Technische Universität München
TUM School of Engineering and Design

On-Demand Mobility Service Evaluation for the Ride Hailing and Ride Pooling Use Case with a Novel Diffusion Customer Model

Marvin Vincent Dominik Erdmann

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. Markus Lienkamp

Prüfer*innen der Dissertation:

1. Prof. Dr.-Ing. Klaus Bogenberger
2. Ass. Prof. Dr. Michael Hyland, University of California, Irvine, USA

Die Dissertation wurde am 10.11.2022 bei der Technischen Universität München eingereicht und durch die *TUM School of Engineering and Design* am 26.03.2023 angenommen.

Executive Summary

The rising population in urban areas around the world amplifies issues that are present nowadays, like the lack of housing and the ecological impact of large cities. The problem of increasing traffic is also worsening, especially for growing cities and their citizens. On-demand mobility (ODM) services have the potential to be a part of the solution of this problem, addressing needs of the key stakeholders: city authorities aim to decrease the net vehicle mileage, service providers try to maximize the profitability, while offering a high-quality service that is perceived well by the users. Ride hailing and ride pooling service models are offered by various companies and studied extensively in literature. In this work, a research background is provided, including overviews of the state of the art in regard to problem formulations, optimization techniques and current ODM service models.

One challenge for ODM service models is to operate a service that combines the optimization of assignments of service vehicles to requests while being able to quickly respond to customers. In the literature, ODM service simulations are often based on very simplistic customer models that can impair the plausibility of results in studies of ODM services. Hence, this work aims to answer two research questions:

- How to design an ODM service model in order to bring together optimal request assignments and quick responses to users, considering both ride hailing and ride pooling use cases?
- How to improve the comparability of system performances of ODM service models to real-world applications and how to measure the impact of customer models on the system performance?

Three service models for the ride hailing and ride pooling use cases are compared in agent-based simulations within a case study of Manhattan, using the open source New York City Taxi data set. These service models differ in their respective way to communicate and interact with customers. Service Model 1 is based exclusively on heuristic assignments of vehicles to user requests. This allows the service operator to quickly respond to requests and to immediately send out exact pickup information. Service Model 2 combines heuristic assignments with periodic global optimization in a 2-step service model. That model includes initial service offers with projected pickup time windows based on heuristics. In a second offer, users receive their exact pickup times after vehicles potentially have been reassigned according to the outcomes of global optimization. In Service Model 3, the assignments of vehicles to requests is based entirely on global optimization. This implies that many customers do not receive any immediate response to their service requests. Instead, the service operator decides at the end of each optimization period which of the new requests are served by which vehicles according to the currently optimal solution to the dynamic assignment problem.

The system performances of these service models are compared with respect to key performance indicators (KPIs), such as daily profit, percentage of customers served, empty vehicle mileage, and user waiting time in scenarios with 10 % of the demand from the NYC taxi data set and 100 to 500 vehicles. In simulations with a conventional customer model, in which every offer that implies a pickup waiting time that is shorter than a certain maximum waiting time is immediately accepted, results indicate that the differences between service models of each of the use cases in most of these KPIs are rather small in total scale. The profitability of Service Model 3 is up to 3 % (2 %) higher than that of Service Model 1 in the ride hailing (ride pooling) use case, mainly due to the higher percentage of customers served. However, because of the implied delay in response times, Service Model 3 performs worst in average user waiting time until pickup, which is one of the most relevant measures of the perceived quality of the service from user perspective. In many KPIs, Service Model 2 performs as good as Service Model 3 or better, and does not imply late responses due to the use of heuristics. A parameter sensitivity analysis is conducted for Service Model 2 in both use cases. For most evaluated parameters, the service model is found to be resilient against changes, especially in the ride hailing use case. In the ride pooling use case, modifications of the objective weight of distance in the objective function are found to have a significant impact on the system performance.

In order to address the second research question, a diffusion customer model is introduced. In contrast to conventional customer models, it allows to take into account the individual decision-making process of service users. The analysis of the interaction between customers modeled in this way and service operators in three different service models is one of the main contributions of this work. The decision-making process is simulated by means of a diffusion model, based on a drifting random walk. This means, the probability of accepting an offer as well as the average decision duration is determined by the quality of the offer provided by the service operator and certain model parameters that define the characteristics of decision-making of customers. Offers that are perceived as good imply a positive drift in the random walk model and are more likely to be accepted than offers that imply long waiting times until pickup and are therefore perceived as bad. If an offer is neither particularly good nor bad, customers in this model tend to take longer in their decision making, because the associated drift in the random walk model is smaller in value. Longer decision durations imply uncertainty for service operators as they need to decide if and when to send vehicles to potential pickup locations. On the one hand, they want to avoid delayed pickups due to hesitance, on the other hand, if a vehicle is on its way to a pickup location and the respective user rejects the offer, unnecessary mileage is produced, which adds traffic in the business area and costs for the service provider. The diffusion customer model represents real-world circumstances in the context of ODM services better than conventional customer models. The findings of simulations of ODM services that use this customer model are therefore assumed to be more reliable and conclusive than findings of studies that make use of conventional customer models.

Results conducted in simulations with Service Model 2 indicate that the more diverse the modeled offer qualities, the more the system performance varies. In both use cases, ride hailing and ride pooling, KPIs like daily profit, requests served, empty mileage and correctly predicted pickup time windows are worse in scenarios with diffusion customer models with higher diversity of modeled offer qualities, while the average user waiting time until pickup

decreases. These observations indicate that the diffusion customer model works as expected: customers that receive “bad offers” (with long associated waiting times) are more likely to reject these offers, which reduces the average waiting time of all customers served, but also the overall number of requests served compared to customer models in which the offer quality is less diverse. In the latter case, requests that cannot be served quickly are more likely to be rejected by the operator immediately, which reduces the risk of rejections by the customers and leaves more freedom to optimize the assignments of vehicles, leading to fewer empty mileage and more profit. In a sensitivity analysis of two critical parameters of the diffusion customer model, decision durations and acceptance rates are found to be correlated with the modeled decisiveness of individual service users. Such customer-specific characteristics are another advantage of the diffusion customer model compared to conventional customer models.

This work answers both of its research questions: in a detailed study of three service models in the ride hailing and ride pooling use cases, a conclusive comparison between their respective strengths and weaknesses is presented. A 2-step service model that combines the advantages of quick initial assignments based on heuristics with the potential of global optimization is found to perform better than purely heuristic approaches while avoiding long response times for users implied by service models based exclusively on global optimization. A diffusion customer model is introduced, which is able to simulate the decision-making process of service users more accurately than conventional customer models. This improves the comparability of system performances in studies about ODM service models.

The findings of this work can be used in other studies and the diffusion customer model can be adopted to different service models. The characterization of various service user groups and the potential to prioritize and target them optimally is an intriguing research area for the future. Another aspect of the diffusion customer model that needs to be studied in more detail is the response-strategy of service operators. If rejections of offers can be predicted more accurately and service operators prioritize other requests instead, empty mileage can be avoided. This work lays the ground work for a variety of potential follow-up research and allows service providers to make better predictions about the real-world performance of simulated service models.

Contents

Chapter 1

Introduction

In our globalized world more and more people can move and travel wherever they want. Since cities often offer better job opportunities, infrastructure and other benefits, there is a trend of urbanisation [United Nations, 2019], which means that the population of cities all over the world increases relative to the rural population. This development amplifies the problems already existing in cities nowadays, such as gentrification, a lack of housing, ecological and social conflicts, as well as traffic.

Urban mobility has always been a very important aspect of city planning, because it directly affects the quality of life of the citizens. It has an impact not only on people who are actively moving from one point to another and are spending a considerable amount of life time either in – or waiting for – a public transportation vehicle, riding their bike on often insecure bike lanes or sitting in their own vehicle during rush hour congestion. Residents are also affected by it when staying at home due to air pollution, noise emission and a reduction of usable space within the city – space that could be used for parks, playgrounds or community areas instead of broad streets and parking vehicles.

Therefore, in the last decades many cities all over the world started to search for alternatives to the way mobility worked so far. Some of the examples include: building more bike lanes and improving the safety of people using them [National Institute for Transportation and Communities, 2014], expanding the public transportation sector, improving their schedules and frequencies, modernizing railroads and vehicles, and exploring new alternatives to traditional transportation modes. Meanwhile, the continuous digitalization and rapidly evolving technologies are changing every aspect of daily life, especially since the rise of internet at the start of this millennium. One of the businesses that is transformed by this development is the transportation sector.

Concepts like "smart mobility" [Porru et al., 2020] and applications for multi-modal transit [Pinto et al., 2020] are facilitating mobility for people, especially in urban areas, while also allowing cities to optimize traffic for the existing infrastructure. Last-mile solutions emerge for logistics and transit, opening a whole new business area of so-called "micro-mobility" [Shaheen et al., 2020]. However, one of the most disruptive trends in the industry might be autonomous driving. Fleets of self-driving cars on the streets would potentially increase safety and reduce emissions due to a more regulated and cleaner style of driving than average human drivers. While in transit, passengers of autonomously driven vehicles could use their time to work, relax or even sleep. A large-scale realization of this vision is the goal for many competitors in the automotive and tech industry, but it is uncertain how long it will take to get there [Coppola and Silvestri, 2019].

Meanwhile, more and more people are gaining access to high speed mobile internet. Especially in urban areas, the rate of internet users rises continuously and the mobile-broadband network coverage is close to a hundred percent [International Telecommunication Union and United Nations, 2021]. This enables online service providers to make service offers for all aspects of life to almost everyone owning a smartphone in urban environments. The trends of sharing economy and on-demand services are results of this development, both of which have a direct impact on urban mobility.

On-Demand Mobility Services and Their Role in Modern Urban Traffic

Companies such as Uber and Lyft compete in the market of on-demand mobility (ODM). ODM service providers offer the benefits of individual door-to-door transportation without the need for users of the service to own a car. This provides a crucial complement to traditional modes of transportation in cities. The market for ODM services is currently worth more than $140 billion in China, Germany and the USA alone [Accenture, 2020], with an anticipated compound annual growth rate of between 15 and 28 % [Arthur D. Little GmbH, 2020].

This enormous economic potential only increases considering the future outlook of autonomous vehicles, which would reduce the costs for drivers of ODM vehicles effectively to zero. Hence, the competition between companies willing to take part in this business is fierce. More often than not the aggressive expansion to new cities and service areas as well as reports about bad working conditions for employees cause conflicts with local taxi drivers and politics, ultimately leading to bad reputation and sometimes costly penalties and the withdrawal of local services.

Nevertheless, service providers like Uber and Lyft have become a vital part of urban mobility, especially in the USA, because they offer some major improvements in ease of use for their users. Without making a commitment to privately own a car – including high costs for purchasing, maintenance and insurance – users are promised to experience the same level of mobility and reliability when using an ODM service. They also avoid the tedious search and payment of parking in urban centers, which tends to take a considerable amount of time and is often expensive. Another benefit of ODM services is the fact that people that are not able to drive a car by their own – be it permanently because of disabilities or age, or situational because of alcohol or drugs – can still be mobile within the service areas.

In comparison to taxi services, which offer similar advantages and are also widely available, ODM companies offer transportation services at lower fares for the customer, while being able to often provide shorter customer waiting times and a more convenient booking procedure. This allows parts of society to use individual transportation that traditionally could not afford it or were otherwise excluded from it.

ODM services are most commonly categorized in two types: ride hailing and ride pooling. In ride hailing services, users are guaranteed to be transported directly from their pickup location to their destination, without sharing their rides with other groups of people. In the ride pooling use case, rides can be shared between service users, which typically increases the average number of customers on board and the efficiency of the service fleet. Such services imply detours for customers, but are often offered at cheaper conditions.

Since ODM services have become such an important part of the transportation system, especially in metropolitan areas, much recent research focus on their effects on traffic as well as the targeted user group. Most of them show that ODM services are actually increasing the amount of vehicles and driven on-street mileage in many cities [Schaller Consulting, 2018a; Schaller Consulting, 2018b; Henao and Marshall, 2018]. It is found that most customers are not substituting their own vehicle, but are using ODM services instead of public transportation, biking, walking or would not have made the trip in the first place. The additional vehicle mileage – much of which is driven emptily due to pickup trips of ODM vehicles – often leads to an even greater traffic problem, making ODM providers a part of the problem they are claiming to solve with their services.

The COVID-19 pandemic had an immense impact on the transportation sector and urban traffic [McKinsey Center for Future Mobility, 2020a], especially on ODM services. The risk of infections became the most important reason to choose a transportation mode [McKinsey Center for Future Mobility, 2020b], which lead to a significant increase in the relative share of rides being made in private vehicles compared to other modes of transportation, including ride hailing and ride pooling. At the same time, the overall traffic in urban areas was drastically reduced [Texas A&M Transportation Institute, 2021], which lead to less congestion and fewer accidents, showcasing the potential benefits of fewer vehicles in urban street networks, which is what ODM services claim to be aiming for.

The challenges for ODM service providers as well as cities willing to include such services in their urban transportation system are therefore manifold. On one hand, in order to contribute to an improved traffic efficiency in urban areas, ODM providers have to make sure their services are not producing significantly more vehicle mileage than needed in order to serve the current demand for mobility. This can be achieved by minimizing empty vehicle mileage, improving fleet utilization and increasing the average occupancy of ODM vehicles. On the other hand, in order to run a successful long-term business, ODM services need to be profitable, which implies that the service also needs to be optimized in respect of minimizing the fleet size and maximizing the number of customers served and overall revenue generated.

Meeting the local regulatory requirements for sustainability and traffic reduction, while providing a high quality service and running a profitable business is not a trivial task. Therefore, most ODM service providers use an optimization algorithm to operate their fleets. Such an algorithm needs to be able to both handle high user demand fast enough to allow quick responses to customer requests and find optimal assignments in order to minimize fleet costs and maximize the revenue for the operator.

Most currently used ODM service models allow only one of these goals to be met. In one version, customers have to wait a considerable amount of time before getting a response to their service request while the algorithm tries to find an optimal match. In another model, an assignment is found very quick, but the algorithm cannot guarantee that the assigned vehicle is globally optimal in terms of service profitability and/or quality, because the search procedure to find this match was designed to be quick, not exact. On top of this dilemma, there are plenty of uncertainties for both customers and providers of ODM fleets:

- Unforeseen incidents like traffic jams or accidents might delay the pickup of waiting customers, who then need to be updated about changing pickup times, which decreases the perceived service quality and may even lead to cancellations, and therefore a loss in revenue for the operator.

- In case the ODM vehicles are driven by human drivers, there is always a chance for human failures. Taking wrong routes, responding late or not at all to assigned pickup tasks or even being unfriendly in conversations with customers might impair the service experience of users, decreasing the probability of using the service again or recommend it to friends causing long-term losses for the service providers.

- From an operator's perspective, customers can affect the seamless service process in multiple ways. Cancellations, late arrivals at pickup locations or no-shows have the potential to not only cost the service provider the revenue of this one service request but also might delay later planned pickups and increase the empty mileage driven by the ODM fleet.

- Another variable in customer behavior is the amount of time it takes the customer to accept or decline an offer made by the operator in response to the request. In addition, the decision to accept or decline an offer altogether might vary from one customer to another, adding another layer of uncertainty for service operators.

Since it is most often not a practical option to test new algorithms and service models with real customers in a live service, tests are normally run using simulations. These simulations are designed as confined environments, that represent certain aspects of the real world while being simple enough to be comprehensible and reproducible, yet complex enough to be plausible and conclusive. Operators evaluate service parameters and scenarios, comparing the system performance in terms of indicators such as driven mileage and customers served, as well as computation time. This helps to make future business decisions and presents transparent and compelling arguments to cities and stakeholders. Hence, well designed simulation frameworks and models are crucial for ODM service providers in order to make reliable predictions and suitable decisions.

Research Questions

As the role of ODM services in urban traffic systems became increasingly important and the value of the business for service providers increased, the research interest in this area grew considerably throughout the last decades. This work focuses on two main research questions (RQ).

The first research question (RQ1) addresses the problem that service providers need to make sure that their service (a) is profitable in order to run a successful business and therefore want to optimize the assignments of requests and vehicles while (b) providing quick responses to customer requests for a good service experience. These conflicting service goals need to be met for both ODM use cases considered in this work, the ride hailing and the ride pooling use case. RQ1 can be summarized as follows:

> *How to design an ODM service model in order to bring together optimal request assignments and quick responses to users, considering both ride hailing and ride pooling use cases?*

This work tries to answer RQ1 with the following research contributions:

- Introduction and implementation of three ODM service models for the ride hailing and ride pooling use cases
- Incorporation of all service models into a modular simulation framework
- Definition of key indicators to measure system performances
- Implementation of a case study of the three service models for both use cases by means of simulations of the services within the business area of Manhattan
- Parameter sensitivity analyses of service model parameters
- Examination of all service models considered with respect to profitability and user experience, as well as their impact on urban traffic and the environment

The second research question (RQ2) focuses on the plausibility of models used to simulate ODM services, especially the aspect of the customer model. As conventional customer models include many assumptions and simplifications that affect the system performance, they are prone to overestimate the capabilities and potentials of any method evaluated during a simulation. Service providers using such customer models might find that their service in fact is not profitable or performs worse than expected in other key categories, like saving mileage. RQ2 is stated as follows:

> *How to improve the comparability of system performances of ODM service models to real-world applications and how to measure the impact of customer models on the system performance?*

In this work the following steps are taken in order to answer RQ2:

- Explanation and implementation of a novel diffusion customer model
- Definition and setup of model parameters
- Evaluation of key performance indicators in comparison to the conventional customer model considered in the ride hailing and ride pooling use case
- Model evaluation and parameter sensitivity analysis of the diffusion customer model
- Interpretation of the results and the analysis of the impact for service providers

This work answers these research questions by running and evaluating agent-based simulations in a modular framework. The simulation framework is designed to compare service models in the ride hailing and the ride pooling use cases.

Structure of the Work

The structure of this work is as follows. In Chapter 2, an in-depth review of the literature of research topics touched during this work is presented, providing insight to the current state of the art in solving similar problems. Chapter 3 describes the evaluated services, the underlying optimization problem and the conventional customer model, which is used as a base model, as well as the case study, including the simulation framework, data, parameters and key performance indicators. The subsequent Chapters 4 and 5 present the results obtained for the ride hailing and ride pooling use case, respectively, using the conventional customer model. Each provides a description of use-case-specific assignment approaches, an evaluation of all service models considered and a parameter sensitivity analysis for one of them. In Chapter 6, the diffusion customer model is introduced in detail and evaluated for both use cases. The work concludes with a discussion of the results and their implications and an outlook to what questions remain open for future research (Chapter 7).

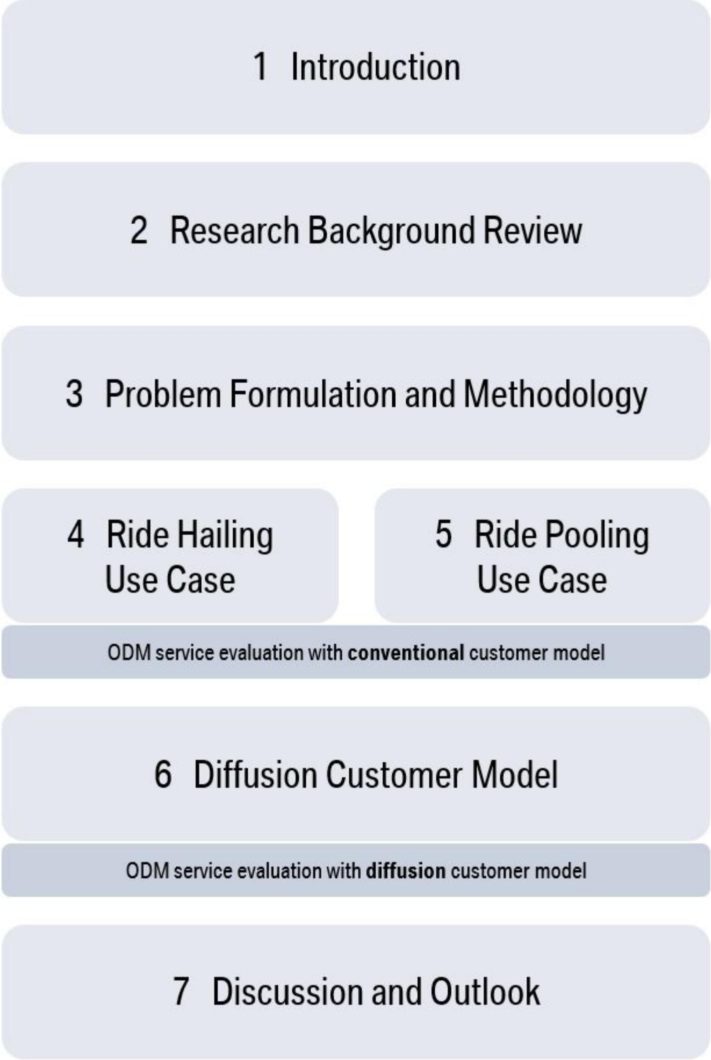

1 Introduction
2 Research Background Review
3 Problem Formulation and Methodology
4 Ride Hailing Use Case
5 Ride Pooling Use Case
ODM service evaluation with **conventional** customer model
6 Diffusion Customer Model
ODM service evaluation with **diffusion** customer model
7 Discussion and Outlook

Figure 1.1: Overview of the structure of the work.

Chapter 2

Research Background Review

With the increasing interest in the potential of ODM services during recent years and decades, the research area became rich with contributions. Because of the complexity of the topic, as well as the sheer endless number of models, approaches and applications there is plenty of literature on many of the aspects of the problems and research questions this work is trying to solve and answer.

In order to understand the formal optimization problem in the core of ODM operations, the first section of this chapter will give a short introduction to the taxonomy of the problem and the family of vehicle routing problems. It connects the origin of mathematically formulated logistic problems, the traveling salesperson problem, with the most recent variations of dial-a-ride problems and the problem formulation used in this work.

The subsequent section focuses on the approaches and techniques that were used in recent works to solve these optimization problems. It presents the backgrounds of heuristics, metaheuristic procedures and exact optimization software, including the methods that are part of the methodology used in the model examined in this work. The section also presents alternative approaches and summarizes the respective strengths and weaknesses.

