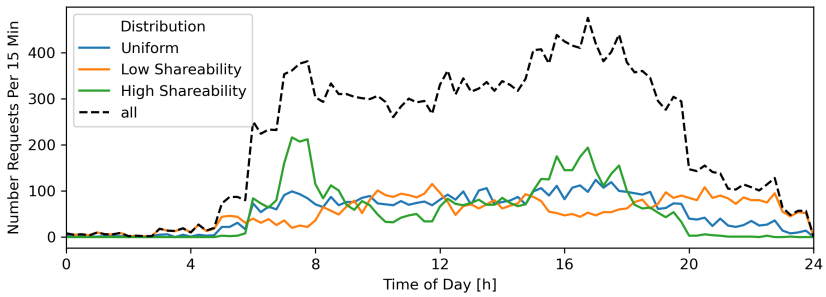
(a) Uniform Distribution: Reservation Request Density.



(b) High Shareability Distribution. Increase in Reservation Request Density compared to Uniform Distribution.

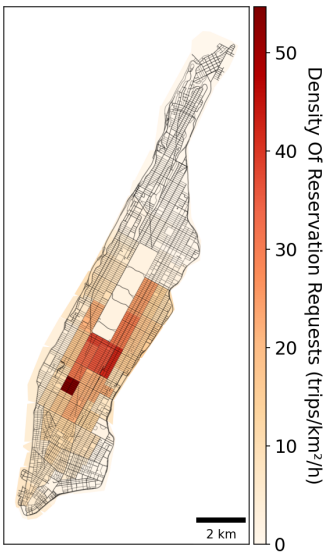


(c) Low Shareability Distribution. Increase in Reservation Request Density compared to Uniform Distribution.

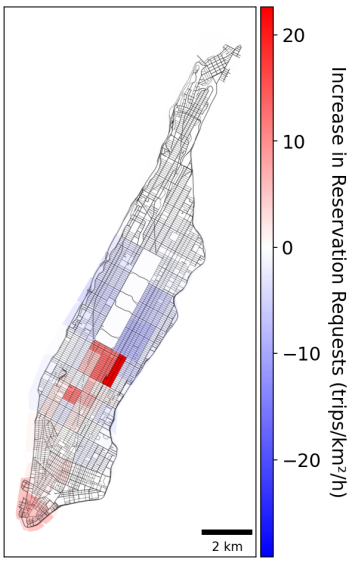


(d) Temporal distribution of reservation times of pre-booking requests for the different generating distributions in 15 min bins.

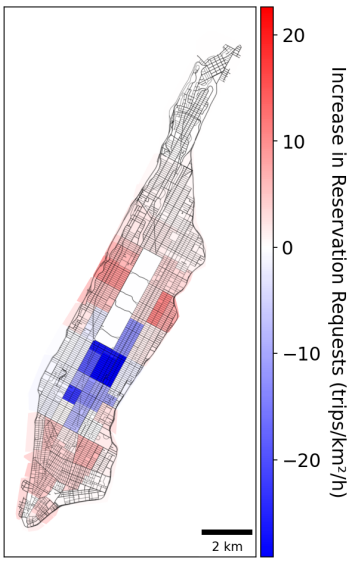
Figure IV.1: Spatial and temporal distributions of pre-booking customers when applying uniform, low shareability and high shareability on the overall request set for the Munich Case Study. A share $S = 25\%$ reservation requests is used for all cases.



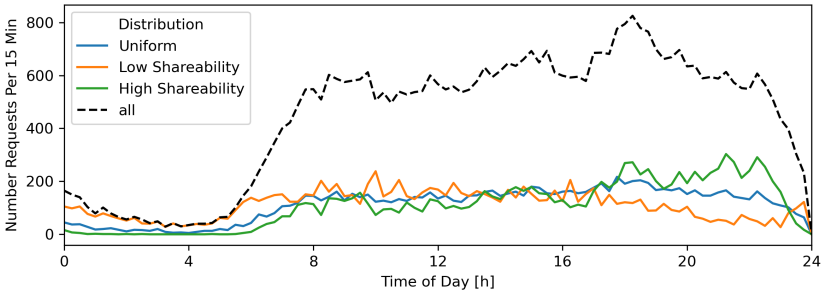
(a) Uniform Distribution: Reservation Request Density.



(b) High Shareability Distribution. Increase in Reservation Request Density compared to Uniform Distribution.



(c) Low Shareability Distribution. Increase in Reservation Request Density compared to Uniform Distribution.