The last section of this chapter reviews service models and the evaluation of simulated ODM services in the literature. It highlights service models that are currently standard in the industry, algorithms that proved to outperform the competition, and introduces fleet and customer models that are used as base models in subsequent chapters.

2.1 Problem Formulation

One of the first mathematical formulations of a transportation problem was published in 1949, titled "On the Hamiltonian game (a traveling-salesman problem)" [Robinson, 1949]. Thereafter, the term "traveling salesperson problem" (TSP) was used to describe an optimization problem in which it is the goal to minimize the total length of connecting lines (travel distance) between points in a network, starting from and finishing at one specific point (the salesperson), so that all points in the network are connected. As simple as this problem sounds, research struggled to find a generic approach to find globally optimal solutions for this kind of problem. The main reason for this: when increasing the number of points in a network, the number of possible combinations of pairwise connections between all points in this network grows much faster. In 1972, it was found that the TSP is a so-called "NP-hard" problem [Karp, 1972], which means that it is not solvable in polynomial computation time. Following the "exponential time hypothesis" it is even implied that the computation time increases exponentially with growing problem size [Impagliazzo and Paturi, 2001].

The TSP over time became one of the most widely studied combinatorial optimization problems. An overview of applications, formulations, exact and approximate algorithms can be found in [Matai et al., 2010] and [Laporte, 1992]. Many variants and generalizations of it have been formulated to date in order to solve more and more complex transportation problems of modern societies. Most of these problem formulations are considered part of the family of "vehicle routing problems" (VRPs). First formulated in 1959 as "the truck and dispatching problem" [Dantzig and Ramser, 1959], the VRP in general deals with the optimized routing of a fleet of vehicles instead of only one individual. As a generalization of the TSP, the VRP and its variations are NP-hard problems as well.

Taxonomy of the Problem

The number of publications in this research area grew steadily due to the increasing interest from the transportation research community, the economy, and politics, especially since worsening traffic problems hampered the free flow of goods and people's individual mobility. The resulting monetary losses, as well as the negative impact on the quality of life for residents and car drivers alike, triggered many studies and research projects focusing on finding solutions for growing transportation problems. As the VRP is the generalization of most problems concerning fleets of vehicles moving through a defined network, many of these studies included a variation of the VRP in one form or another.

To give an overview of the variety of the VRP, many books and review papers present classifications of the problem and list a selection of representative publications for each of them. One of the first works that classified publications on VRP in this manner is [Bodin and Golden, 1981]. Its taxonomy includes the definition of dimensions to consider when formulating a variant of the VRP, e.g., the type of time constraints, the number of depots, the fleets size and heterogeneity, as well as the types of vehicles, the demand and the underlying network. It also classifies the operational constraints in terms of the kind of ODM service that is offered, the costs and objective function. Additionally, it presents an overview of solution strategies for these kinds of problems, ranging from greedy heuristics to exact procedures. Moreover, a "hierarchy of vehicle scheduling problems" is also introduced, spanning the range of problem varieties from simple VRPs to more complex ones, which include time window constraints and scheduled rides. Most of the publications about classification and taxonomy of the VRP are structured in a similar fashion, some of them with a focus on a single aspect.

[Toth and Vigo, 2002a] lists more characteristics of vehicles and customers, and more optimization objectives to solve the problem. It focuses on the problem definition of VRP variants and their interconnections and classifies basic models for the VRP, including vehicle flow and set-partitioning models. In [Irnich, Toth, et al., 2014], the spectrum of problem formulations is even further explored, adding variations in the types of transportation requests and mixed versions of established problem definitions, including location routing problem, in which the objective is to simultaneously optimize location and routing of the vehicle fleet.

More recent works focus on the thorough exploration of the problem parameter space in which the VRP and all its variations are defined. In [Drexl, 2012], this parameter space is cut into five distinct categories, in which all the variables are sorted in. These categories are

- "requests" – including the parameter "type of time windows",
- "fleet" – with parameters like "type of costs", "type of capacity constraints" and "type of driving speed",
- "route structure" – summing up parameters like "interdependence of routes" and if routes are needed to be closed or open-ended,
- "objectives" – a category that covers the "dimensionality of objective functions" used for optimization, the specific "optimization target" and the "hardness of constraints",
- "scope of planning" – including parameters like the "time horizon" of a problem, as well as its "data availability and accuracy".

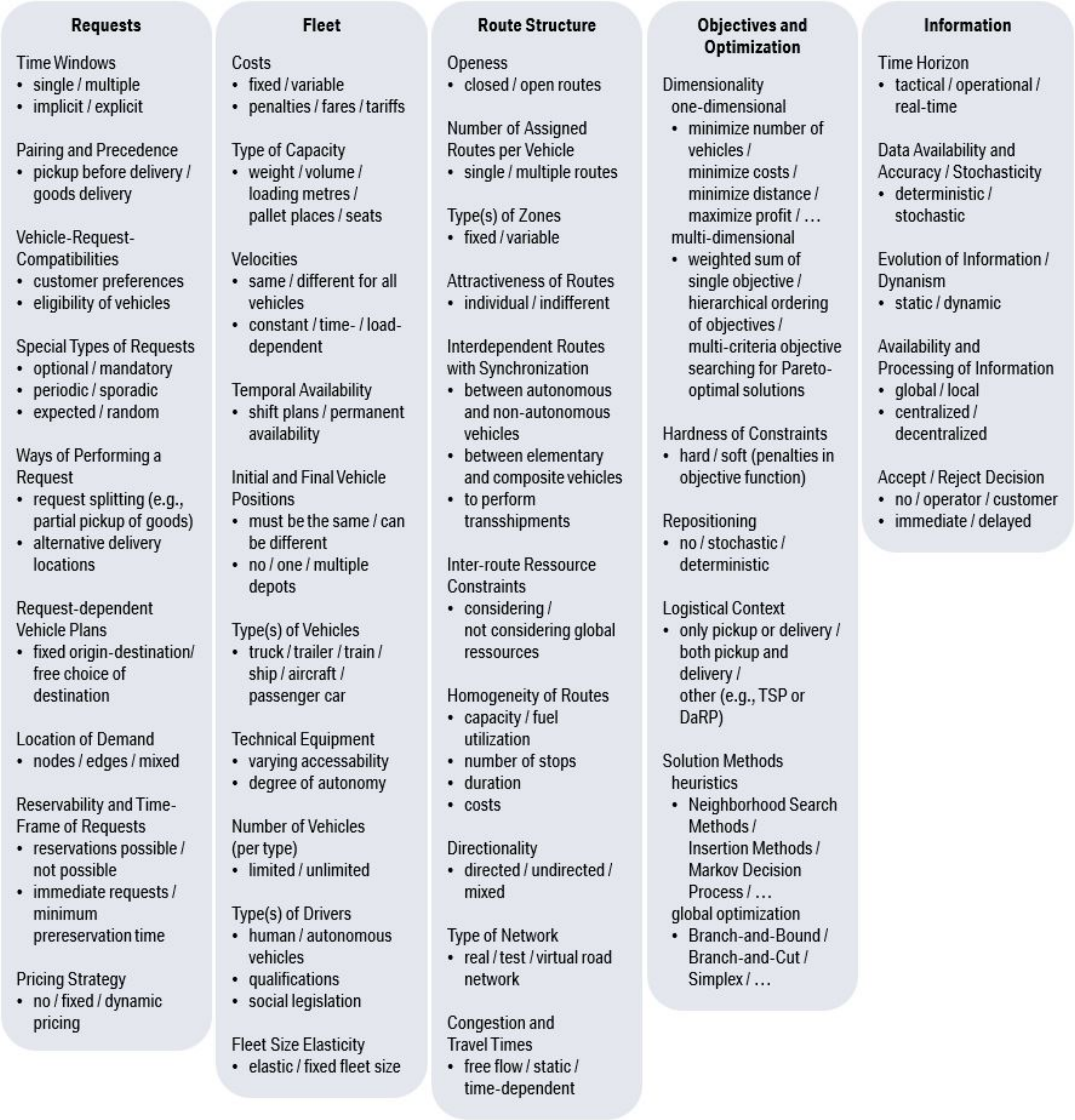
This categorization helps to keep an overview of the numerous dimensions VRPs can vary in and helps to identify problem formulations that are similar to one another. Later publications did not necessarily follow these categories, however. [Psaraftis et al., 2015] defines eleven categories, many of which include similar parameters as in [Drexl, 2012] and are therefore rather split up and rearranged. Additionally, it presented new parameter space dimensions, like the "nature of dynamic elements", including options such as dynamic requests, travel times or vehicle availability, as well as "solution methods", listing a number of heuristic and exact optimization procedures. An even richer taxonomy is presented in [Hyland and Mahmassani, 2017]. On top of the traditional taxonomic categories, it introduces a number of new ones, many of which are closely connected to the variations of the VRP that are used for describing ODM services. The most important ones in this regard are the "fleet size elasticity", the inclusion of "reservations" and the respective time horizons, the "pricing" concept and the "type of repositioning". Figure 2.1 shows a combined taxonomy of parameters used in various formulations of the VRP.

The Family of Vehicle Routing Problems

In order to understand what kind of problems have been formulated in this vast parameter space, in the following a short introduction is given into some of the most important members of the family of VRPs.

One of the earliest and most established variants of the VRP is the capacitated vehicle routing problem (CVRP). In its basic form, the CVRP consists of a single depot and a number of points of demand. CVRPs can be formulated on undirected or directed graphs. The problem statement is to find tours to optimally deliver goods from the depot according to the demand using a homogeneous fleet of vehicles that have the following properties:

- all tours of the vehicles start from and end in the depot,
- all have the same capacity, meaning they all can transport the same maximum amount of goods,
- all are operating at identical costs.



Requests

Time Windows
- single / multiple
- implicit / explicit

Pairing and Precedence
- pickup before delivery / goods delivery

Vehicle-Request-Compatibilities
- customer preferences
- eligibility of vehicles

Special Types of Requests
- optional / mandatory
- periodic / sporadic
- expected / random

Ways of Performing a Request
- request splitting (e.g., partial pickup of goods)
- alternative delivery locations

Request-dependent Vehicle Plans
- fixed origin-destination/ free choice of destination

Location of Demand
- nodes / edges / mixed

Reservability and Time-Frame of Requests
- reservations possible / not possible
- immediate requests / minimum prereservation time

Pricing Strategy
- no / fixed / dynamic pricing

Fleet

Costs
- fixed / variable
- penalties / fares / tariffs

Type of Capacity
- weight / volume / loading metres / pallet places / seats

Velocities
- same / different for all vehicles
- constant / time- / load-dependent

Temporal Availability
- shift plans / permanent availability

Initial and Final Vehicle Positions
- must be the same / can be different
- no / one / multiple depots

Type(s) of Vehicles
- truck / trailer / train / ship / aircraft / passenger car

Technical Equipment
- varying accessability
- degree of autonomy

Number of Vehicles (per type)
- limited / unlimited

Type(s) of Drivers
- human / autonomous vehicles
- qualifications
- social legislation

Fleet Size Elasticity
- elastic / fixed fleet size

Route Structure

Openess
- closed / open routes

Number of Assigned Routes per Vehicle
- single / multiple routes

Type(s) of Zones
- fixed / variable

Attractiveness of Routes
- individual / indifferent

Interdependent Routes with Synchronization
- between autonomous and non-autonomous vehicles
- between elementary and composite vehicles
- to perform transshipments

Inter-route Ressource Constraints
- considering / not considering global ressources

Homogeneity of Routes
- capacity / fuel utilization
- number of stops
- duration
- costs

Directionality
- directed / undirected / mixed

Type of Network
- real / test / virtual road network

Congestion and Travel Times
- free flow / static / time-dependent

Objectives and Optimization

Dimensionality

one-dimensional
- minimize number of vehicles / minimize costs / minimize distance / maximize profit / …

multi-dimensional
- weighted sum of single objective / hierarchical ordering of objectives / multi-criteria objective searching for Pareto-optimal solutions

Hardness of Constraints
- hard / soft (penalties in objective function)

Repositioning
- no / stochastic / deterministic

Logistical Context
- only pickup or delivery / both pickup and delivery / other (e.g., TSP or DaRP)

Solution Methods

heuristics
- Neighborhood Search Methods / Insertion Methods / Markov Decision Process / …

global optimization
- Branch-and-Bound / Branch-and-Cut / Simplex / …

Information

Time Horizon
- tactical / operational / real-time

Data Availability and Accuracy / Stochasticity
- deterministic / stochastic

Evolution of Information / Dynanism
- static / dynamic

Availability and Processing of Information
- global / local
- centralized / decentralized

Accept / Reject Decision
- no / operator / customer
- immediate / delayed

Figure 2.1: Summary of the taxonomy of problem parameters in [Drexl, 2012], [Psaraftis et al., 2015], and [Hyland and Mahmassani, 2017], categorized as in [Drexl, 2012].

An important supplement to the classical VRP is the so-called arc routing problem (ARP), where the general goal is to visit all the edges of the graph instead of all the vertices, as is the case in the classical VRP. This kind of problem is often referred to as the "postman problem", because it is closely associated with the daily business of a postman or a postal carrier trying to cover the streets of his or her area of responsibility effectively. In [Cordeau and Laporte, 2006], the most general version of the ARP is referred to as the Chinese postman problem

(CPP), which is solved by determining a shortest closed path on a graph. The constraint that no edge is allowed to be traversed more than once makes this graph Eularian. The CPP can be defined on both directed and undirected graphs. In both cases optimal solutions can be found in polynomial computation times. If the problem is defined on mixed graphs, the CPP becomes NP-hard. More constrained variants of the ARP include the rural postman problem, in which only a subset of required edges needs to be traversed by one vehicle, or the capacitated ARP (CARP), in which a fleet of identical vehicles with a certain capacity needs to serve a non-negative demand on the edges of the graph. Both of these variants are NP-hard as well. The CARP in particular is an active research area with many subvariations of its own, much like the CVRP. A summary can be found in [Wøhlk, 2008] together with an introduction to common optimization methods and a number of real-world applications. An even more extensive overview is presented in [Corberán and Prins, 2009], in which the whole spectrum of ARPs is covered. Since the problem formulated in this work is part of the family of VRPs, the remainder of this subsection focuses on these variants instead of ARPs'.

Because of its generality, the classical VRP, and the CVRP in particular have mostly academic relevance. These early variants are more focused on static use cases. Nowadays, due to digitalization and the widespread access to mobile internet, transportation problems tend to be more dynamic in their nature. The formulation of the CVRP, however, introduces variables and constraints that are necessary to understand many of the more recently defined problem variants. [Irnich, Toth, et al., 2014] introduces the chapter about the family of VRP for this reason and states four different mixed integer programming (MIP) models that are able to optimally solve the CVRP, at least as long as the number of vertices in the network is small enough. The CVRP is also depicted as the core version of all variants of the VRP in [Toth and Vigo, 2002a], where it is split up in several subcategories, depending on if the graph the problem is defined upon is directed (asymmetrical CVRP) or undirected (symmetrical CVRP) and if the capacity constraint is defined for the vehicles' capacity to load goods (classical CVRP), the length of the routes (distance-constrained VRP [DVRP]) or both (distance-constrained CVRP [DCVRP]).

A step towards applicability is done when considering time intervals in which tasks at certain nodes in the system need to be performed. This broad area of problem formulations is referred to as VRPs with time windows (VRPTW). VRPTWs come in many different variations on their own, depending on properties like what aspects of the problem are associated with time windows: the availability of vehicles, the pickup of goods or the time of arrival at the destination. Another factor is the type of constraint that is imposed by time windows. Are assignments outside of the interval considered feasible but carry a penalty in the evaluation of the quality of the solution (soft time windows) or are they not allowed at all (hard time windows)? An introduction and overview to the variety of VRPTW is given in [Desaulniers et al., 2014] and [Cordeau, Desaulniers, et al., 2002]. In [Tan et al., 2001], several heuristic methods to solve a version of the VRPTW are described, evaluated and compared. A combination of simulated annealing and large neighborhood search is used in [Bent and Van Hentenryck, 2004] to first minimize the number of vehicles considered and then their travel costs, in order to serve a given number of customers.

Besides intra-route constraints, another area in which VRPs can be differentiated is the field of fleet characteristics and the number of depots. A version of the VRP in which more than one depot exists is presented in [Renaud et al., 1996]. This variant is referred to as the multi-depot VRP (MDVRP) and in combination with the constraints of the CVRP it is often more applicable to real-world logistic problems which do not necessarily include one single hub of operation.

In another expansion of the problem, the fleet of vehicles considered is heterogeneous (HFVRP), meaning the vehicles of the fleet vary in capacity, costs per traveled distance or both [Naji-Azimi and Salari, 2013]. This problem formulation allows for the presentation of a number of real-world applications [Irnich, Schneider, et al., 2014]. A subcategory of the HFVRP is the truck-and-trailer routing problem (TTRP) in which (at least) two groups of vehicle types are considered separately: single trucks and trucks with trailers. In [Chao, 2002], a tabu search methodology is used to solve this problem heuristically.

A crucial aspect of VRPs is their respective degree of dynamism (DoD). This measure indicates how dynamic a problem is in respect to the extent of information revealed in real time compared to the overall information [Lund et al., 1996]. In general, VRPs can be characterized as static, if the DoD is close to or equals zero, which is equivalent to the case in which all or most of the information is available at the beginning of the solution finding process, or as dynamic, which generally means that significant shares of information are only revealed after parts of the problem have already been solved. Static VRPs have long been the standard formulation of the problem, because the access to all the information about vehicle positions and requests allows to focus on the formulation of the problem itself rather than the challenges that come with the lack of information and related questions [Berbeglia, Cordeau, Gribkovskaia, et al., 2007; Brandão, 2011].

However, for many of the applications for which the VRP is used, a dynamic formulation of the problem is needed. In [Jaillet and Wagner, 2008], a dynamic (referred to in the publication as "online") version of a TSP is presented and later generalized to multiple vehicles. [Berbeglia, Cordeau, and Laporte, 2010] surveys dynamic pickup-and-delivery problems (PDPs), in which people or objects need to be collected and transported in real time. In-depth investigations into the subject are provided by [Pillac et al., 2013] and [Bektaş et al., 2014].

Another version of the VRP, in which the information is not fully known at the beginning, is the stochastic formulation of the problem. Here, the uncertainty of system conditions is generally described by probability distributions. The goal of stochastic VRPs is to evaluate the impact of uncertainty on the system and the respective solution quality. In order to optimize routes to unknown future demand, in [Bertsimas, 1992], sequences of customers are calculated a priori based on probabilistic assumptions with the goal to minimize the total length of routes. It is found that this approach is a competitive alternative to dynamic formulations of similar problems. A broader review of stochastic VRPs is presented in [Gendreau, Jabali, et al., 2014], in which, similar to the dynamic problem formulation, is stated that the source of uncertainty can be the volume of the demand, the time and location of customer requests, the travel times for vehicles or some other parameter of the system.

One of the broadest categories in which VRPs differ is the type of transportation request that is served. If service vehicles deliver or collect goods from or to the nodes of a network, the problem is part of a subcategory referred to as "delivery and collection". One of the most-common formulations of such a problem is the VRP with simultaneous pickup and delivery (VRPSPD). In its most general form, this formulation describes the problem, in which customers of a service can both send and receive goods at their positions on the graph. One of the first works on the VRPSPD is [Min, 1989], in which a model and solution procedure for a case study of a library distribution system is presented. In [Bianchessi and Righini, 2007], the definition of the problem is to optimally integrate goods distribution and waste collection, when the operations can be performed in any order. For this purpose, a number of heuristics is applied to solve the problem for use cases in which customers may request waste collection, deliveries of goods or both at the same time. A detailed overview of applications and models of the VRPSPD is presented in [Battarra et al., 2014].

Another variant of the VRP that deals with the delivery and collection of goods that are brought to or collected from nodes in a network, is the so-called VRP with backhauls (VRPB). If the handling and rearrangement of goods inside of the vehicles is difficult or not possible at all, e.g., if bulky or fragile materials are transported, all deliveries must be performed first before the pickups of new goods. [Toth and Vigo, 1997] formulates the VRPB as an integer linear programming model and solves it optimally by using a branch-and-bound algorithm. A heuristic approach for the same kind of problem is presented in [Osman and Wassan, 2002], in which the solutions are found to be competitive with the considered benchmarks. Comprehensive overviews of use cases, solution methods and modeling approaches are given in [Ropke and Pisinger, 2006a] and [Irnich, Schneider, et al., 2014].

In contrast to problem formulations of delivering and collecting goods from and to customers at the nodes in a system, "point-to-point transportation" problems describe services that deal with the transportation *of* the customers or treat the transported goods as such. The latter is also referred to as PDP and differs from the VRPSPD and the VRPB in one important aspect, which is that goods are not necessarily directly from or heading to a depot, but are instead transported from one regular node to another. This kind of problem is also known as one-to-one problem, whereas the problems summarized before are characterized as many-to-many problems, in case the transported goods are identical and interchangeable, or as one-to-many-to-one problems, in which there are either certain kinds of goods for certain customers or the load is otherwise identifiable and cannot be interchanged. A more detailed explanation and definition of the PDP and the differences to related problem variants is presented in [Savelsbergh and Sol, 1995]. The most important models and solution approaches for the PDP are summarized in [Cordeau, Laporte, and Ropke, 2007]. In [Berbeglia, Cordeau, and Laporte, 2010], the focus is the dynamic version of the PDP, its formulation and the introduction to some specific solution methods. One of these methods is presented in [Mitrović-Minić et al., 2004], which tries to approach the dynamism of the problem by a heuristic based on a so-called double-horizon. This means, that the optimization objective is split into a short-term and a long-term goal, reflecting that the currently best solution (short-term) is not necessarily suited to be the solution that leads to the optimal outcome over a certain period of time (long-term).

If point-to-point transportation is requested for customers themselves rather than goods, the problem is referred to as dial-a-ride problem (DaRP). This variant of the VRP received much attention in recent years and decades, especially since the emergence of ODM services. The unique feature compared to any other type of VRP is the combination of fleet management with the goal to maximize the profit of the operator running the service and the challenge to provide a high service quality to the customers in terms of minimal waiting times, a high percentage of requests served and other comfort indicators, like quick response times and information transparency. The DaRP was first investigated 1980 in [Psaraftis, 1980] for a single vehicle. It already included the aspect of customer dissatisfaction and was formulated in two separate variants, as static and dynamic problem. The static version of the DaRP is mainly relevant for academic purposes and as part of the proof of concept for new problem variants. In [Cordeau and Laporte, 2003], a tabu search metaheuristic is applied to solve a rather simple version of the static DaRP in a way that can easily be adapted to other, more specific problem formulations, including heterogeneous fleets, multiple depots and more sophisticated objective functions. [S. Ho et al., 2018] proposes an upgraded version of the tabu search in order to further accelerate the search procedure by quicker identifying an initial feasible solution and faster convergence to the optimal solution. Dynamic versions of the DaRP are very common to describe real-world applications of ODM services of all kinds. Since the problem formulations considered in this work share the characteristics of a dynamic DaRP, Section 2.3 is dedicated to the research background of this subcategory of problems. An overview of variants, models and solution approaches of the DaRP is presented in [Cordeau and Laporte, 2003], a literature review on research developments in the area of this problem can be found in [S. C. Ho, Szeto, et al., 2018]. A really comprehensive introduction to the topic is given in [Doerner and Salazar-González, 2014], where the DaRP is also put in perspective to other VRPs.

To conclude this introduction to the family of VRPs, it should be mentioned that this is by no means a complete representation of the broad scope of variants of the VRP. There are countless specifications, variations, problems and subproblems to all kinds of versions of the challenge to optimize a fleet of vehicles according to certain objectives and constraints. In order to give an impression of the width of the research area, two more problem variants may be mentioned. In the vehicle scheduling problem (VSP) the goal is to find optimal routes and schedules for regular transportation services, offered by public transportation vehicles, like busses or trains. The number of variants of this problem formulation alone is immense, considering the variability of use cases in which public transportation is found. An overview of VSPs is presented in [Desrosiers, Dumas, et al., 1995].

The last example of a member of the family of VRPs is the periodic VRP (PVRP) in which customer requests are given as visiting patterns within the planning horizon of the service provider. If customers require a service twice per week, without the need of specifying which days exactly, feasible visiting patterns might be (1,4) and (2,5) representing visits at Monday and Thursday or Tuesday and Friday. The PVRP needs to solve the subproblems of selecting feasible patterns, assignments of vehicles to these patterns and routing the selected vehicles optimally. This complex problem formulation is needed for use cases in waste collection, product distribution and health care. An introduction to the topic is given in [Irnich, Schneider, et al., 2014].

The range of variations of the VRP grew constantly in recent years and decades and still does. An overview of the presented, non-exhaustive selection of VRP variants is shown in Figure 2.2.

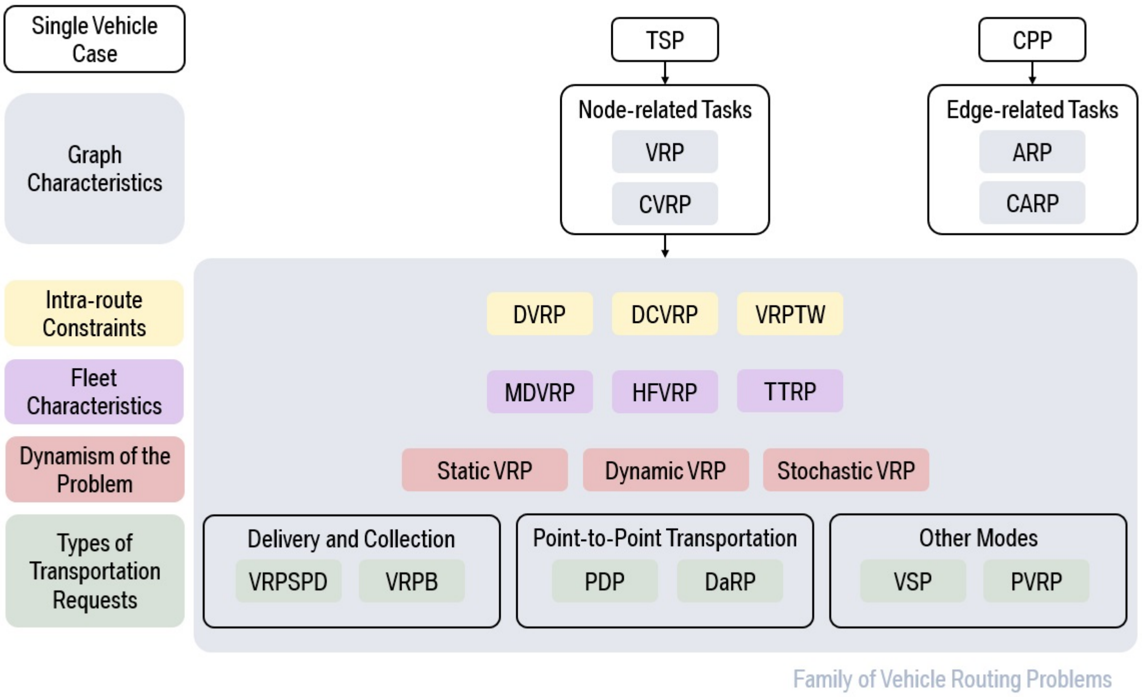


Figure 2.2: A non-exhaustive overview of varieties of the vehicle routing problem and related problem classes, ordered by the properties and abbreviated as found in the text.

2.2 Optimization Techniques

After giving an overview of the problem formulations of the VRP, the upcoming section covers the methods that are used in the literature to solve these kinds of problems in more detail. The order in which these methods are presented is based on their respective approach to the problem. There are two fundamental metrics in which the performance of optimization problem solving techniques are measured:

- *Optimality* is the system performance in terms of the quality of the found solution. In static problem instances, optimality is normally measured by the objective function value relative to the global optimum and the ability to identify this optimal value. In dynamic problem formulations, it is often measured in terms of key performance indicators (KPIs) like “distance driven by the fleet”, “customers served”, or “profit generated”, also in comparison to solutions that provide the best respective KPI.

- The *computation time* of an optimization technique represents the resources it takes to find the solution to a given problem. The most critical resource is often time, assuming the hardware the problem is solved on is generally able to calculate the solution. Therefore, the computation resources are normally measured as a value of time, spanning from fragments of a second to hours or more, depending on the complexity and size of the problem.

Most methods known can be categorized into one of three groups of approaches: “heuristics”, “metaheuristics”, and “exact optimization”. Each of these concepts focuses on different

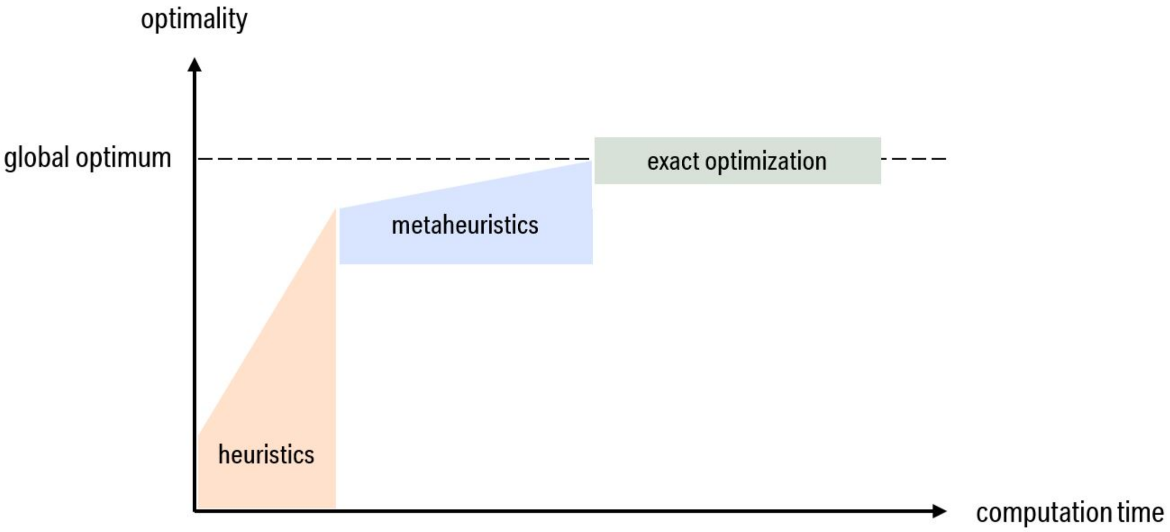


Figure 2.3: Schema of optimization approaches in dimensions of optimality and computation time.

aspects. Heuristics are methods based on simple rules to quickly find a solution to a given problem. The kinds of rules differ depending on the respective heuristic. The advantage of being able to potentially solve very hard, complex and large optimization problems in short computation times is opposed by the fact that the quality of the solution found is uncertain, meaning the solver is not able to give an estimation of how far away the solution is from the global optimum. Metaheuristics are more sophisticated and generalized procedures that aim to find solutions to optimization problems in computation times comparable to heuristics, while providing solution qualities which are closer to the global optimum. The means by which metaheuristics work are manifold and the scope of accomplished computation times and optimality measures is wide. The group of exact optimization techniques solves problems globally optimal. The approaches of the members of this group vary, but most of them include an algorithmic element that can guarantee that certain parts of the solution space are not worth of further exploration as the solution currently found is ensured to be better than the best solution possible out of this part of the solution space.

Figure 2.3 is a schematic representation of the three groups of optimization approaches in terms of their respective strength and weakness between optimality and computation time.

Heuristic Approaches

Some methods to solve optimization problems rely on simple sets of rules to search the solution space. This approach is referred to as "heuristic" and is very common for problems that are either too hard to solve globally optimal because of their complexity or size, or that rely on speed rather than optimality. In literature, there are many examples for heuristic methods and their applications in research and real-world use cases.

One of the most intuitive procedures to assign requests that are added to a system in a certain order is the first-come-first-served heuristic (FCFS). Such an ordering is needed dynamic formulations of the problem. The only criterion that is considered when making the decision which vehicle is assigned to a new request, is the availability of vehicles at the time of the request. If more than one vehicle is able to serve the customer, the closest one is assigned. If no vehicle is available, the request is put on a waiting list, until the next vehicle is available and assigned to the waiting request. This heuristic's focus is to minimize the complexity of the problem solving procedure in order to solve very dynamic or large problems, or to compare such simple approaches to more sophisticated ones. The latter is done in [Bertsimas and Ryzin, 1991], where a model for the stochastic and dynamic VRP is introduced and a number of policies is presented to find solutions for varying use case scenarios. It is found that the FCFS is competitive compared to other models in scenarios with light traffic conditions, but struggles to perform on similar level in scenarios with heavy traffic.

The heuristic concept referred to as *nearest neighbor policy* (NNP) can be described as the procedure of searching for the vehicle that is able to pick up a new request the earliest or the vehicle that is able to perform the pickup while producing the lowest cost according to an objective function. The set of vehicles considered can be restricted, e.g., to those that are idle, meaning vehicles that do not currently have planned assignments. In [Lee et al., 2004], the NNP is used to manage a fleet of taxis in a case study of Singapore. The search algorithm considers assignments as fixed, as soon as they are made. This implies that customers can be informed about their respective assignment immediately. On the other hand, the fact that requests cannot be reassigned after the initial match with a vehicle can lead to suboptimal solutions later. This shortcoming is addressed in [Sheridan et al., 2013], where a dynamic version of the NNP is presented. If it is beneficial for the overall system, vehicles on their way to a pickup location can be reassigned to new requests. Depending on the constraints used in that version of the NNP, responses to customer requests can only be sent after a certain period of time, or customers that already received an estimated pickup time can even be rejected later, which translates to a very bad user experience. A comparison between the NNP and the FCFS is presented in [Maciejewski and Bischoff, 2015] for a case study of Berlin, in which the results further indicate that the NNP is superior to the FCFS in scenarios with high demand.

Both FCFS and NNP are part of a group of methods referred to as "insertion heuristic". The general idea such heuristics are based on, is to minimize the additional costs according to the respective objective function whenever a new request is added to the system by inserting this new request into the task queue of a vehicle without violating the feasibility to the customers already assigned to the respective vehicle. [Jaw et al., 1986] demonstrates how to use this heuristic for the DaRP. After first identifying all feasible insertions of a new request for all vehicles of the fleet, the algorithm aims to find the optimal assignment of this request by comparing the costs that would be added by each of the configurations. In [Madsen et al., 1995], this approach is applied to the problem of transporting elderly and disabled people with specialized vehicles, adding a number of multi-dimensional capacity constraints to the problem, making it considerably more complex. In a recent paper, an insertion heuristic is used to assign customers of a ride-pooling service to the vehicles of an ODM fleet in a case study of Munich [Dandl, Engelhardt, et al., 2021].

Insertion heuristics are also often used as part of a multi-step optimization procedure. In [Gendreau, Hertz, et al., 1992], the TSP is solved using a combination of two methods.

First, a request is inserted into an oriented route between two requests that have already been assigned to the tour, taking into account all combinations of two requests, not only those which are planned to take place consecutively. After reconnecting the requests, it is checked if all requests are part of the tour. If not, the process is repeated inserting the new request between another combination of requests. In a second step, a route is further optimized by removing a request from it and trying to insert the request in another position of the tour. [Mitrović-Minić et al., 2004] applies a double-horizon based approach to solve the PDP with time windows and splits the problem into two subproblems, the routing and the scheduling problem. The routing problem is solved by inserting new requests that have been added to the system during a certain period of time and reinserting those that were not picked up yet. After a feasible solution is found, a tabu search is performed during the next optimization period in order to optimize the assignments.

Because of the vast range of problem formulations, use cases and corresponding heuristic approaches, there are plenty of publications presenting reviews of the most important developments in this area of research. They often compare the performances of classical heuristics with those of metaheuristics ([Cordeau, Gendreau, Laporte, et al., 2002], [Laporte et al., 2014], [Prodhon and Prins, 2016]) and exact optimization techniques ([Cordeau and Laporte, 2006]).

Metaheuristics

Compared to classical heuristics, metaheuristics are more sophisticated methods, which aim to find close-to-optimal solutions for hard optimization problems in computation times comparable to heuristics by intelligent search procedures. Generally, there are two distinct parts of the search procedure when it comes to metaheuristics: diversification and intensification. Diversification refers to the ability to identify promising areas of the solution space, in which the solution qualities are overall better than average and chances to find the global optimum are higher. Searching promising areas of the solution space for the very best solutions and being able to identify the globally optimal one is referred to as intensification.

The scope of metaheuristics for variants of the VRP is very wide, spanning from methods that use smart combinations of classical heuristics, to techniques that are inspired by nature's processes of optimization, to applications that make use of artificial intelligence. This wide field of approaches can be divided into "local search" algorithms, "population-based" algorithms, and "machine learning" algorithms.

Local search algorithms typically start with an initial solution s_i, then build a neighborhood of solutions that are similar to s_i according to specific similarity rules, find the best solution s_{i+1} in this neighborhood in terms of the objective function that satisfies the given constraints and repeat that procedure iteratively until a termination criterion is met. Depending on the neighborhood-defining parameters, some local search metaheuristics tend to be prone to cycling, which means that already found solution are considered as "new" later in the search procedure, leading to iteration loops that reproduce the same solutions over and over again. Therefore, the most successful of the local search algorithms use techniques to circumvent this weakness in one way or another.

In [Prosser and Shaw, 1997], four simple heuristics are used to define the neighborhood of solutions. In order to find one neighbor in such a neighborhood, each of these heuristics manipulates the current solution by changing one or two of the routes that are part of this solution in a specific way. Such a change is referred to as "local move" and is either considered to be "intra-route" (if the change affects only one route, e.g., if the order of two requests that are scheduled to be picked up by the same vehicle is reversed) or "inter-route" (if the move happens between two routes, e.g., if two requests swap the vehicles, they are assigned to). Local moves are performed in all local search metaheuristics, however, the number, types, and complexity of moves vary a lot. In [Prosser and Shaw, 1997], the neighborhood is searched following a steepest descent approach, which means the neighbor which offers the best solution quality is chosen and the process is repeated until no neighbor is found that offers a better solution quality than the last one found. This approach tends to terminate very quickly, because it stops the search at the first local optimum encountered. This solution, however, can be arbitrarily far away from the global optimum.

The concept of large neighborhood search (LNS) metaheuristics is to build neighborhoods that are large enough to assume that their local optimum is close to the global optimum of the problem. It was first introduced in [Shaw, 1998] using constraint programming techniques, which aim to solve optimization problems along the restrictions that are set by the constraints rather than trying to explore the solution space more freely. Since the generation and search of large neighborhoods can be very time consuming, such combinations with methods that filter the search space are found to be very promising optimization techniques to solve VRPs that are not too dynamic in their nature. Another unifying feature of LNS procedures is the use of heuristics that purposely destroy and rebuild the current solution in order to find better combinations of its elements. In [Ropke and Pisinger, 2006b] and [Syed, Kaltenhäuser, et al., 2019], three different removal heuristics and two insertion heuristics are used to generate very large and diverse neighborhoods. Furthermore, the heuristics are dynamically weighted, depending on how often they produced a better solution in recent iterations of the search. The respective weighs decide which of the respective removal and insertion heuristics is used in the next iteration of the search.

Another metaheuristic concept is "simulated annealing". The underlying idea for this method originates from the process of the physical annealing with solids, in which crystalline substances are heated and then allowed to slowly cool down until they achieve a lattice configuration that is stronger than the original one. If the cooling process is sufficiently slow, the lattice energy level is minimized and the resulting crystal is free of structural defects. This behavior observed in thermodynamics is translated to discrete optimization problems by implementing a temperature parameter, which decreases steadily over the course of an optimization process. The "temperature" of the search is directly affecting the probability of accepting non-improving solutions out of a neighborhood. This concept aims to ensure a diverse search at the beginning, when the temperature and the probability of escaping local optima is high, while being able to intensify the search in promising areas of the solution space towards the end, when temperatures are low and only improving solutions are accepted. A thorough introduction to the methodology and the history of applications is presented in [Nikolaev and Jacobson, 2010]. In [Baugh et al., 1998], a multi-objective DaRP is solved with simulated annealing, showing that simulated annealing performs well in terms of optimality and computation time for this kind of problem. [Osman, 1993] combines simulated annealing with a steepest descent heuristic for the initial solution and tests the algorithm on 26 instances of static, symmetric VRPs, in many of which the optimal solution was found considerably faster compared to other methods.

Like simulated annealing metaheuristics, "tabu search" methods avoid cycling and premature terminations of searches by allowing worse or even infeasible solutions to be considered as next step in the iterative search under certain conditions. These methods depend on lists of solutions that already have been encountered during the search, not only to compare new solutions with the best one found so far, but also to declare recently found solutions as "tabu", which avoids cycling. Besides these "tabu" solutions, the iterative search is allowed to overcome local optima by choosing neighboring solutions that are worse than the current solution. Implementations of tabu search methods were found to be very effective for many variants of the VRP, both for static and dynamic problem formulations. Static versions of the HFVRP and the DaRP are solved with tabu search metaheuristics in [Brandão, 2011], [Li et al., 2012], [Cordeau and Laporte, 2003], [Pandi et al., 2018], and [S. Ho et al., 2018]. Both the MDVRP and the PVRP are considered in [Cordeau, Gendreau, and Laporte, 1997]. [Toth and Vigo, 2003] presents a granular tabu search approach for the DVRP and DCVRP in which the neighborhoods are kept very small in order to further accelerate the search procedure. Use cases of tabu search metaheuristics for dynamic variants of the VRP are presented in [Gendreau, Guertin, et al., 1999] and [Attanasio et al., 2004], where the algorithm was implemented on a parallel computing platforms to further increase the computation speed. The dynamic DaRP is solved by means of a hybrid tabu search and constraint programming algorithm in [Berbeglia, Cordeau, and Laporte, 2012]. Even more applications of tabu search metaheuristics and overviews of various implementations can be found in [Bräysy and Gendreau, 2002], [Cordeau and Laporte, 2005], [Gendreau and Potvin, 2010] and [Glover, 2013].

The second group of metaheuristics – the population-based algorithms – is characterized by their shared inspiration from natural concepts. The range of implementations of these concepts is very wide, and so is their applicability to various forms of the VRP. Probably the most studied population-based approaches are "genetic algorithms" and the very closely related "evolutionary algorithms". As the names of these metaheuristics suggest, the inspiration for both of them is the concept of evolution in nature, where information is transmitted via genes from one generation to the next. As in nature, populations (of solutions) can mutate whenever offspring (new solutions) is produced, leading to the eventual displacement of one part of the population (old, bad solutions) by another (new, good solutions). The first formulation, the theoretical foundations and first applications of a genetic algorithm are presented in [Holland, 1975]. In [Prins, 2004], it is first implemented for a version of the VRP and shows the potential to effectively find optimal solutions for static problem instances. The static DaRP is solved with a genetic algorithm in [Jorgensen et al., 2007], a dynamic version of the DaRP for the use case of ride pooling is considered in [Herbawi and Weber, 2012].

Two methods that can be seen as improvements of procedures like genetic algorithms are "scatter search" and "path relinking". The former addresses the fact that genetic algorithms tend to produce solutions of poor quality at the step of so-called "cross-over operations", when old solutions are randomly combined to produce an offspring solution. This potentially detrimental element of the search procedure is avoided by the definition of a reference set of solutions, which is a subset of the overall population that includes the best and most diversified solutions. This reference set is then iteratively combined and improved by means of local search procedures. An implementation of this metaheuristic is applied on the VRPSPD in [T. Zhang et al., 2012], where its performance is compared to a genetic algorithm and found to be more efficient, especially for large problem instances.

Path relinking, on the other hand, produces new solutions by gradually transforming one solution into another, preferably connecting two solutions that are fairly far apart in the solution space, and thereby exploring it effectively. This technique is often used to improve the diversification and intensification of other metaheuristics, rather than as a stand-alone optimization approach. In [S. C. Ho and Gendreau, 2006], this technique is combined with a tabu search metaheuristic, which is found to improve the performance in terms of both optimality and computation time. [Resende and Ribeiro, 2010] presents a detailed introduction to both scatter search and path relinking techniques, together with applications and common use cases.