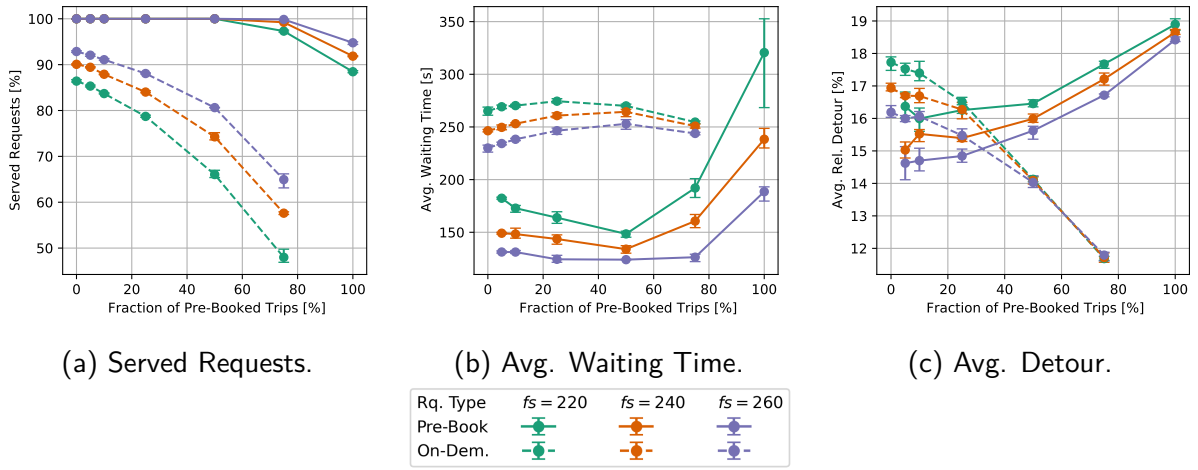
(d) Temporal distribution of reservation times of pre-booking requests for the different generating distributions in 15 min bins.

Figure IV.2: Spatial and temporal distributions of pre-booking customers when applying uniform, low shareability and high shareability on the overall request set for the Manhattan Case Study. A share $S = 25\%$ reservation requests is used for all cases.

Impact of Long-Term Reservations

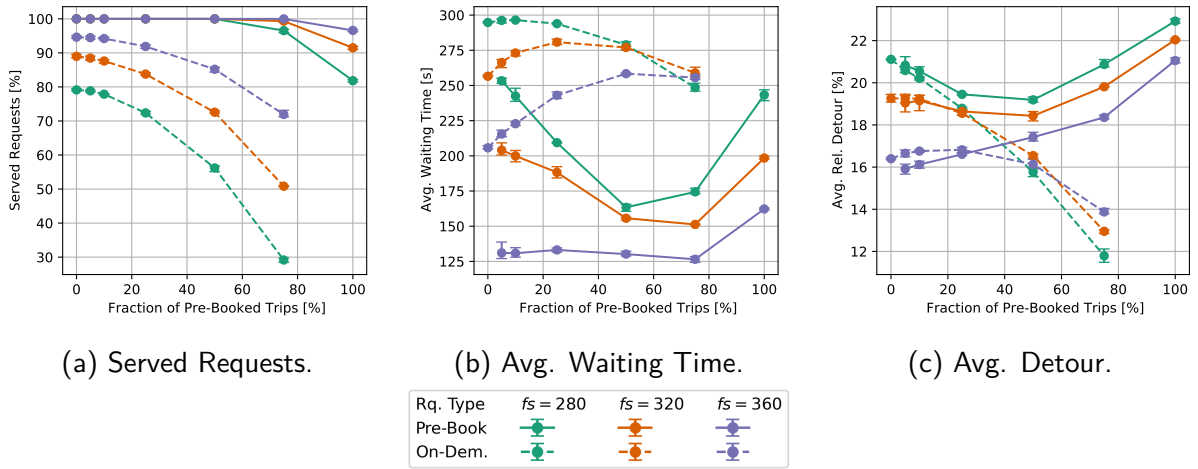
In extension to the results in section 5.4.2 and Figure 5.16 for the Chicago case study, Figures IV.3 and IV.4 show the impact of long-term reservations for on-demand and reservation requests for the Munich and Manhattan case studies, respectively.

Similar to the Chicago case study, the number of served reservation requests drops if the fraction of reservation requests exceeds 50%. The service rate for on-demand requests is not significantly impacted by the share of reservation requests and can drop to below 30% for the Manhattan case study with 280 vehicles operated. As the approach of a vehicle is planned early on, the waiting times tend to be lower for reservation requests. Nevertheless, the average detour increases for reservation requests as the chances of finding shared schedules increase.



(a) Served Requests. (b) Avg. Waiting Time. (c) Avg. Detour.

Figure IV.3: Different KPIs for the Munich Case Study with Homogeneous Reservation Request Distribution.