Another concept to solve discrete optimization problems that is directly inspired by nature is the so-called "ant colony" algorithm, which rely on information transmission based on pheromone deposits. Starting from an initial solution, the search proceeds by choosing solutions that are good according to the respective objective function. Subsequent iterations follow that trail of former found solutions because of the pheromones that are placed there, thereby increasing the pheromone deposit even further. Hence, the algorithm focuses on promising areas of the solution space, intensifying the search where it needs to and moves on from explored regions, whenever more promising areas are discovered. This approach is found to be very effective for the minimization of fleet sizes in the static DaRP in [Tripathy et al., 2017]. An overview of implementations and applications is presented in [Dorigo and Stützle, 2010].

The third and final group of metaheuristics is summarized under the term "machine learning". Even though, technically, these concepts are inspired by nature as well, namely by the processes that happen in brains, they are distinct enough from any other metaheuristic approach to be considered as a group on their own.

Out of all approaches presented, the field of machine learning algorithms is probably the one getting the most attention in recent years. The potential of its problem solving abilities is huge, for all kinds of VRPs and optimization problems overall. With the increasing research focus, the range of variants grows continuously. A few of the most promising ones are presented in the following.

For purposes of categorization of machine learning approaches in context of the family of VRPs, this work follows the structure from [Bai, Chen, et al., 2021], which provides a review of recent literature on machine-learning-based optimization approaches for the VRP and its variants. The first category considered decomposes large problem instances into subproblems, thereby cutting the solution space into smaller parts that are faster to explore. In [Cömert et al., 2017], this is achieved by first assigning customers to vehicles based on the solutions found with three different machine learning approaches: k-means, k-metoids, and density-based spatial clustering of applications with noise (DBSCAN). Both k-means and k-metoids build clusters of nodes by learning to minimize the distance between points that are labeled as part of a group and the respective center of this group. In the case of k-metoids, this center needs to be a node itself, which is not the case for k-means algorithms. DBSCAN is based on a nearest neighbor logic that clusters densely packed nodes, while isolating outliers. Out of these three approaches, the best solution found is then further improved with a linear problem solver. This two-stage method is shown to be very effective for the VRPTW. Another decomposing strategy is applied in [Morabit et al., 2020], where a "graph neural network" is used to select the most promising subset of columns generated in each iteration of a column generation process. The concept of neural networks is based on the idea of richly interconnected nodes to transmit information. Each link between these nodes is dynamically weighted, depending on the "experience" of the algorithm with solutions generated using this link. The approach is found to improve the column generation process by up to 30 % in terms of computation time.

The next category of machine learning approaches improves (meta-)heuristic procedures by guiding the search process more efficiently. [Bai, Burke, et al., 2007] uses a hyper-heuristic method to solve the VRPTW and selects the heuristic to use at each step of optimization by applying "reinforcement learning". The idea of this concept is to interpret the consequences of a decision made by an agent (in this case the selection of a heuristic) in the system into a reward and weigh future decisions accordingly, depending on the respective resulting state of the system. Instead of reinforcement learning, [Syed, Gaponova, et al., 2019] uses a neural network to find the best parameters for a LNS without manual tuning.

The third and final use case for machine learning techniques in the optimization of VRP variants presented here is the construction of (initial) solutions from scratch. The CVRP is solved using reinforcement learning in [Nazari et al., 2018], where this technique is found to outperform classical heuristics in small to medium-sized problem instances with comparable computation times, if the training time of the learning model is not considered. "Deep reinforcement learning" is used to solve a complex formulation of a stochastic and dynamic CVRP that also includes pickup-and-delivery constraints and time windows. In addition to the concept of reinforcement learning, deep learning procedures allow the input data for the problem to be less structured by the utilization of neural networks to transform it before the optimization process starts. This method outperforms classical heuristics in terms of optimality for small problem instances.

Before the conclusion of the review of metaheuristics for the VRP it should be mentioned that in addition to the pure implementations of individual metaheuristics there is a whole field of research about metaheuristic hybrids. As explained in more detail in [Raidl et al., 2010] and [Talbi, 2016], methods from various categories can be combined to make use of the respective strengths of the approaches in order to improve the diversification or intensification of the search, and hence the performance of the algorithm in terms of optimality and computation time. Overviews of metaheuristics, both pure and hybrid, can be found for all kinds of VRP formulations, e.g., [Gendreau, Laporte, et al., 2002] focuses on the CVRP, [Tan et al., 2001] on the VRPTW, and [D'Souza et al., 2012] on the PDP.

Exact Optimization

In contrast to both heuristic and metaheuristic approaches, exact optimization algorithms are designed to be able to find solutions to discrete optimization problems that are globally optimal according to the respective objective function. Another defining feature of these techniques is their ability to find bounds in which the optimal solution value is guaranteed to exist during the search procedure. Hence, they can assess the optimality of the current solution relative to the upper or lower bound of the problem, depending on if the objective function value is to be maximized or minimized, and stop the search procedure when the respective bound is found. Exact optimization methods are therefore often used as benchmarks to evaluate other optimization techniques for all kinds of problem formulations.

The downside of this guaranteed global optimality is the relatively low speed in terms of computation time compared to heuristic procedures. In many problem formulations, these approaches rely on thorough searches of large parts of the solution space in order to find the global optimum. In the family of VRPs, exact optimization algorithms are therefore often used for static problem formulations, which are not too large or multi-dimensional in terms of problem size and complexity. Exceptions to this rely on formulations that include smart cuts into subproblems, strict constraints or other methods that effectively reduce the size of the solution space.

Most exact optimization concepts rely on some kind of solution space reduction. One of the first successfully implemented optimization methods for the TSP was the so-called "branch-and-bound" algorithm in [Little et al., 1963]. The basic idea of this approach is to explore the solution space by building a so-called decision tree. The "branches" of this tree represent values of decision variables, each of which indicates whether or not a specific element (e.g., a vehicle-request pair) is part of a solution. In order to avoid searching large parts of the solution space, branch-and-bound algorithms determine upper and lower bounds during the search by means of relaxation. This concept estimates the lower bound (in the case of a minimization problem, without loss of generality) by relaxing certain constraints of the problem and finding the global optimum for this relaxed problem formulation, which is designed to be quickly solvable. The implementations of branch-and-bound algorithms most often differ in the way the relaxation is executed. In [Little et al., 1963], the number of available vehicles is relaxed, which means the lower bound of the problem to minimize the distance driven by a single vehicle, while visiting all nodes in a network is found by calculating the minimal distance assuming an arbitrary number of vehicles. Later publications focus on improvements of the relaxation method in branch-and-bound algorithms, for example relaxing the constraint of integer decision variables, hence reformulating the integer programming problem to a linear problem. Overviews of these methods are presented in [Toth and Vigo, 2002b] and [Semet et al., 2014].

If the number of constraints of a VRP (or an integer programming problem in general) is too large or the relaxation is found to be not strong enough, the problem often cannot be solved by branch-and-bound algorithms in a reasonable amount of time. One concept that aims to fill this gap is referred to as "branch-and-cut" method, which first solves the linear relaxation of any formulation of the VRP. If the solution includes non-integer decision variables, a "cutting-plane" algorithm is applied to find additional linear constraints that are satisfied by the integer parts of the solution but not the fractional ones. Such constraints are called cuts, and solving the resulting two subproblems with the bounds defined by the solutions of the relaxed problem follows the branch-and-bound method. This procedure is repeated iteratively until an optimal solution to the original formulation of the VRP (which is an integer programming problem) is found. Deeper introductions to branch-and-cut algorithms as well as a number of applications in VRP solvers are presented in [Naddef and Rinaldi, 2002] and [Semet et al., 2014].

Another approach to solve large-scale formulations of the VRP is to not consider each link of the network the problem is defined upon to be part of the solution in the form of a decision variable that is either 1 (in case the link is part of the solution) or 0 (if not). In [Balinski and Quandt, 1964], the CVRP is served instead by treating whole routes (lists of nodes to cover, referred to as "columns") of vehicles as variables that constitute a solution. Solutions are only considered feasible, however, if each node in the network is visited exactly once. This kind of problem formulation is called "set partitioning". The main weakness of this concept is its scaling behavior: the number of columns grows exponentially with the number of customers. The concept of "column generation" aims to avoid that. It starts with solving the linear relaxation of the original problem with only a subset of feasible routes. Out of all the routes not included, the optimal one is derived according to an objective function that aims to find variables that improve the original objective function value. This subproblem is often referred to as "pricing problem". If the pricing problem is not able to find a variable that further improves the original objective function, the solution is optimal. Column generation is found to be very effective for the exact optimization of the VRPTW [Desrosiers, Soumis, et al., 1984] and the HFVRP [Taillard, 1999].

Combinations of branch-and-cut algorithms and column generation methods are currently found to be among the most effective exact optimization techniques to solve a variety of VRP formulations. Such hybrids are termed "branch-and-cut-and-price" algorithms. [Fukasawa et al., 2004] presents an implementation that solves the CVRP optimally with 135 customers. In [Pecin et al., 2016], the approach was further improved to solve similar problems with up to 360 customers. A more complex problem formulation is considered in [Ceselli et al., 2021], where electric vehicles are routed optimally between charging stations that use varying charging technologies. An overview of more applications of branch-and-cut-and-price algorithms is presented in [Costa et al., 2019].

Depending on the problem formulation, the number of constraints and the complexity of the model, optimal solutions to variants of the VRP can also be found using simpler procedures. Commercial solvers, such as CPLEX ([IBM ILOG CPLEX, 2017]) and Gurobi ([Gurobi Optimization, 2021]), solve generic integer programming and linear problems using the "simplex" method. This method makes use of the fact that the feasible region of the solution space of such problems can be represented as a polytope. The simplex algorithm is designed to move along the edges of such a polytope, representing an improvement of the objective function, until another vertex is reached. If an edge is found to be infinitely long, the problem is not solvable, otherwise the next vertex represents a better solution than the one before. If no further improvement can be obtained, the optimal solution is found. This procedure is found to be very effective for a variety of different problems in the family of VRPs, e.g., for the VRPTW in [Baldacci et al., 2011] or the DaRP in [Hyland and Mahmassani, 2018] and [R. Zhang, Rossi, et al., 2016].

Many real-world applications of the VRP are intrinsically dynamic and need to be solved quickly. If the solution space of such problems is too large, exact optimization algorithms alone may take too long to find feasible solutions for the whole problem. However, many dynamic problems can be divided into smaller subproblems that are constrained enough for

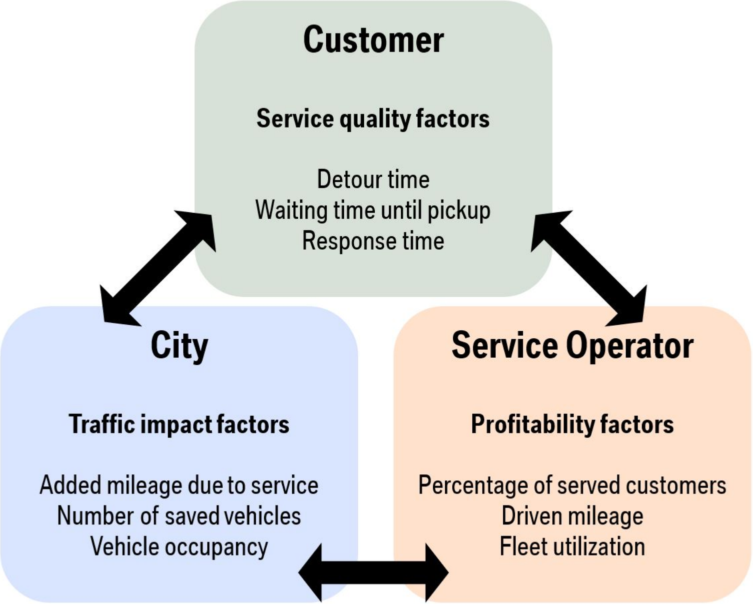


Figure 2.4: The three stakeholders of ODM services and a selection of performance indicators that need to be optimized from their perspectives.

such solvers to find globally optimal solutions. Aggregating the solutions of these subproblems does not add up to the global optimum of the overall problem in general. Nonetheless, this method produces good results for dynamic problem formulations, especially in combination with heuristic procedures. Examples can be found in [Berbeglia, Cordeau, and Laporte, 2012], [L. Zhang et al., 2017] and [Erdmann, Dandl, and Bogenberger, 2021], all solving dynamic DaRPs, which is the standard problem formulation used in modern ODM services, as discussed in this work.

2.3 On-Demand Mobility Services and Customer Models

The final section of this research background focuses on presenting the applications and use cases of the problem formulations and optimization methods introduced in the other sections, specifically in ODM services. Such services allow customers to be transported comfortably within a defined business area. In order to provide a high-quality user experience, minimize the negative impact on (or even improve) traffic in the business area, and to be profitable at the same time, ODM operators need to make sure their fleets are managed optimally. The three key stakeholders of ODM services are shown in Figure 2.4, together with important performance indicators from their respective point of view. The resulting problem formulation translates to a version of the dynamic DaRP.

Most ODM services can be categorized in “ride hailing” or “ride pooling” use cases. Ride hailing services are characterized by the fact that customers are served individually, which means there are no rides shared with other customers and therefore no detours between a pickup location and the respective destination. Such services are very similar to classical taxi services, one difference being the lower fares. One of many reasons for these lower fares is the central optimization of the ODM fleet in contrast to the more independent taxi drivers. Ride hailing can imply a higher service quality compared to ride pooling due to the direct and individual transportation, but often lacks a positive traffic impact, because in order to pick up the next customer, an ODM vehicle needs to drive there emptily from its respective location. Since ODM services are most profitable in regions with dense demand, they are mainly offered in urban areas, which tend to suffer from traffic problems anyway. Therefore, ride hailing services often are opposed not only by taxi drivers, who fear the competition, but also city officials, who do not want to have additional vehicles on the streets, which on top drive around emptily a considerable amount of time. In [Henao and Marshall, 2018], the average ride hailing vehicle is found to drive around emptily 40.8 % to 83.5 % of the time.

Ride pooling services, on the other hand, offer rides that can be shared between customers, potentially reducing the fares for each of them. When sharing a ride, service users need to be willing to make a detour in order to pick up or drop off another user of the service, which might delay their own arrival at the destination. Together with the lack of privacy and possibly space, pooled rides are often considered to be less comfortable than rides that are taken alone. Due to the high occupancy of ride pooling vehicles, this kind of service has the potential to effectively decrease the traffic in its business areas, even though not every ride pooling trip is actually a shared ride. Recent studies found that most of the customers using ride pooling services are not substituting trips they would normally make with their own car, though. [Schaller Consulting, 2018b] presents a meta-study of the mode replacement due to the availability of ODM services in several US-metropolitan areas and found that even if half of all rides made with a ride pooling service would be shared (which is a rate that is currently not met by any service provider), such services would increase the miles driven in the respective business area by around 120 %, because most of the users switch from modes other than their own car.

Another reason for empty mileage due to ODM services is the process of repositioning parts of the fleet in periods of low demand in order to be able to serve more customers when the demand is higher. This field of managing ODM fleets is not the focus of this work. However, since an effective repositioning algorithm is crucial for the performance of any ODM service, a short review of the research done in this area is presented.

Ride Hailing Services

The variety of studies about ride hailing services is broad, since this transportation mode is closely connected to taxi services, which are part of urban transportation systems for the better part of the last century. The central optimization of assignments of vehicles of a unified fleet to service customers is what distinguishes the ODM ride hailing use case considered in this work from the classic taxi service. Established service providers like Uber, Lyft or DiDi offer such services since around ten years now [Uber, 2021], [CNN, 2021], [Chuxing, 2021]. The concept often includes non-professional drivers who transport customers using their private vehicles. The connection between customer and driver is provided via the respective company's mobile application. Even though labor conditions and payments are reported to be poor [Kollewe and The Guardian, 2019], the driver is by far the largest cost factor for ODM service providers. Therefore, many research studies assume automated ODM fleets, which also allows the use of less complex simulation models because of fewer constraints and model parameters, like individual drivers' working hours, the freedom of choice for each driver to reject customers or human errors, like using the wrong route to a destination.

A dispatch algorithm for DiDi is presented in [Xu et al., 2018], in which a reinforcement learning procedure is used to optimize both the short-term assignments of vehicles to users as well as the long-term service performance. [R. Zhang, Rossi, et al., 2016] proposes a model predictive control approach, which includes the repositioning problem in the general objective function and allows the integration of additional constraints, like the charging of electric vehicles. Another study that combines the optimization of assignments in a ride hailing service with the repositioning problem is presented in [Hyland, Dandl, et al., 2020], indicating that the service performance depends heavily on the quality of the repositioning algorithm. In [Nair et al., 2020], the relative amount of empty mileage produced by a ride hailing service in a case study of Austin is investigated. Empty mileage is found to constitute to at least 36 % of all miles traveled by the ODM fleet, mainly in suburban and rural parts of the business area.

A very commonly used data set for simulations of ODM services in general is the open-source Manhattan taxi data set [NYC Taxi & Limousine Commision, 2021]. This data set provides rich data about the entirety of customer requests for yellow cabs in the business area of Manhattan in New York City, including exact time stamps, ride durations and pickup as well as drop-off locations, which are provided as zones rather than exact coordinates since July 2016. In [Syed, Kaltenhäuser, et al., 2019], ride hailing services with varying batching times are tested in an asynchronous simulation framework using 5 %, 10 % and 20 % of this data set. The batching time is the period of time the operator of an ODM service waits for new requests before globally optimizing the assignments. As this study is conducted in asynchronous simulations, meaning the optimization is done in parallel to the actual simulation flow, the batching time is also equivalent to the maximum duration allowed to optimize the assignments, since at that point a batch of new requests would be needed to be optimized. It is found that longer batching times improve the system performance, while they intrinsically mean longer response times for customers, as they need to wait until the initial assignment of their request is found during the first optimization after they have sent it.

In [Hyland and Mahmassani, 2018], artificially generated demand in a Manhattan grid is studied, introducing six variations of ride hailing services, changing model parameters, for instance the kind of optimization (heuristic or global), the set of considered vehicles (only idle ones or also vehicles en-route to a drop-off location) and the inclusion of reassignments of requests. It is found that global optimization improves the system performance, especially in scenarios considering more subsets of vehicles and requests, in particular if the fleet size is small relative to the demand rate. A study of ride hailing services in Manhattan using the full data set is conducted in [Erdmann, Dandl, and Bogenberger, 2021]. In order to avoid long response times for customers, a two-step optimization approach is used, which first formulates pickup time windows based on one of two considered heuristics, referred to in this context as "immediate response strategies", before periodically performing a global optimization. This approach, first introduced in [Erdmann, Dandl, and Bogenberger, 2019] and improved in [Erdmann, Dandl, Kaltenhäuser, et al., 2020], is compared to service models that either only use immediate response strategies or only global optimization and it is found that the two-step method performs very similar to the service model relying only on global optimization without the drawback of long initial response waiting times.

This work builds upon the findings of [Erdmann, Dandl, and Bogenberger, 2021], by enhancing the evaluation of the ride hailing services used in this study, to find an answer the question, how an ODM service model needs to be designed to combine optimal assignments of requests with quick response times, as formulated in RQ1 at the end of Chapter 1.

Ride Pooling Services

Ride hailing services are associated with much empty mileage driven and low vehicle occupancy, which are disadvantageous not only for cities, but also for service providers that need to pay for the energy consumed during empty trips and want to maximize the number of paying customers. This led to a shift of focus of ODM providers towards ride pooling services, which allow customers to share their rides, often in exchange for discounts on their fares. Operators on the other hand can thereby increase the efficiency and profitability of each vehicle in the fleet, which leads to more customers who can be served or smaller fleets to serve the same amount of customers compared to ride hailing service models.

[Alonso-Mora, Samaranayake, et al., 2017] presents an optimization algorithm that is able to efficiently find optimal assignments of groups of customers to available vehicles. The customer groups need to be precalculated in order to find combinations that result in solutions with minimal travel delays for service users. This method is shown to be very effective for large problem instances, which makes it rather unique in the area of ride pooling optimization algorithms, which often suffer from the enormous solution space involved in such problems. Results indicate that a fleet of 3000 vehicles can serve 98 % of the demand currently covered by roughly 13 000 taxis in Manhattan when ride sharing is offered. A case study for Munich which makes use of a similar optimization algorithm is presented in [Engelhardt et al., 2019]. It also finds the algorithm to be very effective for the assignment problem. Evaluating 500 to 7500 requests per hour, results indicate that with growing demand and proportionally increasing fleet sizes the percentage of customers served also increases, further indicating the capability of the algorithm to perform well in large problem scenarios. The system performance is measured for varying maximum detour times, which is the allowed additional trip duration relative to the time needed for the direct ride from the pickup to the drop-off location. Longer maximum detour times result in more shared rides, more served customers and improved service performance overall. These results are also confirmed for a case study of Manhattan in [Hyland and Mahmassani, 2020].

Other important model parameters are examined in [Bilali, Dandl, et al., 2019] and [Bilali, Engelhardt, et al., 2020]. The maximum waiting time, as well as the time that requests are reserved before the planned pickup, are both shown to also have a significant impact on the system performance. The models used in these studies are based on the analytical calculation of the "shareability", first introduced in [Tachet et al., 2017]. This concept relies on the analytical computation of the chance to find shareable trips in a business area. It is deterministic, in contrast to most of the other studies mentioned in this section, which rely on agent-based simulations. Therefore, the impact of service parameters can be evaluated easily, since the system performance depends directly on each of them. However, such models cannot reflect systematic effects of individually acting agents, which typically results in additional effects on the service performance that cannot trivially be accounted for in analytical models and implies a certain level of randomness and unpredictability for sufficiently complex models. The uncertainty in ride pooling services is further investigated in [Fielbaum and Alonso-Mora, 2020]. In addition to the uncertainty of the individual waiting times of customers, which is also present in ride hailing services, in ride pooling another unreliable factor is the trip duration, which can change while a customer is already on board due to new requests that are served by the same car.

In this work, the challenging question, how to make full use of the potentials of ride pooling by using an ODM service model, that needs to handle the complex assignment of shareable trips, is examined further. The investigated service models are compared in their system performances and their respective way of interacting with the customer, as explained in RQ1.

Repositioning Methods

In [Horn, 2002], repositioning is part of a service that includes ride hailing and ride pooling considering "online" requests for immediate transfer and "scheduled" trips reserved by customers a certain amount of time before the pickup is supposed to happen. The repositioning algorithm uses heuristics to determine which locations in the network are undersupplied with vehicles to serve the projected demand. Two heuristics are applied: one assumes a "pseudo-omniscient" forecast of demand, taking into account the entire data set used in the respective simulation and counting the requests during a certain period of time within predefined areas of the network, accumulating vehicles accordingly. The second method projects upcoming demand based on historical data, interpolating the demand that recently occurred in the areas of the network. Both of these concepts are common in many repositioning algorithms, the latter varies in what historical data is used to project the demand, though.

The concept of using time-varying Poisson models for demand prediction was first introduced in [Ihler et al., 2006]. It is based on the idea to use the periodicity of demand patterns, e.g., anticipating higher demands for mobility on weekdays or during rush hours. The system performance varies depending on three parameters: information quality, temporal and spatial granularity. The information quality typically rises with the amount of time that is used to determine the periodic demand patterns and the number of requests that constitutes this demand, because smaller data sets generally tend to be more error-prone. The impact of temporal and spatial granularity is evaluated in [Wen et al., 2019], [Dandl, Hyland, et al., 2019], and [Hyland, Dandl, et al., 2020], indicating that higher spatial resolutions improve the system performance significantly, whereas the temporal granularity is found to have a smaller impact.

The repositioning problem can be solved as part of the vehicle-assignment problem, integrated in the objective function of the respective VRP, or as a separate optimization problem, solved periodically, using the vehicle schedules as input and constraints. The integrated method, also referred to as short-term repositioning, is used in [Alonso-Mora, Wallar, et al., 2017] and [Dandl, Hyland, et al., 2019] for ride pooling services and is found to perform well in highly dynamic scenarios, but struggles to optimally distribute ODM fleets over large business areas. A survey of a number of short-term forecasting methods is presented in [Sayarshad and Chow, 2016].

On the other hand, long-term repositioning allows to consider longer time horizons of demand forecasts and therefore to balance the supply of ODM vehicles over the entire business area according to the anticipated demand. However, long-term strategies suffer from the fact that predictions further in the future are less reliable and cannot react on short-term imbalances. A case study for the city of Austin using this method in a ride pooling service is presented in [Fagnant et al., 2016]. A combination of both short- and long-term repositioning is introduced in [Dandl, Hyland, et al., 2020], making use of the advantages of both approaches and showing promising results in ride pooling service simulations for the cities of Chicago and New York City.

Customer Models

In the research area of ODM services and the simulation of such, the subject of customer models is a rather new one and the literature background is rather scarce. The system performance as well as the degree of realism of the respective simulation evaluated is heavily impacted by the choice and design of it, however. Most of the aforementioned works use a very simplistic customer model that includes a certain minimum level of offer quality, which results in an immediate acceptance of the offer if it is met and an immediate rejection otherwise. This offer quality is mostly defined exclusively by the waiting time associated with the respective offer, meaning that operators who are aware of the maximum waiting time accepted by their customers, can make sure to maximize the number of accepted offers by completely ignoring requests that cannot be picked up in time and only optimizing requests which can be assigned to vehicles that serve them before their maximum waiting time is up.

Such a customer model is problematic for a number of reasons. First, simulations using it tend to overestimate the system performance in comparison to real-world applications because of the neglected reaction times of customers when responding to offers made by the service providers and the possibility of delayed rejections. They also do not account for the fact that each customer is different from another, including the priority and length of the maximum waiting time when it comes to the decision if an offer is accepted or not. This may result in unexpected rejections of offers, potentially after the assigned vehicle is already on its way, causing extra empty mileage, costs and blocked vehicles that might have been able to be assigned to other requests in the meantime.

The need for more sophisticated customer models is addressed in recent publications. A probabilistic model is presented in [Al-Kanj et al., 2020], where a customer model is implemented in a ride hailing service model that includes "surge pricing", a concept in which the fares for customers vary depending on the current demand. Increasing the fares during peak hours also increases the probability of customers to reject offers, which is bearable for the service provider because in these periods of the day, the utilization of the ODM fleet should be close to maximum if the fleet size is chosen appropriately. In [Dandl, Engelhardt, et al., 2021], it is shown that it can be beneficial to reject customers immediately who cannot be picked up within a defined maximum waiting time, allowing operators to focus on a relevant subset of all requests. If this maximum waiting time is sufficiently long, it can be assumed that the share of rejected customers is low enough to guarantee a net profit from this approach compared to service models that do not include any parameter for the maximum waiting time.

A novel approach to model customers in ODM services is presented in [Yu and Hyland, 2020]. The concept of this so-called "diffusion customer model" is based on the idea of users going through decision-making processes when receiving an offer from the service provider. The duration and outcome of each decision-making process depends on individual user parameters as well as the quality of the offer. Because of the novelty of this approach, it is mainly found in other research areas, often closely connected to behavioral science. The impact of age on the decision-making process is examined in [Theisen et al., 2020], where it is found that older people are slower in non-decisional processes, like understanding information and the motoric execution of responding, and tend to make more conservative decisions than younger persons, who decide more spontaneous. A detailed overview of applications of diffusion decision models is presented in [Ratcliff et al., 2016].

A detailed evaluation of the diffusion customer model as part of an agent-based ODM service simulation has not yet been conducted. This work implements a variation of it in order to address the lack of customer models used in evaluations of ODM service models, as stated in RQ2. That includes varying durations of decision-making processes, depending on offer qualities as well as customer model parameters and analyses of the interaction between customers and service models and the effects on the system performance.

Chapter 3

Problem Formulation and Methodology

After the introduction to the state of the art in the research field of ODM services, the following chapter provides details of the approach used in this work to answer the research questions formulated in Chapter 1.

The investigated service models are described and compared in Section 3.1. Section 3.2 provides the mathematical problem formulation of the optimization problem used to manage the ODM fleet throughout the conducted simulations. It also describes the differences and relations between the objectives of a service and the control function used to make assignments during a dynamic optimization problem. Section 3.3 introduces a customer model used in many studies presented in the literature, which serves as base model for this work.

The subsequent section focuses on the case study presented by first introducing the simulation framework used to test the various service models in Section 3.4.1. The parameters used during these simulations are described in Section 3.4.2, differentiating between constants and variables evaluated in parameter sensitivity analyses of the service models. The performance indicators used to compare the models are presented in Section 3.4.3.

The definition of the service models investigated and the formulation used to describe the dynamic optimization problem in this work are essential to understand its means and results. This section therefore first introduces the ODM use cases considered, namely ride hailing and ride pooling. Subsequently, three service models are described: Service Model 1 relies on assignments of user requests and ODM vehicles only based on a heuristic procedure, Service Model 2 combines this heuristic approach in a 2-step service model with the potential of global assignment optimization, while Service Model 3 does not use any heuristics and only uses globally optimized assignments.

The concept of how the fleet of ODM vehicles is managed in each of these service models varies, albeit some aspects remain unchanged. For example, the global objective and the control function used in all three service models are the same. Also, the repositioning algorithm used to increase the availability of vehicles in all relevant parts of the business area stays the same throughout all simulations. Both of these constant elements of this work are described in Section 3.2, followed by the definition of the customer model used as a base model in this work.

3.1 Service Models

Like many other ODM services described in the literature, the models considered in this work share some common properties:

- Vehicles in the fleet are coordinated by a central operating algorithm ("operator"), which assigns jobs to them, that are listed chronologically in each vehicle's own task queue.
- While the fleet as a whole is homogeneous, each individual vehicle acts as an agent within the simulation framework.
- Service requests are unknown to the system until they are sent to the service operator and need to be handled dynamically.
- Assignments of vehicles to requests are based on a control function designed to optimize the service performance.
- If the user waiting time t_w until pickup implied by an assignment made by the operator surpasses a certain maximum waiting time t_max, the user rejects the service offer.
- The repositioning of vehicles within the business area of the service is managed by a separate algorithm which remains unchanged between the simulations.

Besides these shared features, the three service models that are compared in this work differ in their basic concepts for assigning vehicles to requests and their respective customer-operator interactions, which are explained in detail next.

Service Model 1: No Global Optimization

The first service model is in many ways the simplest one. The reason for this is its main characteristic: assignments of vehicles to user requests are exclusively based on a heuristic procedure. It is therefore referred to as Service Model 1 with "no global optimization".

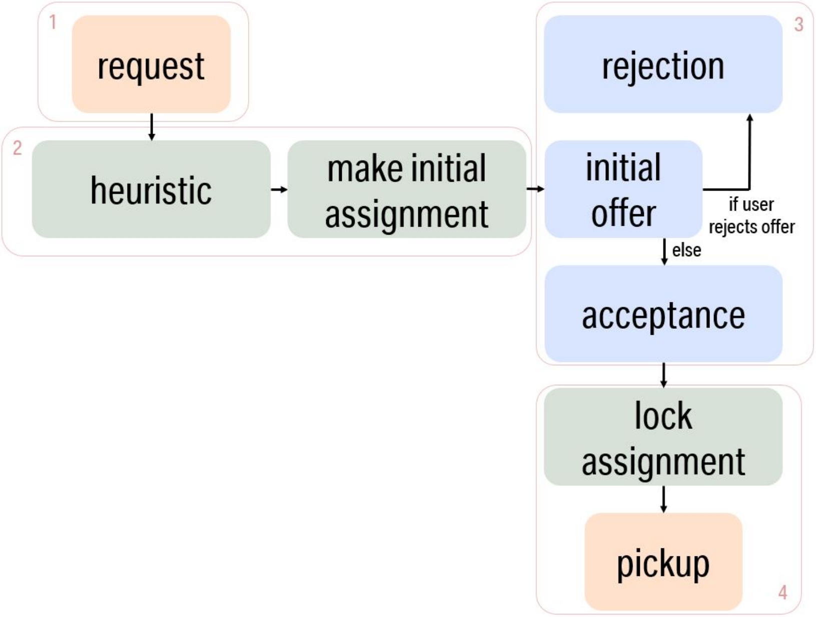


Figure 3.1: Concept of Service Model 1.

Figure 3.1 illustrates the concept of the assignment process in Service Model 1. Orange items represent the start ("request") and end point ("pickup") of the assignment process, green items describe backend processes on the side of the operator, and blue items indicate elements which depend on user decisions.

Using this service model, after a user has sent a service request (box 1), the operator is able to communicate an offer immediately, based on an assignment made with a use-case specific heuristic method (box 2). If the service offer is rejected by the customer, the request is removed from the system (box 3). If it is accepted by the user, the assigned vehicle is immediately locked to the respective request. The user then receives the vehicle identification number (ID) as well as either (i) the exact pickup time, because later changes are impossible due to the design of the heuristic method (ride hailing use case), or (ii) a projected pickup time window, because changes within the task queue of the assigned vehicle are allowed by the (ride pooling) heuristic until the projected waiting time becomes shorter than t_{lock} (box 4).

Service Model 1 can provide a very good service experience for customers in terms of response time and reliability, because the initial assignment can be found and communicated very quickly. On the other hand, the assignments made for each request independently, based on the heuristic methods, do not necessarily lead to an optimal – or even good – solution for the whole system.

Service Model 2: 2-Step Service Model

The second service model aims to combine the advantages of quick responses due to initial assignments based on heuristics with the potential of global optimization. The concept includes possible reassignments of individual requests that can be part of solutions found during periodical optimizations, which include all assignments that are not locked yet. When the projected pickup of a particular user is imminent, the currently assigned vehicle is locked to the respective request and the user receives a second offer with the details of the pickup. Service Model 2 is therefore referred to as "2-step service model", because unlike Service Model 1 two offers are sent to the users.

An overview of the assignment process of Service Model 2 is presented in Figure 3.2. Similar to Service Model 1, after the request is sent and the initial assignment is made based on a heuristic method, the respective user decides whether or not the associated offer is accepted or not (boxes 1-3). The initial offer includes a pickup time window in which the user is projected to be picked up in. If the initial offer is accepted, the operator checks if the implied user waiting time t_{w} until pickup is shorter than a predefined service variable t_{lock}. This parameter is referred to as "lock time" and specifies at which point before the pickup an assignment is locked. If the pickup is imminent and hence the waiting time shorter than t_{lock}, the assignment is immediately locked and the user does not receive a second offer until the eventual pickup (box 4). In fact, the service experience for a user of Service Model 2 is then indistinguishable from Service Model 1.

If, however, the time until pickup is longer than t_{lock}, the assignment is not yet locked, but instead subject to the next global optimization, happening at the end of each optimization period, during which requests are collected and bundled. This process is referred to as "batching". As a result of optimization, vehicles can potentially be reassigned to other requests or assignments can remain unchanged, depending on what is globally the best solution according

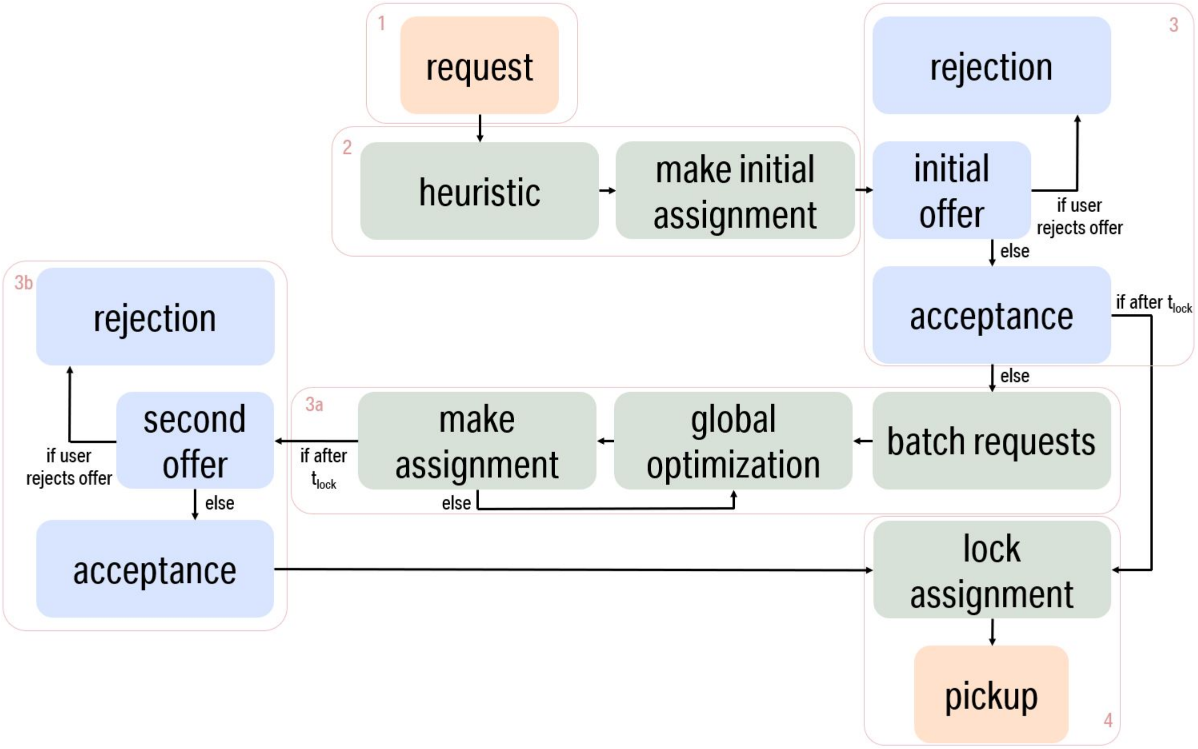


Figure 3.2: Concept of Service Model 2.

to a predefined control function (box 3a). A request can only be reassigned to another vehicle during global optimization as long as the assignment is not locked. As soon as the currently assigned vehicle is close enough to the pickup location, so that the remaining waiting time is shorter than t_{lock}, a second offer is sent to the user, including the exact pickup time as well as the vehicle ID (box 3b). Again, this offer can be rejected by the customer, which results in the request to be eliminated from the system. If it is accepted, though, the assignment is locked and will not be changed until the pickup.

This 2-step service model targets the weak spot of Service Model 1 by including periodic reassignments of vehicles according to a control function, which aims to optimize the system performance overall. Hence, the user experience is no worse in terms of response times, because the same heuristic procedures are used, and the average waiting times until pickup are potentially better because of the global optimization. The downside of decreased reliability for customers because of the chance to be picked up before or after initially projected is addressed with the inclusion of a pickup time window in the initial service offer, that limits the uncertainty of the waiting time.

A flaw of both Service Models 1 and 2 is the intrinsic first-come-first-serve principle that leads to a higher chance for users that send their requests later to receive a worse service offer, ultimately potentially resulting in a rejection. The probability for this is independent of the potential profit or the impact on the system implied by the requests in question, but solely depends on the order in which they have been sent. The potential forfeit in system performance can only be avoided by not immediately respond to user requests, but instead delay the initial offer in order to evaluate service requests that are sent shortly after and compare them with each other before sending the best service offers to the most beneficial ones for the overall system. Such an approach can decrease the perceived service quality for users, though, because the response time after the initial offer is crucial for a good user experience.

Service Model 3: Only Global Optimization

The third and final service model considered in this work exploits precisely this chance for improved system performance at the cost of longer response times. It does not make use of any heuristic method to find an initial assignment for new requests, but delays the initial service offers until the end of each optimization period after taking into account all new and all unlocked requests in a global optimization. Hence, Service Model 3 is referred to as the one using "only global optimization".

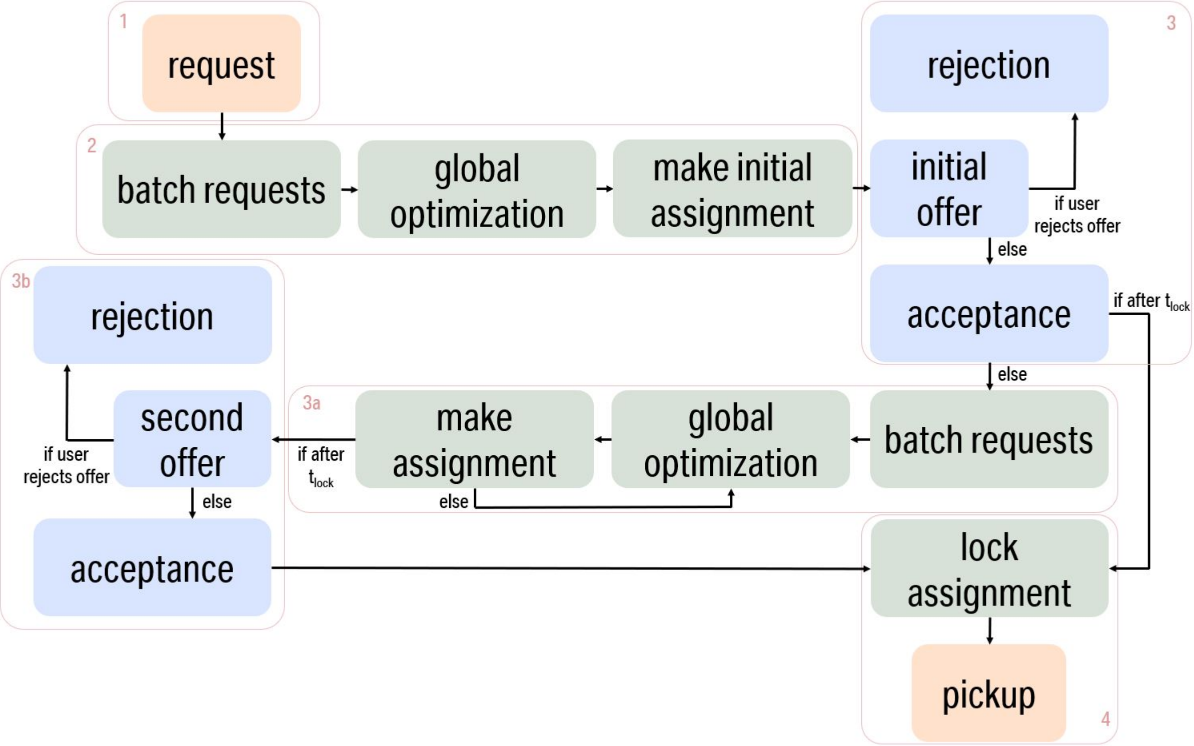


Figure 3.3: Concept of Service Model 3.

In Figure 3.3, the concept of this service model is illustrated. Most of the service's aspects are similar to Service Model 2, including the period between initial offer acceptance (box 3) and eventual pickup (box 4). The crucial difference between both models can be found between the service request (box 1) and the initial offer (box 3). Instead of immediately responding to the user, the initial assignment of a vehicle to the new request is only made after it is batched with other new requests during the current optimization period and the subsequent periodic global optimization takes place (box 2). Since the optimization period is a fixed service model parameter, the average response time is half this period's length, which results in a noticeable delay of information as well as potentially longer pickup waiting times t_{w} experienced by customers due to the delayed assignment of vehicles.

The optimization potential added by the globally optimized assignments leading to the initial offers to service users has to make up for these systematic flaws of the service model. While the simulated users in this work take into account the longer waiting times when reacting to service offers, the added response time is not directly considered in the evaluation of offer qualities. The system performance achieved with Service Model 3 therefore has to be considered as an upper bound of the optimization potential.