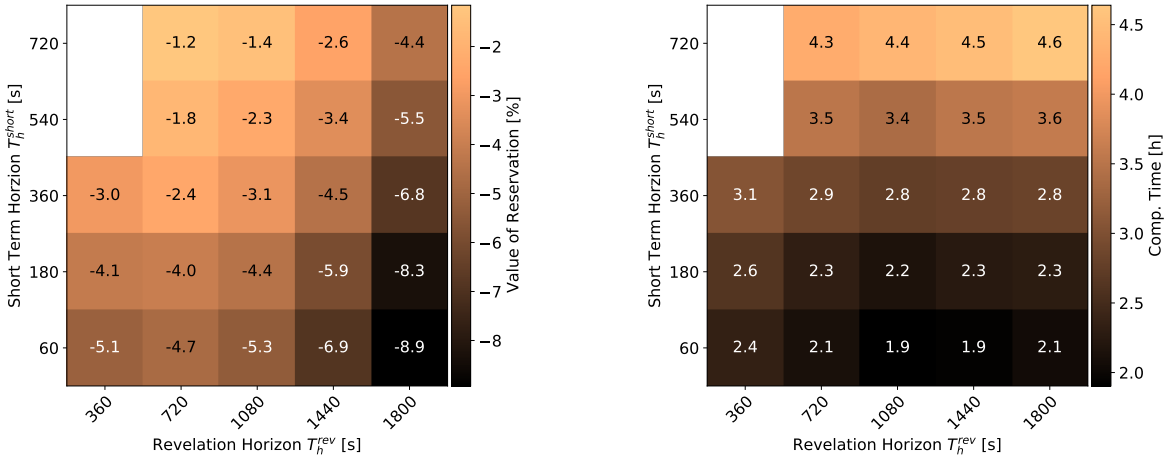
(a) Served Requests. (b) Avg. Waiting Time. (c) Avg. Detour.

Figure IV.4: Different KPIs for the Manhattan Case Study with Homogeneous Reservation Request Distribution.

Evaluation of Rolling Horizons

In addition to Figure 5.22 in section 5.4.7, Figure IV.5 and IV.6 show the impact of the rolling horizon parameters on the Munich and Manhattan case studies, respectively.

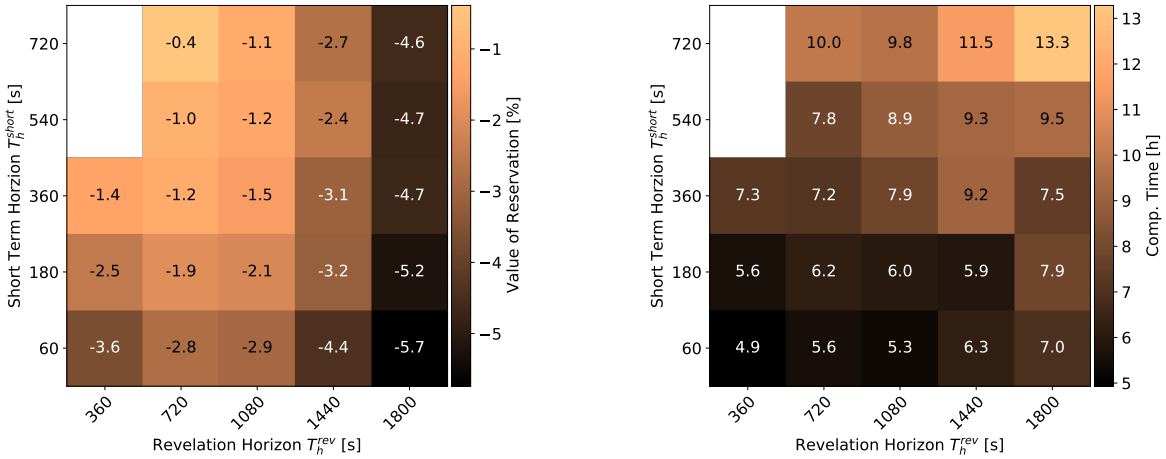
Similar observations to the Chicago case study can be made: The value of reservation increases with the short-term horizon T_h^{short}. Another increase is observed with the revelation horizon T_h^{rel} until 720s in the case studies for Munich and Manhattan. For the Chicago case study, a value of $T_h^{rel} = 1080$s has been observed, likely due to longer trips that are beneficial to be revealed earlier to the online optimization. With increasing horizons, the overall computational time also increases. Especially for the Manhattan case study, a substantial increase can be observed. As the Manhattan case study consists of a high density of short trips, the rise in the number of active requests in the optimization problem is higher, resulting in a non-linear increase in computational time.



(a) Value of Reservation.

(b) Overall Computational Time.

Figure IV.5: Impact of Horizons for Munich Case Study with 25% Reservation Requests from the Homogeneous Distribution and 240 vehicles.



(a) Value of Reservation.

(b) Overall Computational Time.

Figure IV.6: Impact of Horizons for Manhattan Case Study with 25% Reservation Requests from the Homogeneous Distribution and 320 vehicles.