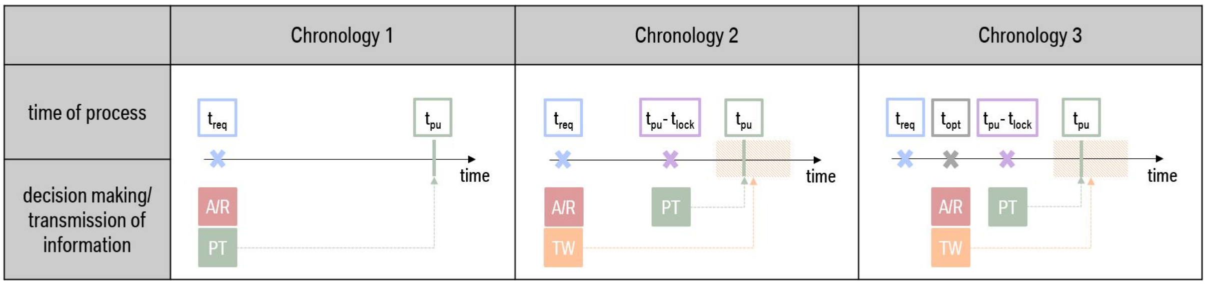


Figure 3.4: Chronologies of communication for three service model concepts. Note varying timings of communicating acceptances/rejections (A/R), pickup times (PT) and time windows (TW). Inspired by [Erdmann, Dandl, and Bogenberger, 2021].

All three service models presented in this work include unique features that bring benefits as well as disadvantages for service providers, official authorities and service users. Evaluating them and quantifying these differences will help decision makers to decide which service models fits their needs best. An overview of the chronologies of communication processes between service providers and users for various assignment concepts is presented in Figure 3.4.

Note that Chronology 1, shown on the left, only represents the temporal order of communications in Service Model 1 in the ride hailing use case. As described before, the decision if the request is accepted or rejected ("A/R"), as well as – in case of an acceptance – the information about the exact pickup time ("PT") can be communicated to the customer immediately when the request is answered by the operator.

In the ride pooling use case, Service Models 1 and 2 appear very similar from a customer's point of view. In both models, when initially responding to the customer, the service operator communicates a projected pickup time window ("TW") to accepted users. As soon as the remaining waiting time until the anticipated pickup time t_{pu} reaches a predefined lock time t_{lock}, the exact pickup time is locked and sent to the customer. This chronology is presented in the center of the figure and also represents Service Model 2 in the ride hailing use case.

Chronology 3 is the temporal order of communication in Service Model 3 in both the ride hailing and ride pooling use case. As shown, the decision if a certain request is accepted or not is made by the operator during the first global optimization after the time of request. At this point, accepted users receive a projected pickup time window, before also receiving the exact pickup time as soon as it is locked.

3.2 Fleet Management

The ODM services evaluated in this work are all operated by a central entity, referred to as service operator. The service operator aims to efficiently coordinate a fleet of vehicles in order to fulfill certain objectives. These objectives can be manifold and complex, because the operator has to consider multiple stakeholders, including the service provider, who wants to maximize the profitability, the customers, who need to be satisfied with the service experience in order for the service to be successful, as well as the official authorities of the area the service is offered in, which have the power to effectively shut down the local business entirely.

The means how exactly to achieve that goal varies between the service models and use cases considered in this work. However, one central element of the operator remains the same in order for the scenarios to be compared equitably: the control function F_{con}. This function defines the basis on which assignments between service vehicles and user requests are made, both in heuristic and globally optimized procedures. Because these assignments are made dynamically, the control function is not necessarily identical with the objective function F_{obj} of the service, which represents the long-term optimization of certain service parameters. However, F_{con} and F_{obj} are closely connected and need to be formulated carefully in order for the service to perform well in the most important categories.

The control function used in this work includes three terms F_1, F_2 and F_3. F_1 represents the total driven mileage associated with a solution, F_2 the total user waiting time and F_3 the number of requests served. Their respective priorities are represented by the weights α, β and γ, each associated with one of the terms. The general form of the control function is

$$F_{\mathsf{con}} = \alpha F_1 + \beta F_2 + \gamma F_3. \tag{3.1}$$

F_{con} evaluates the quality of a solution to the current state of a dynamic DaRP, which consists of assignments of task queues ξ_i of length N_i to vehicles $i \in I$. Every task queue is a list of jobs, which the respective vehicle needs to perform at an associated location. In the context of vehicle-user assignments, a job can be either a user pickup or a user drop-off.

The standard set of constraints of the DaRP formulation used in this work can be summarized as follows:

- Constraint 1: *Unitarity*
  Each user is served by one vehicle at most and to each vehicle at most one task queue is assigned.
- Constraint 2: *Precedence*
  Each served user's pickup takes place before that user's drop-off.
- Constraint 3: *Capacity*
  At no time any vehicle carries more passengers than it can carry.
- Constraint 4: *Maximum Waiting Time*
  No assignment implies a user waiting time t_{w} longer than t_{max}.

All of these constrains are commonly used in DaRPs and guarantee that solutions are valid and physically feasible. In the simulations of this work, Constraints 2-4 are handled as part of the preprocessing of the optimization.

Depending on the use-case specific constraints, a task queue might consist of multiple jobs associated with one or more user requests. The task queue of a vehicle $i \in I$ that serves a subset ("bundle") of users $k \subset J$ on the best tour, is defined as ξ_{ik}. The means of how to find the best tour for a bundle of users are described in the chapters covering the ride hailing and the ride pooling use cases, respectively. Additionally, K_j describes the set of bundles $k \in K_j$ which contain request $j \in J$, K_i the set of bundles $k \in K_i$ which can be assigned to vehicle $i \in I$, and I_k the set of vehicles $i \in I_k$ that are connected to a bundle k. The number of users in bundle k who are picked up by vehicle i outside of their respective time window when applying the associated task queue ξ_{ik} is referred to as $n_{\mathsf{p}}(\xi_{ik})$, while the subset $J_{\mathsf{a}} \subset J$ contains users $j \in J_{\mathsf{a}}$ which already accepted an offer and should hence be part of subsequent solutions. These sets are also generated in the aforementioned preprocessing step of the assignment procedure.

If an assignment of vehicle $i \in I$ to a bundle of users $k \subset J$ is part of a solution, the corresponding decision variable x_{ik} is defined as 1, otherwise $x_{ik} = 0$. Since the solution, encoded as a set of decision variables x_{ik}, is the only variable in the control function, the minimization of F_{con} can be written as

$$\min_{x_{ik}}(F_{\mathsf{con}}(x_{ik})) = \min_{x_{ik}} \left(\alpha F_1(x_{ik}) + \beta F_2(x_{ik}) + \gamma F_3(x_{ik})\right) \tag{3.2}$$

$$\text{s.t.} \sum_{k\in K_i} x_{ik} \leq 1 \qquad \forall i \in I \tag{3.3}$$

$$\sum_{k\in K_j}\sum_{i\in I_k} x_{ik} \leq 1 \qquad \forall j \in J \tag{3.4}$$

$$\sum_{k\in K_j}\sum_{i\in I_k} x_{ik} = 1 \qquad \forall j \in J_{\mathsf{a}} \tag{3.5}$$

$$x_{ik} \in \{0,1\} \qquad \forall i \in I, \forall k \in K_i. \tag{3.6}$$

Equation 3.3 ensures that to each vehicle $i \in I$ at most one bundle of requests $k \in K_i$ and therefore one associated task queue ξ_{ik} is assigned. In Equation 3.4, the maximum number of vehicles $i \in I_k$ to which a bundle k of requests can be assigned to, is set to one for all requests $j \in J$. Already assigned requests are guaranteed to be part of the solution by Equation 3.5. Together with the definition of x_{ik} as a binary decision variable in Equation 3.6, the combination of the constraints in Equations 3.3 and 3.4 represents the unitarity constraint.

The first term F_1 represents the total mileage driven by all vehicles $i \in I$ of the fleet. It is calculated by adding the distances d between the current position of the vehicle to the first stop and between all stops associated with the jobs in each vehicle's task queue ξ_i.

$$F_1 = \sum_{i,k}\left(d\left(z_i, z(\xi_{ik}^1)\right) + \sum_{n=2}^{N_i} d\left(z\left(\xi_{ik}^{n-1}\right), z\left(\xi_{ik}^{n}\right)\right)\right) x_{ik}, \tag{3.7}$$

where n is the index of the task queue of vehicle i with length N_i, $z\left(\xi_{ik}^n\right)$ is the location associated to the job with index n in the task queue ξ_{ik} and $d(z_1, z_2)$ is the route length between locations z_1 and z_2. Note that if a task queue is empty, the subset k of served requests is empty too and the mileage of vehicle i does not contribute to the overall sum. If only one job is in the task queue ξ_{ik} of vehicle i, $N_i = 1$ and the second sum in Equation 3.7 covering the distances between jobs in ξ_{ik} does not contribute to the distance driven by the vehicle.

Because the optimization target in Equation 3.2 is to minimize the overall sum of the weighted terms, preferable solution attributes contribute negative values, while undesired aspects of assignments are modeled as positive parameters, that need to be avoided in order to minimize F_{con}. Hence, the weighing factor α is chosen to be positive. The values chosen for all weighting factors are presented in Section 3.4.2, in which the parameter sets of the case study are described.

The second term F_2 in Equation 3.1 is the representation of the total user waiting time implied by a solution. The sum of individual waiting times $t_{\mathsf{w},k}(\xi_{ik})$ of each bundle $k \subset J$ of users is calculated for all potential task queues ξ_{ik} as part of the preprocessing. The total user waiting time then reads as

$$F_2 = \sum_{i,k} t_{\mathsf{w},k} x_{ik}. \tag{3.8}$$

Again, this term is targeted to being minimized, hence β is chosen to be positive.

The third and final term F_3 of the control function represents the total number of user requests served by vehicles assigned according to the solution and penalizes assignments that imply pickups outside of time windows associated to users. Because of the unitary constraint, the number of requests served is equivalent to the sum over the cardinalities of the subsets $k \subset J$ of requests that are part of the solution.

$$F_3 = \sum_{i,k} \left(|k| - p_{\mathsf{tw}} n_{\mathsf{p}}(\xi_{ik})\right) x_{ik} \tag{3.9}$$

The penalty $p_{\mathsf{tw}} > 0$ is subtracted from the sum for each user who is not picked up within the time window communicated in the initial offer. Because $n_{\mathsf{p}}(\xi_{ik}) \leq |k|$, if the value of p_{tw} is chosen to be not greater than 1, $F_3 \geq 0$. Unlike F_1 and F_2, the goal of the operator is to maximize F_3. Therefore, it contributes a negative value to F_{con} by choosing γ to be negative. Because of the relatively low values expected for F_3 compared to F_1 and F_2, the absolute value of $|\gamma|$ should be chosen to be considerably higher than $|\alpha|$ and $|\beta|$, especially if the priority of accepted requests is set high. The relation of α, β and γ states the priority of assignments between minimizing driven distances, user waiting times and pickups outside of pickup time windows. Their relation is further explored in parameter sensitivity analyses in Sections 4.2.2 and 5.3.2.

In addition to the control function used to make the assignments of user requests and service vehicles, the operator has another tool to efficiently manage the ODM fleet: repositioning. This term describes the (periodic) redistribution of vehicles in anticipation of upcoming local demand. The repositioning approach used in this work is based on the rebalancing policy introduced in [R. Zhang and Pavone, 2014] and the forecasting method used in [Dandl, Hyland, et al., 2019].

The concept can be summarized as follows: in predefined intervals t_{r} the operator executes an optimization process which is separated from the assignments of vehicles and user requests discussed before. This process aims to assign a third kind of job to vehicles that are currently idle and whose task queues are therefore empty, namely "reaching repositioning zone". A repositioning zone is one of several parts of the business area, all designed to be roughly the same in size. For all zones the demand during the day is assumed to be approximately known from historic data.

At each repositioning decision time step t, for each zone z the number of vehicles v_z is counted, which a) are currently in the zone and idle, b) are currently heading towards the zone and finish their task queue there, or c) are scheduled to do so. This number of vehicles v_z is compared to the anticipated demand c_z in zone z at a certain time in the future, referred to as repositioning time horizon t_{hor}, and the difference is called "excess vehicles" $v^z_{\text{exc}}(t)$.

$$v^z_{\text{exc}}(t) = \max(v_z(t) - c_z(t + t_{\text{hor}}), 0). \tag{3.10}$$

Note that $v^z_{\text{exc}}(t)$ is defined to be 0 if the projected demand $c_z(t + t_{\text{hor}})$ is higher than the number of vehicles v_z that is expected to be in zone z. The goal of the optimization procedure is to evenly split the excess vehicles among all zones while minimizing the driven distance to do so, only using idle vehicles. If a repositioning task is assigned to an idle vehicle $i \in I$, the associated job is added to the respective task queue ξ_i.

This approach is efficient in terms of balancing vehicle supply according to upcoming demand. Its effectiveness, however, depends on the definition of the zones, as well as the temporal and spatial distribution of demand. Because it only considers idle vehicles eligible for repositioning, it tends to be rather conservative in the sense that it produces less mileage compared to other methods, while being less active in periods of high demand, potentially leading to growing imbalances during the day.

The repositioning algorithm used in this work is the same throughout all simulations. Its parameters are listed in Section 3.4.2.

3.3 Conventional Customer Model

In ODM service models, the other integral part besides the fleet management is the customer model used to simulate the behavior of service users. As mentioned in 2.3, many studies rely on rather simple customer models to focus on the system impacts of the evaluated services and assignment methods. This work introduces a more sophisticated approach to model customers in Chapter 6, which is compared to a "conventional customer model" (CCM). The latter is described in this section.

The first property of the CCM is also a simulation framework specification: user requests join the system dynamically, making it necessary for the operator to react accordingly. This characteristic makes the vehicle-user assignment optimization problem to a dynamic DaRP.

Each of the user requests contains the following individual attributes:

- Identification number (ID) of the request $j \in J$,
- Request time t_{req},
- Pickup location $z_{\text{pu},j}$,

- Drop-off location $z_{\text{do},j}$,
- Number of passengers

The ID $j \in J$ is unique for each request and is used to track each user during the simulation. The request time t_{req} is the simulation time at which the request joins the system. It also defines the point in time at which the user starts waiting for the pickup. The pickup location $z_{\text{pu},j}$ as well as the destination $z_{\text{do},j}$ of user $j \in J$ are coordinates inside of the business area of the service. Each request is associated with a certain number of passengers that are treated as an inseparable group of people that is picked up and dropped off together at the same location and the same time. For the purpose of easy reading, in this work such a group of passengers is referred to as "customer" or "user", independent of the size of the group, if not specified otherwise.

Besides these variable request attributes, all users in the CCM share some model parameters and features. Every boarding process, both entering and leaving a vehicle, takes a certain amount of time, referred to as boarding time t_{boa}, which is assumed to be constant for all users. This boarding time is neither added to the waiting time t_{w} when a vehicle arrives at the pickup location, nor to the travel time on board of a vehicle, but merely increases the overall time it takes for vehicles to serve user requests on top of the travel time on the road. This feature of the CCM can be considered a step towards a realistic customer model compared to models used in most of the works in literature that neglect this additional delay in the operation of ODM services.

However, the CCM also simplifies certain aspects of real-world users. If the waiting time t_{w} associated with an assignment made by the operator is not longer than a predefined maximum waiting time t_{max}, users in the CCM always accept the offer. Likewise, if the offer is considered "bad" because $t_{\text{w}} > t_{\text{max}}$, the offer is always rejected. This simplification of the perceived offer quality does not take into account gradual differences between offers or individual user expectations.

Another crucial feature of the model is the fact that every decision, both acceptances and rejections, are assumed to be made within one simulation step t_{step}. Again, this is a clear simplification of the real-world behavior of service users, who often take a considerable amount of time weighing the options or compare offers of multiple ODM service providers.

The CCM is a simplistic customer model designed to be used in large-scale ODM service simulations. It facilitates the understanding of correlations within the framework by reducing the properties of users to their most fundamental characteristics. Therefore, it is chosen as a base model in this work.

3.4 Case Study

To test the service models introduced in Section 3.1, agent-based simulations are conducted in a framework designed to evaluated system performance. This simulation framework is stated in the following Section 3.4.1, including technical details, used data sources and the typical simulation flow, which is used in both the ride hailing and ride pooling use case.

Since the models' performances depend on fine-tuned parameter settings, a parameter sensitivity analysis is conducted for crucial system variables of the 2-step service model. Those are introduced alongside the constant system parameters which remain the same throughout all simulations in Section 3.4.2. The most important key performance indicators are defined and specified after that in Section 3.4.3.

3.4.1 Simulation Framework

For comparability, the simulations presented in this work are all executed on a single central processing unit of one multi-core computer, namely an Intel Xeon Silver 4114 processor with ten physical cores at 2.20 GHz and 64 GB random access memory. This allows ten simulations to run in parallel without affecting each others computation performances, thereby speeding up the overall process of conducting all simulations considered in this work.

The main language used to write the code of the simulation framework is Python 3.7 [Python Software Foundation, 2021], which facilitates the implementation of all aspects of such a framework with its vast open-source libraries and packages, including data management (*pandas* [McKinney et al., 2010]), array computation (*numpy* [Harris et al., 2020]) and visualization (*matplotlib* [Hunter, 2007]). The structure of the code is modular, meaning individual parts of the code can easily be changed without interfering with the rest of it. Therefore, all service model use cases can be simulated using the same general framework, only varying in how the respective service operator manages the fleet or in what customer model is used. This allows to examine and compare the models easily.

The data set used in the simulations is based on the Manhattan taxi data set [NYC Taxi & Limousine Commision, 2021], which is a often used in the research area of ODM services. In order to test the performances of the considered service models in various demand scenarios, simulations are executed over the span of a whole week of simulation data. This allows to include high densities of demand, e.g., weekdays during peek service hours in the morning or the afternoon, as well as periods of low demand, like weekend or night times. The week considered in this work is Sunday, November 12, 2018 to Saturday, November 18, 2018. In addition to the general variations between the days of a week, this data set includes an outlier at Wednesday, November 15, 2018, in which the overall demand drops around 28 % compared to the demand on the other weekdays. This can be explained by the fact that at this day a blizzard hit the area of New York City [Fitzsimmons et al., 2018], resulting in a significantly lower demand for taxis and transportation in general.

In order to fairly compare system performances of service models in the ride hailing and the ride pooling use case, the simulations use the exact same data sets. Since the optimization problem considered is a dynamic version of the DaRP, it is NP-hard and is therefore assumed to scale exponentially with the problem size in terms of computation time, as described in Section 2.1. This prohibits the usage of the full data sets, because simulations would take an unreasonable amount of time to finish, especially scenarios considering the ride pooling use case. Hence, this work uses data sets containing 10 % of the original Manhattan taxi data sets. This allows all simulations to finish in a reasonable amount of time, while still representing problem instances large enough to encounter systematic effects that are associated with dynamic, agent-based simulations.

In Figure 3.5, the number of requests in all seven demand data sets considered in this work are presented. As described, each consists of 10 % of the original Manhattan taxi data set from

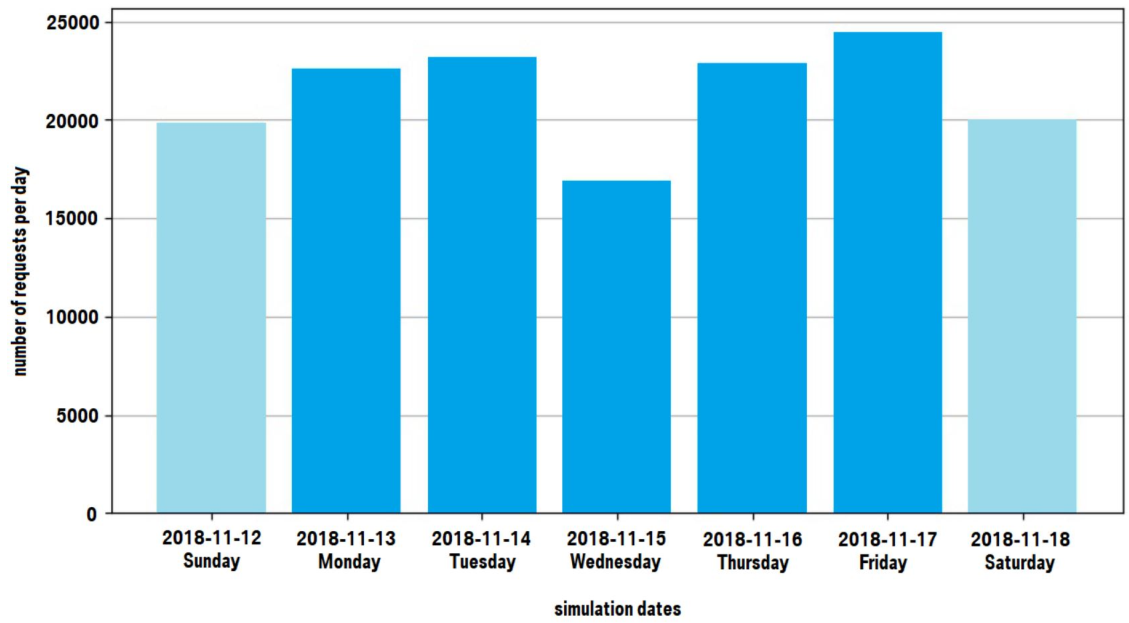


Figure 3.5: Requests per day at simulation dates.

[NYC Taxi & Limousine Commision, 2021]. The respective requests in the sampled data sets are chosen randomly, thereby preserving the original overall demand distribution over the day. The drop in demand at the weekend (Sunday, November 12 and Saturday, November 18) is apparent, as well as the sharp decrease in the number of requests at November 15, which correlates with the aforementioned blizzard in the area of Manhattan.

Besides the overall reduction of the data set sizes, another adjustment of the original data sets is the local accuracy of the pickup and drop-off positions, which is provided as one of 265 zones in the original data, segmenting the entire business area of Manhattan. The adjusted data sets connect specific network coordinates with the pickup and drop-off locations of each request, which represent randomly chosen locations in the respective zones the user was picked up and dropped off in according to the original data set.

The final adjustment made to the data set is the number of passengers associated with each request. Because taxis in Manhattan vary in size and their number of passenger seats from four to nine, individual requests can be made for groups of up to nine passengers. In the service model considered in this work, however, the service vehicles are all supposed to have four passenger seats, which means all potential requests of five or more passengers could not be served and would hence be rejected by the service provider. It is therefore assumed that such requests would not even be made in the first place, meaning the demand data sets include only requests with one to four passengers. The number of passengers associated with a request is determined randomly using a probability distribution based on the respective number of requests with one, two, three and four passengers in the original data sets.

Figure 3.6 illustrates a typical simulation flow. After initializing the simulation and setting the simulation time t_{sim}, as well as measured quantities, like the number of served requests R_{s} and driven vehicle mileages D_i, to zero (box 1), the main simulation loop is started (box 2). In each iteration of the loop, t_{sim} is increased by the simulation time step t_{step} and a check is

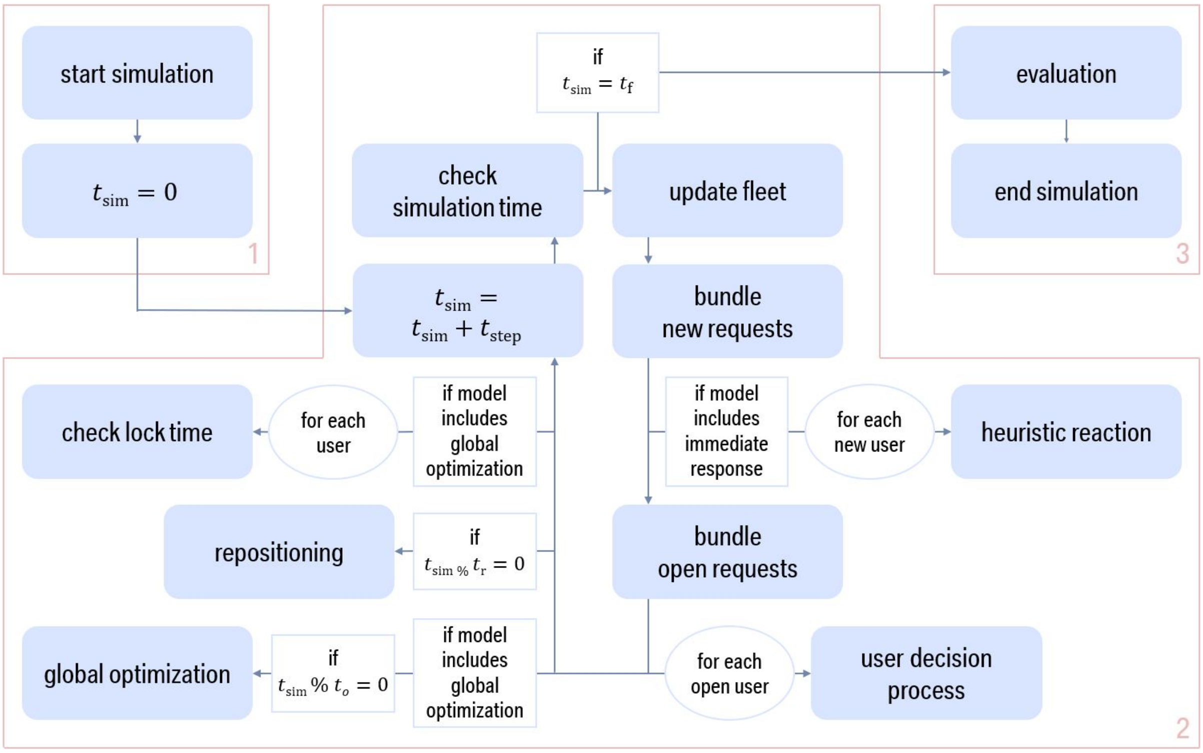


Figure 3.6: Flow chart of the simulation.

performed, which stops the simulation if the final time step t_{f} is reached (box 3).

If not, the status of the fleet is updated, meaning the positions of vehicles are adjusted with respect to their respective task queues. The driven mileage D_i of each vehicle $i \in I$ is increased accordingly. In addition, if a vehicle executes a job of its current task queue, e.g., reaching the pickup location of a user request, the job is deleted from the task queue and the execution of the next job is started, e.g., the boarding process of a user. For each request that is dropped off during this step, R_{s} is increased by one and the request ID j is added to the set of served request J_{s}.

After updating the positions and task queues of all vehicles of the fleet, the simulation proceeds by reading the demand data file and bundling all requests that are sent at the current time t_{sim}. If the service model used in the simulation includes an immediate reaction by a heuristic, the respective method is carried out for each of the new user requests. Depending on the decision made by the heuristic, the request is either accepted, an offer is made based on the heuristic assignment, the task queue of the respective vehicle is updated and the request is added to the list of currently open requests or the request is rejected and removed from the system.

When every request has been handled according to the service model in use, all users that are considered open begin or continue their respective decision-making process. Its parameters depend on the customer model used in the simulation. The available outcomes of one iteration of this process are (a) the rejection of the last offer made by the operator, (b) the acceptance of the last offer made by the operator, or (c) the continuation of the decision-making process. In case of (a), the request is deleted from the list of open requests and from the system as a whole. If the outcome of the decision-making process is (b), the request is also not considered open anymore, but is instead added to the list of accepted requests. In simulations of service models that do not include global optimization, the assignment is also immediately locked. If the process ends neither in a positive nor negative decision, the outcome is (c) and the request remains open.

In the subsequent part of the simulation loop, the global optimization is conducted in case it is part of the service model considered and if the current simulation time t_{sim} coincides with the optimization period t_{o}, meaning the remainder of a division of t_{sim} by t_{o} is zero. This remainder is calculated with the so-called modulo function $(\%)$. Therefore, if t_{sim} is a multiple of t_{o} and the optimization period is over, $t_{\text{sim}} \ \% \ t_{\text{o}} = 0$. The global optimization takes into account all requests, which are associated with a user that accepted the service offer, but which is not locked to a vehicle yet. The assignments implied by the optimal solution found during the optimization process overwrite the latest vehicle assignments and the vehicle task queues are adjusted accordingly.

Similar to the global optimization, the repositioning of vehicles happens periodically in intervals of t_{r}. If $t_{\text{sim}} \ \% \ t_{\text{r}} = 0$, this interval is over and the repositioning algorithm described earlier in this section is executed. The task queues of vehicles that are identified to be relocated by the algorithm are updated with the repositioning jobs when the process finishes.

The last part of a simulation loop before its next iteration starts is the check for users that are close to being picked up and whose assignments therefore become locked. This step is only necessary in simulations which use a service model that includes global optimization, because otherwise all accepted requests are automatically locked. If executed, during this step all projected pickup times t_{pu} implied by assignments associated with accepted requests are compared to t_{sim}. If the remaining time between t_{sim} and t_{pu} is shorter than a certain lock time t_{lock}, the request is removed from the list of accepted requests and instead added to the list of locked requests. Hence, the respective vehicle-user assignment is fixed and cannot be changed anymore during future global optimizations.

With that, the main simulation loop closes and t_{sim} is increased again by the value of t_{step}. If $t_{\text{sim}} = t_{\text{f}}$, the loop stops and the simulation enters its final phase (box 3). All remaining jobs in the task queues of the vehicles are executed and the statistics collected during the run are saved. Finally, these statistics are evaluated in the form of performance indicators, which can be compared between all simulations.

3.4.2 Parameter Sets

The simulation framework of this work depends on a number of system parameters that affect the performance of the service models investigated. Some of these parameters are kept constant throughout all simulations considered in this work, while others are treated as variables and analyzed in a parameter sensitivity analysis (PSA).

Constant Parameters

Parameters of the simulation framework that are considered constant meet one or more of the following conditions.

- The parameter is a fundamental part of the framework and variations would imply massive changes in the overall system.
- Variations of the parameter would reduce the reproducibility, transparency or plausibility of simulations and results.
- Changing the parameter would result in obvious consequences for the system performances of all service models and are therefore irrelevant for their evaluation.

Fundamental parameters are the underlying network the simulations run on, including distance matrices conducted from a preprocessed routing procedure based on Dijkstra's algorithm [DIJKSTRA, 1959], business areas and vehicle models, as well as the repositioning algorithm that is used to efficiently distribute vehicles to areas of the business area in which the upcoming demand is high.

An example for a parameter that is kept constant because modifying it would imply worse reproducibility or transparency is the static travel time matrix used in all simulations. Instead of dynamically changing traffic and weather conditions and therefore varying travel times between nodes of the network, this work assumes constant values during all simulation dates. This choice was actively made, even though it is a simplification of the model that might seem to decrease its plausibility.

The reason for this decision is the fact that dynamic travel times are a layer of modeling, which adds complexity to the overall system without adding significant insight to the evaluation of the models. Results found with dynamic travel times may be closer to real-world service performances, depending on the quality and accuracy of the data used for generating the matrices. However, differing results between evaluated scenarios would be harder to evaluate because of the additional modeling layer, affecting system performances independent of the respective service model considered.

This is avoided in this work by using a single travel time matrix, which is not updated during simulations or between simulation dates. The values in this matrix are generated by calculating the average travel times between two nodes in the network throughout all simulation dates. Hence, vehicle velocities tend to be overestimated during peak hours, because travel times would normally be higher than average, while the static travel time matrix implies trips being made outside of periods with high demand take longer than they would be if dynamic travel times would be used.

One constant element of the simulation framework that actively adds plausibility to the evaluation is the static demand data set for each simulation day considered. This means for all simulations of a specific day, the demand data consists of the same set of requests with their respective request time, pickup and drop-off locations and associated number of passengers. As described before, this data set is based on the original Manhattan taxi data set of the week from November 12, 2018 to November 18, 2018 (see Section 3.4.1), assuring a realistic local and temporal distribution of requests. Every date is simulated independently to allow parallel simulations. Considering several days, including weekdays and weekend, allows to avoid unnecessary randomness and arbitrariness, which would be implied by other methods, like randomly generated data based on only one simulation date.

The last kind of constants are parameters which would lead to obvious changes in the performances of the evaluated models. There are plenty of such, e.g., an increased value for the fleet costs due to driven distances would directly affect the potential profit of a service or a longer boarding time would imply a lower vehicle efficiency, resulting in less served requests and longer user waiting times overall. Model parameters like these are set to certain values that are assumed to represent realistic service parameters and remain the same throughout all simulations. A list of the most important constant parameters used in this work is presented in Table 3.1.

Parameter	**Symbol**	**Value**
Simulation duration	t_f	24 h
Demand	R	Manhattan taxi data, Nov.12-18 '18 (10 %)
Fleet sizes	M	100-500 vehicles
Vehicle capacity	C	4 passengers
Simulation step	t_step	1 s
Boarding and deboarding time	t_boa	30 s
Repositioning decision time step	t_r	15 min
Repositioning time horizon	t_hor	30 min
Vehicle fixed costs	c_fix	\$25
Service base fare	p_base	\$1.50
Distance costs	c_dist	0.25 \$/km
Distance fare	p_dist	0.50 \$/km
Obj. weight of waiting time	β	$1\,\mathrm{s}^{-1}$
Obj. weight of accepted requests	γ	10^6
Time window violation penalty	p_tw	10^{-4}

Table 3.1: Constant system, model, and optimization parameters.

Variable Parameters

Besides constant parameters, there are model variables worth investigating, because they represent decisions made by the service provider when designing an ODM service. Each of these parameters can be adjusted, resulting in positive or negative impacts on certain performance indicators. Some can also have an effect on the service quality perceived by users, which can have long-term consequences for the appeal of the service. In this work, a PSA is conducted for such parameters of the 2-step service model. The variable model parameters investigated in this work are the following:

- Optimization period t_o,
- Objective weight of distance α,
- Assignment lock time t_lock,
- Pickup time window length t_twl.

The duration t_o between two subsequent global optimizations is shown to have a significant impact on the system performance in service models solely depending on periodical optimization of assignments, e.g., in [SYED, KALTENHÄUSER, et al., 2019]. However, it is unclear what impact this parameter has on a service model, which relies on an initial response to service users based on a heuristic. The optimization periods considered in this work range from 10 s to 60 s. The standard value used outside of the PSA is 30 s.

In order to examine the impact of varying optimization objectives, a weighing factor α is integrated in the objective function presented in Equation 3.1. This factor effectively specifies the priority of the optimization of assignments between saving mileage driven by vehicles of the fleet on one side and the overall user waiting times between request and pickup as well as the penalization of pickups outside of associated time windows on the other. In general, the former is important to reduce the overall traffic in the business area due to unnecessary trips of ODM vehicles, which is a priority for the acceptance of the service from the perspective of a city such an ODM service is supposed to be offered in. On the other hand, short waiting times and reliable pickup time window predictions are crucial for the service quality perceived by users. Hence, α is an important parameter for service providers to carefully fine-tune when running any ODM service.

Since α is multiplied with the driven distance associated with a solution, the values examined in the PSA of this work are measured in units of m^{-1} to make the term αF_1 in Equation 3.1 unitless. For example, $\alpha = 1\,\text{m}^{-1}$ implies that each meter driven corresponds to one second of waiting time, since β is set to $1\,\text{s}^{-1}$ in the control function F_con. The range of values of α is evaluated exponentially from $6 \times 10^{-4}\,\text{m}^{-1}$, corresponding to 100 km of driven distance being penalized equivalent to 1 min of user waiting time, which heavily prioritizes saving user waiting times over distance, to $6 \times 10^{1}\,\text{m}^{-1}$, which represents an objective function weighing 1 m of driven distance equivalent to 10 min of user waiting time, thereby virtually neglecting the latter. The standard value is set to $\alpha = 6 \times 10^{-2}\,\text{m}^{-1}$, which aims to prioritize both objectives equally by penalizing 1 km of driven distance like 1 min of user waiting time.

The last two parameters analyzed in PSAs are directly linked to the service user experience. The period between when the final assignment of a vehicle to a request is made by the operator and the corresponding pickup time t_pu, is referred to as t_lock. At $t = t_\text{pu} - t_\text{lock}$, users are informed about their exact pickup times and the vehicles that are assigned to them, because the assignments are fixed and will not be subject to changes due to global optimizations anymore. This means, the longer t_lock, the earlier users can plan when to leave the house or otherwise schedule their remaining time until pickup, while being certain when exactly they will be picked up. Such services are perceived better than options in which assignments are fixed very late or not at all, resulting in uncertainty and stress for users being informed on short notice. On the other hand, the longer the lock time, the smaller is the optimization potential, because requests are less often part of global optimization, limiting the number of times a potentially better assignment can be found.

Similarly, the length of the pickup time window directly affects the perceived user experience and the size of the solution space during global optimizations. The larger this parameter t_twl, the more uncertain it is for service users when they will be picked up, which again translates to a worse perceived service quality. Shorter time windows allow users a more precise short-term planning immediately after the initial response is received. However, a shorter t_twl also means assignments tend to include pickup times closer to the pickup times associated with the originally found assignment based on the heuristic that defined the pickup time window in the first place, even though solutions with individual pickup assignments outside of the associated time window would result in globally better solutions.

On top of the variables evaluated in the PSA of the 2-step service model, an additional parameter investigated in this work is the maximum user waiting time for pickups t_{max}. This parameter marks the longest time a service user is willing to wait between submitting the request and being picked up. In all considered models, the operator of the service is assumed to be aware of this threshold and the fact that no offer that implies a pickup later than t_{max} is accepted by a user. This simplification is reasonable, considering the capabilities of large-scale data analyses of real-world ODM services that allow service providers to identify correlations between user waiting times and accepted service offers. The assumed maximum waiting time depends on the customer model used. In the conventional customer model, $t_{\text{max}} = 450\,\text{s}$ or seven and a half minutes. Otherwise, t_{max} ranges between 450 s and 900 s (15 min).

3.4.3 Key Performance Indicators

A precise and meaningful evaluation of system performances is crucial to understand and quantitatively measure the differences between service models and varying parameter sets. Because there is a great number of measurable statistics in agent-based simulations, it is not trivial to determine the most important ones to assess the system performance. Also, multiple perspectives need to be taken into account when evaluating the quality of an ODM service.

The service provider aims to maximize profit, hence maximizing the number of requests served with a fleet that is as small as possible. The fleet operator is also interested in low requirements for computational resources for the management and simulation of service models. First and foremost because a quick optimization of assignments and hence short response times for users are desirable for a high-quality service, but also because hardware necessary to operate large ODM fleets is an expense not to be underestimated. An additional and often one of the biggest expense factors is the mileage driven by the vehicles of the ODM fleet, because the energy needed to move them needs to be paid for.

Because less driven mileage by the ODM fleet means less traffic in the business area the service is offered in, political decision makers are interested in keeping this number low, too. More specifically, the added kilometers driven due to the ODM service are expected to be minimized, meaning the difference between the total driven distance of all vehicles of the ODM fleet and the sum of all trip distances requested by service users. If that value is positive, e.g., due to empty mileage driven by the service vehicles on their way to pickup locations, the net traffic in the streets of the city the service is offered in is increased. Such services might still be judged positively in general by policy makers, because the total number of vehicles necessary to make the trips is drastically reduced due to the high utilization of ODM fleets compared to privately-owned cars.

The third perspective that needs to be considered is the service user and the perceived service quality. If a service provider is not able to make offers that meet the expectations of customers, the service is predestined to fail in the long term, because potential users will look for alternatives. Individual requirements for the service are very divers between customers.

However, the qualities most commonly asked for are short waiting times, both for request responses and pickups, and reliability in the sense that if an offer is made, the service will be provided at the stated conditions. Obviously, service users also expect to be offered a service when requesting one, at least in most of the instances. In this point, interests of service providers, policy makers and users align.

Derived from these various service quality measures, the following list of key performance indicators (KPIs) is identified and used for the ride hailing use case throughout the remainder of this work:

- *Profit generated*, measured in US-Dollar,
- *Requests served*, measured in percent of the total number of requests made,
- *Computation time*, measured in seconds,
- *Distance driven emptily* by the fleet, measured in percent of the total driven distance,
- *Added distance* due to service, measured in percent of the total direct distances of the requested user trips,
- *Average user waiting time* for pickups, measured in seconds,
- *Pickup-in-time-window rate*, measured in percent of all pickups made.

The profit generated by a service is the difference of the revenue and the costs associated with it. The revenue is composed of a service base fare p_{base} payed by all served customers R_{s} and a distance-related fare p_{dist}, which increases with the direct trip distances $d(z_{\mathsf{pu},j}, z_{\mathsf{do},j})$ of served users $j \in J_{\mathsf{s}}$. On the flip side, the total costs of a service is the sum of the fixed vehicle costs c_{fix}, which equally applies for all service vehicles M, independent of the usage of each vehicle during the simulations, as well as costs c_{dist} related to the driven distance D_i of each vehicle $i \in I$, normally due to some sort of fuel or consumed energy needed to move. The generated profit therefore equates to the objective function

$$F_{\mathsf{obj}} = R_{\mathsf{s}}\, p_{\mathsf{base}} + \sum_{j}^{J_{\mathsf{s}}} \left(d(z_{\mathsf{pu},j}, z_{\mathsf{do},j})\, p_{\mathsf{dist}} \right) - V c_{\mathsf{fix}} - \sum_{i}^{I} \left(D_i\, c_{\mathsf{dist}} \right). \tag{3.11}$$

The percentage of requests served by the evaluated ODM service is an indicator of what fraction of users that made a service request have been picked up and transported to their destination. This number is not identical with the percentage of served passengers, since the number of passengers associated with a request is variable. However, in the ride hailing use case, the difference between these two KPIs is purely random, because the maximum group size for each request is set to the capacity of the vehicles in the fleet and each request is served one-by-one, meaning there is no packing of multiple requests within a vehicle and therefore no tendency to reject requests with more passengers involved. Using the customer model of this section, the percentage of users served is identical with the percentage of users that received a service offer, though, because all requests that can be picked up within the respective maximum waiting time receive an offer, accept this offer and are served by the ODM service vehicles.

Computation time is measured in seconds and reflects the processing time of the complete simulation, including optimization (heuristic and global), fleet management (routing, repositioning), demand data operations (reading, processing) and evaluation (saving of statistics during simulation and calculation of KPIs at the end).

Since the distance driven by vehicles of the ODM fleet is crucial for the evaluation of the service for both the provider and public decision makers in the business area, this KPI needs to be evaluated thoroughly. The overall mileage driven can be split into several parts. Distances driven with customers on board are desirable for all parties, while empty mileage needs to be avoided, because it increases the service costs and the traffic in the business area. The percentage of empty mileage can be further divided into distances traveled directly to pickup locations and trips made based on repositioning assignments.

In addition to the empty distance driven by service vehicles relative to the overall mileage, another metric to measure unwanted trip lengths is the added amount of mileage that is caused by offering the service relative to a scenario in which all requested trips would have been made with privately-owned cars instead. This KPI is a strong indicator of how an ODM service affects the traffic in the business area, besides the benefit of reducing the number of vehicles needed to provide the requested trips.

When it comes to the most important KPIs from a service user's perspective, the average waiting time between request and pickup is among the ones that is most decisive if a service is considered to be good or not. As described in Section 3.1, the service models evaluated in this work differ in their respective response times and the moment, when vehicles are assigned to new requests and therefore begin their trip to the pickup location. Later response times therefore directly translate to longer pickup waiting times in these service models.

Besides waiting times, customers appreciate services which fulfill what they offer, meaning they do not want to be surprised by pickups happening outside of a communicated time window or – even worse – late rejections after they initially had been accepted. The constraints of the global optimization used in this work forbid such late rejections. However, pickups outside of associated time windows are possible. As stated in Equation 3.1, solutions that imply such assignments are penalized, which means in order for them to be considered better than solutions only including pickups in time by the optimization algorithm, they need to be considerably better in terms of other objectives. The percentage of requests that is served within their respective pickup time windows is referred to as "pickup-in-time-window rate".

All of these KPIs are compared between service models and in PSAs in simulations conducted for each of the seven simulation dates and five fleet sizes considered. The values of some KPIs vary tremendously with fleet sizes as well as between the considered simulation dates due to the significant differences in demand shown in Figure 3.5. Hence, in the upcoming sections, results are depicted in specific ways for each KPI side-by-side.

Evaluation of Key Performance Indicators in Relation to Fleet Sizes

When evaluating the system performances in relation to fleet sizes, the total mean value of each KPI averaged over all simulation dates considered is presented on the left. Because of the large range of values, however, it is hard to make out differences between the scenarios compared in the figures for some of the KPIs. In order to facilitate these direct comparisons, there is another metric presented as well. Instead of depicting the total values measured for all KPIs, these figures show the average difference of each individual KPI in a certain scenario to the mean value of all KPIs considered in the figure in this scenario.

For example, consider an arbitrary KPI X in a simulation scenario with 100 vehicles. This KPI is measured in simulations of all dates a considered, so there is an X_a for each simulation date. Further assume three service models to be compared with each other, M_1, M_2 and M_3. In a first step, the average $X_{\text{avg},a}$ of the three values of X_a for M_1, M_2 and M_3 is calculated for each simulation date a. Then the difference of each individual value of X_a to $X_{\text{avg},a}$ is calculated, resulting in a value for ΔX_a for each service model. This procedure is repeated for all simulation dates a considered and the mean value of all ΔX_a is determined, from hereon referred to as "delta to average". Note that the delta to average of X measured using service model M_1 might vary between days, even if the value of X remains constant, because it is calculated relative to the values of the same KPI achieved with other models M_2 and M_3, as described. The whole process is carried out for all fleet sizes, resulting in the graphs seen on the right of figures depicting KPIs over fleet sizes.

The delta to average is suited to provide insight into not only the differences of various service models due to the smaller scale of values to compare. It also allows to examine the variance in performances of service models, more so than an average of KPIs over simulation dates directly would be capable of. The reason for this is that the variation of performances over the simulation of different days is much larger than over service models due to the variance in demand, shown in Figure 3.5.

Evaluation of Key Performance Indicators in Parameter Sensitivity Analyses

The same reason applies to the decision how to depict results in the PSAs conducted in Chapters 4 and 5. Since the fleet size is fixed to a specific value, the variation in total values of KPIs in a specific parameter setup solely comes from the varying demand during the simulation dates. On the left of each figure presented in the PSA section, the respective KPI is shown for all values of the evaluated parameter and each simulation day in varying colors and slightly faded. The average over all simulation dates is presented as bold, black line. The same line is presented separately at the right of each figure, which allows the identification of slighter variations related to the evaluated system parameter.

In Sections 6.3.2 and 6.4.2, parameters of the diffusion customer model are examined in the 2-step service model. The variations of the average values of KPIs in these PSAs are large enough to be investigated without singling out the average in an extra plot. Hence, each KPI is depicted with respect to first offers made by the service operator on the left of these figures, on the right the same KPI is presented in relation to second offers. This allows to compare these two and identify differences more easily.

Please note that in addition to the results presented in the next chapters, the Appendix provides additional insights into the KPIs described above.

Chapter 4

Ride Hailing Use Case

The first ODM use case in which the different service models are tested is ride hailing. This transportation mode guarantees customers to be served individually, without the option for rides to be shared. Thus, user trips between pickup and drop-off locations are made on the shortest route, ensuring travel times similar to trips made with privately-owned vehicles. Ride hailing services are therefore very convenient for customers and reduce the overall number of vehicles necessary to serve the mobility demand in densely populated urban areas. On the other hand, this transportation mode potentially implies added traffic within the business area, because in addition to the distances driven during user trips, mileage is produced during trips that are made emptily to pickup locations.

This chapter introduces the methods used to assign vehicles to ride hailing service requests and presents results of the evaluation of service models and model parameters in this use case.

4.1 Assignment Approaches

The methods used to assign vehicles to service requests are crucial for the system performance of ride hailing services. In the following section, a short description is presented of both the heuristic method used for immediate responses in Service Models 1 and 2 as well as the exact optimization approach applied in Service Models 2 and 3.

Nearest Neighbor Policy

The nearest neighbor policy (NNP) is a heuristic designed to find feasible assignments very quickly. Its concept is simple: for every new user request, the trip durations from the service vehicles' next idle locations to the pickup location of the new user are calculated and the earliest possible pickup time t_{pu} is derived. Because the new request is the only one taken into account, the assignment to the respective vehicle is consequently the best solution possible in terms of the control function F_{con} introduced in Section 3.2.

If $t_{\text{pu}} \leq t_{\text{max}}$, the user request is accepted and a service offer is sent to the customer. In the service model that solely depends on this heuristic procedure, this offer includes the exact pickup time t_{pu} and the assigned vehicle, because both are fixed in this service model, as described in Section 3.1. In the 2-step service model, the offer comprises a time window during which the pickup is expected to take place, which is preliminary, because later reassignments are possible.

The greatest advantage of the NNP is its simplicity, not only allowing this heuristic to easily be applied to most formulations of the VRP, but also to find feasible solutions to the respective problems very quickly. The computation time needed to calculate the pickup times and to find the vehicle that is able to serve the request the soonest scales linearly with the problem size. The NNP is therefore used for finding assignments in many real-world services and often used as a reference in the research area of transportation, as shown in Section 2.2.

When it comes to optimality, however, like other heuristics, the NNP is unable to reliably produce globally optimal solutions to the dynamic DaRP. The more service users need to be served and the higher the ratio of requests to vehicles, the smaller is the likelihood of the NNP to produce solutions anywhere close the global optimum.

Global Optimization Method and Constraints

Unlike heuristics in the ride hailing service models considered in this work, periodic optimization is capable of globally improving the solution quality by taking into account multiple requests and reassigning vehicles if beneficial.

At the end of each optimization period t_{o}, the simulations conducted with Service Models 2 and 3 run into global optimization as illustrated in Figure 3.6. The first step of this process is to find all feasible pairs of vehicles and unlocked requests that could potentially be assigned to each other. In order to identify these pairs, for each request, that has not yet been locked to a service vehicle, an algorithm determines the user waiting time implied by an assignment to every vehicle. If the respective waiting time is shorter than the maximum waiting time of the user, the vehicle-request pair is added to a list of potential assignments.

This search is sped up by considering only one unlocked assignment per vehicle. This simplifying constraint allows to substantially prune the solution space of the global optimization without reducing the optimization potential significantly. That is because assignments of more than one unlocked request in the task queue of a vehicle are assumed to rarely be part of an optimal solution, considering the additional waiting time for users that are picked up after another unlocked request due to the implied detour and boarding times.

In a next step, for all the vehicle-request pairs in the list of potential assignments, the control function value is calculated according to Equation 3.1. After that, the optimization problem is formulated and solved using CPLEX, a software created by IBM ([IBM ILOG CPLEX, 2017]), which is able to reliably find the global optimum of such a problem.

Finally, the new request assignments found during the global optimization are added to the task queues of the respective vehicles, replacing former unlocked requests in them (which are reassigned to other vehicles themselves) if there are any. Reassigned users are not yet notified, though. Only when the pickup by their currently assigned vehicle happens within t_{lock}, users receive the exact pickup time and vehicle ID. This is to avoid confusion and annoyance on the side of customers due to continuous notifications sent after every optimization period, which potentially decreases the perceived service quality rather than improving it because of the transparency.

4.2 Results

In this section, the service models for the ride hailing use case described in Section 3.1 are evaluated in terms of the KPIs introduced in Section 3.4.3. After the comparison of the service models, a PSA is conducted for the two-step service model.

4.2.1 Evaluation of Service Models

The three service models evaluated in this section vary in their respective approach to manage user requests, as described in Section 3.1. The service model illustrated in Figure 3.1 only relies on heuristic methods and does not include any global optimization. It is therefore referred to as Service Model 1, "no global optimization". It is further depicted as red lines in the figures of this section.

Service Model 2 uses both heuristic and exact optimization techniques in order to provide an immediate response to user requests while being able to use the global optimization potential. This model is referred to as "2-step service" and colored black in the upcoming figures.

The third and final service model considered in this section does not make use of any heuristic methods, but instead uses global optimization when it comes to the decision if a user request is accepted or not. Service Model 3 is hence labelled as "only global optimization" in the comparative graphs of this section. Its associated color is blue.

The evaluation of service models is conducted by comparisons of the KPIs described in Section 3.4.3 in relation to ODM fleet sizes of 100 to 500 vehicles. The parameter sets used in the simulations are described in Section 3.4.2. The variable parameters are set to their respective standard values.

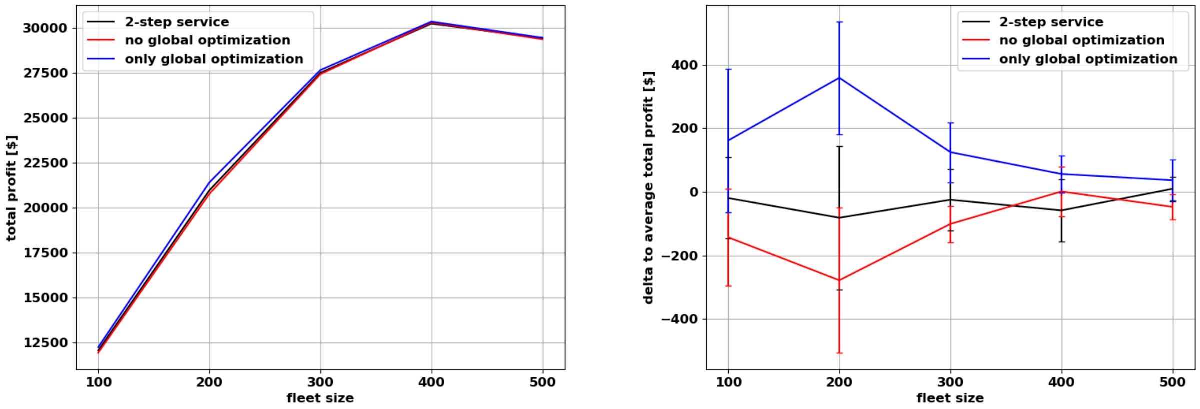


Figure 4.1: Profit in US-Dollars generated with various service models in the ride hailing use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

Figure 4.1 presents the average daily profit in US-Dollars ($) generated by each of the service models on its left, as well as the delta to average as explained in Section 3.4.3 on its right. The maximum profit of approximately $30 000 is generated with fleet sizes of 400 vehicles in all service models, implying that the added revenue of additional vehicles in larger scenarios cannot make up for the costs caused by idle vehicles. The delta to average reveals that Service Model 3 outperforms the others by a margin of $(47 ± 102) in scenarios with fleet sizes of 500 vehicles to $(637 ± 404) in simulations with 200 vehicles. In relation to the overall profit, that is a gain of up to 3 % compared to Service Model 1. For all fleet sizes except 400 vehicles, Service Model 2 generates slightly more profit than Service Model 1, albeit within the standard deviation indicated by the error bars. In general, the differences of generated profit between the service models as well as the standard deviations of individual results are greater in scenarios with smaller fleet sizes. With larger fleet sizes, it becomes more and more likely that assigning the closest vehicle to a user request is globally the best solution, mitigating the gap between the service models.

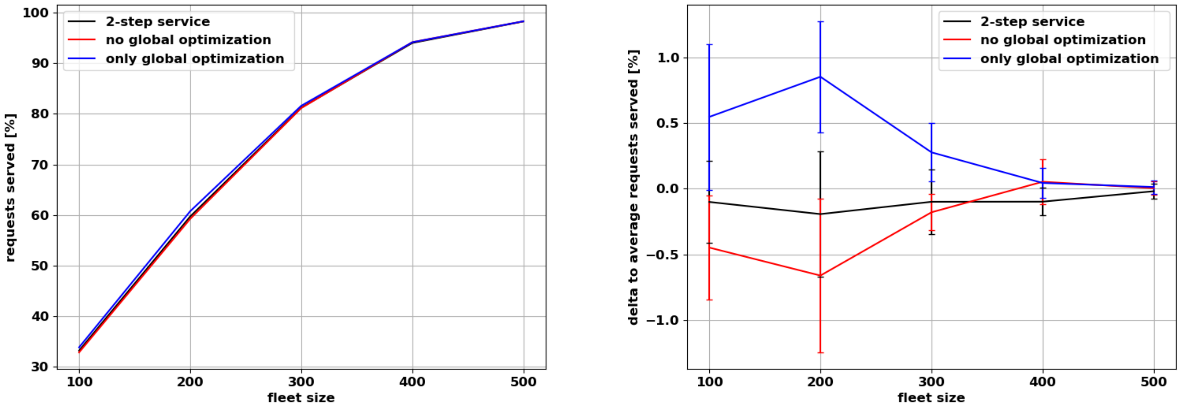


Figure 4.2: Percentage of requests served with various service models in the ride hailing use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

The percentages of requests served during the simulation of the service models shows the direct relation between the number of paying customers and the profitability of an ODM service. Figure 4.2 shows a very similar disparity of this KPI between the considered service models compared to the profit. Again, the differences are clearer in simulation scenarios with smaller fleet sizes for the same reasons explained above. The percentage of served requests ranges from 32.9 % in simulations of Service Model 1 with 100 vehicles to 98.3 % in 500-vehicle scenarios of all three service models. In scenarios with fleet sizes of 300 vehicles or less, Service Model 3 outperforms both of the service models depending on heuristics. In these instances, the benefit of the option for the operator to select the most suitable requests to be served according to the periodically conducted global optimization outweighs the drawback of later average pickup times due to the increased response time of this service model, at least from the operator's perspective. The higher the percentage of served user requests, the less important this advantage becomes, until the number of requests that can be served due to quicker response times by Service Models 1 and 2 negate it completely, which can be observed in simulations with 400 and 500 vehicles.

The computation times associated with each of the three service models is presented in

Figure 4.3. Independent of the service model, the computation times increase with the fleet size, as anticipated. Simulations with Service Model 1 clearly take the shortest computation times in scenarios with all considered fleet sizes. The gap to Service Models 2 and 3 grows with increasing fleet sizes, which indicates the increasing complexity of the global optimization in scenarios with larger fleets. While in scenarios with 100 vehicles, the average computation times of simulations with Service Models 2 and 3 are indistinguishable and around 150 s longer than the computation time of simulations with Service Model 1, the gap between the service models that include global optimization also grows with the fleet size. In the scenarios with the largest fleets considered, simulations with Service Model 1 last around 2230 s (ca. 37 min) on average, simulations with Service Model 2 approximately 2720 s (ca. 45 min) and simulations with Service Model 3 2820 s (ca. 47 min).

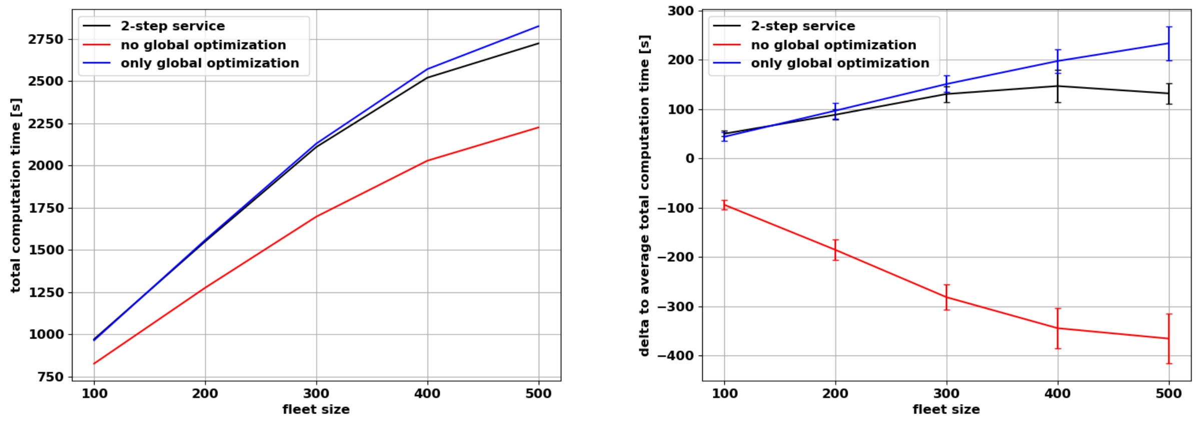
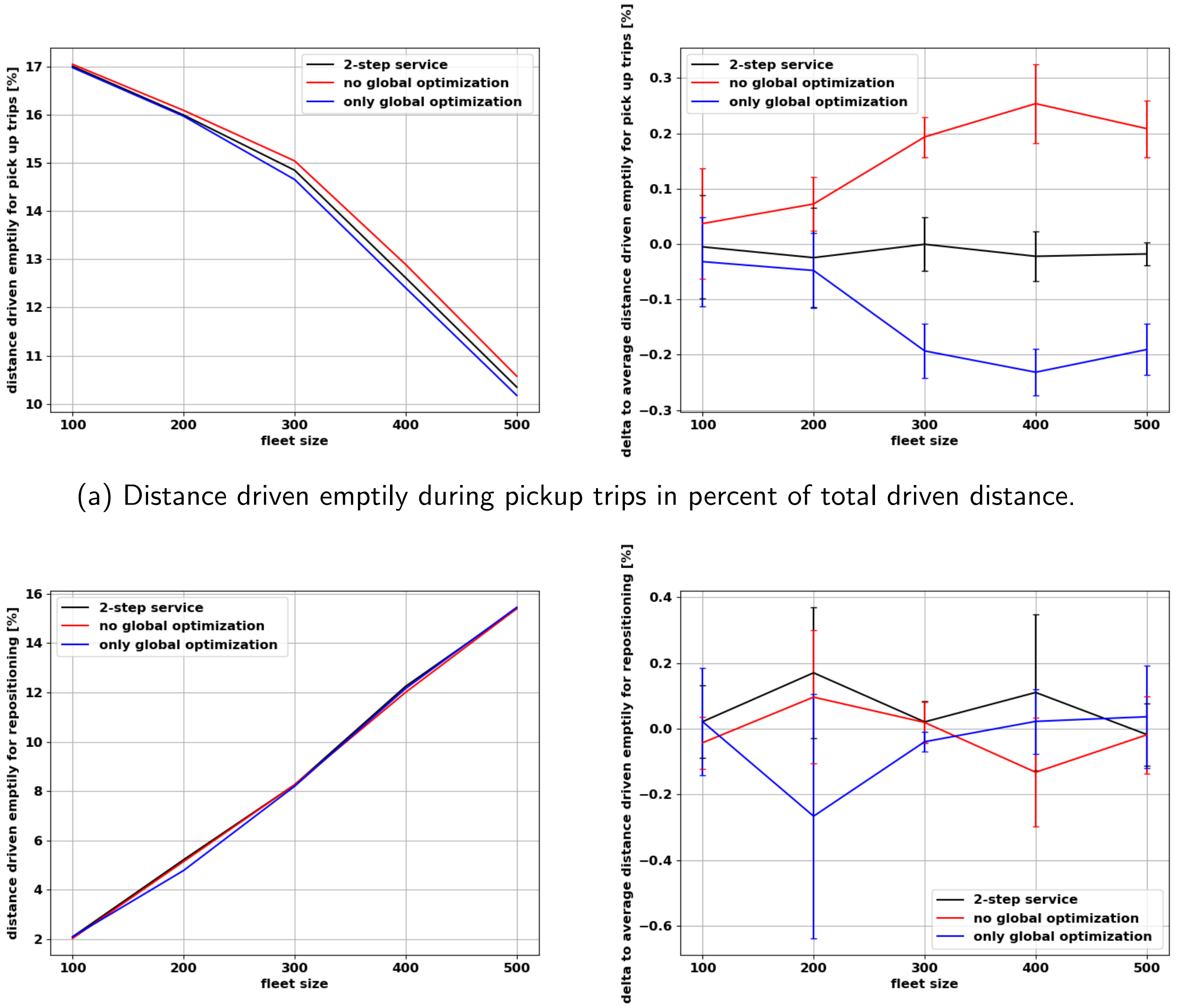


Figure 4.3: Computation times in seconds in simulations of various service models in the ride hailing use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

In Figure 4.4, the percentages of distances driven without a user on board relative to the total distances driven by all vehicles in the fleet are presented. Two categories of empty trips are shown: the distances driven during trips to pickup locations (Figure 4.4a) and the empty mileage due to repositioning trips (Figure 4.4b). In all service models, the total percentage of empty mileage increases with growing fleet sizes, from around 19 % in simulations with 100 vehicles to approximately 26 % in instances with 500 vehicles, even though the average trip lengths to pickup locations decrease in scenarios with more service vehicles, dropping from around 17 % in 100-vehicle scenarios to below 11 % in scenarios with the largest fleets considered. The reason for this is the growing potential for repositioning in simulations with more idle vehicles, which can be observed in Figure 4.4b. While only around 2 % of all trips made in scenarios with 100 vehicles are due to repositioning, this number rises to more than 15 % in simulations with 500 vehicles.

The relative differences between the service models are presented in the figures on the right, showing the delta to average. While the empty distances due to repositioning are very similar for all considered fleet sizes, the distances driven during pickup trips vary considerably, depending on the service model used during the respective simulation. Once more, Service Model 3



(a) Distance driven emptily during pickup trips in percent of total driven distance.

(b) Distance driven emptily during repositioning trips in percent of total driven distance.

Figure 4.4: Distances driven emptily in percent of total mileage for various service models in the ride hailing use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

performs better than both of the other models, especially in simulations with larger fleet sizes. However, there is also a clear gap between Service Model 1 and 2: while the percentage of empty mileage is up to $(0.25 \pm 0.09)\,\%$ higher in simulations of the 2-step service model compared to the service model only depending on global optimization, this value is topped in simulation scenarios using only heuristic assignments, adding another $(0.28 \pm 0.12)\,\%$.

This discrepancy can be explained by the fact, that the global optimization used to reoptimize the initial heuristic assignments in the 2-step service model is capable of reducing the overall distances of pickup trips to requests, while the decision which user requests are served is made by the heuristic. In Service Model 3, this selection is made periodically, allowing the operator to accept the most suitable requests. Additionally, idle vehicles only start their pickup trip to a new request after the first periodic optimization is executed, which further decreases the empty mileage due to avoided detours driven to initially assigned pickup locations, that are reassigned during the global optimization.

It is worth mentioning here, that the relative amount of emptily driven mileage is a pivotal KPI for service providers and operators. However, in order to evaluate the impact of the service on the traffic in the respective business area, the measure changes in a slight yet important way. Instead of measuring the percentage of empty distance relative to the total distance driven by the fleet, the more interesting question for public decision makers is "how much more mileage is produced due to the service being offered compared to the direct trip distances of the service users"? Or in other words, if all customers served by the ODM service would have used a privately-owned vehicle and directly travel from their origin to their destination, how much less empty mileage would have been produced? In the ride hailing use case, this can easily be answered by simply inverting the fraction of trips being made with customers on board, because each user is served individually and the direct travel distance is equivalent with the distance driven with customers on board.

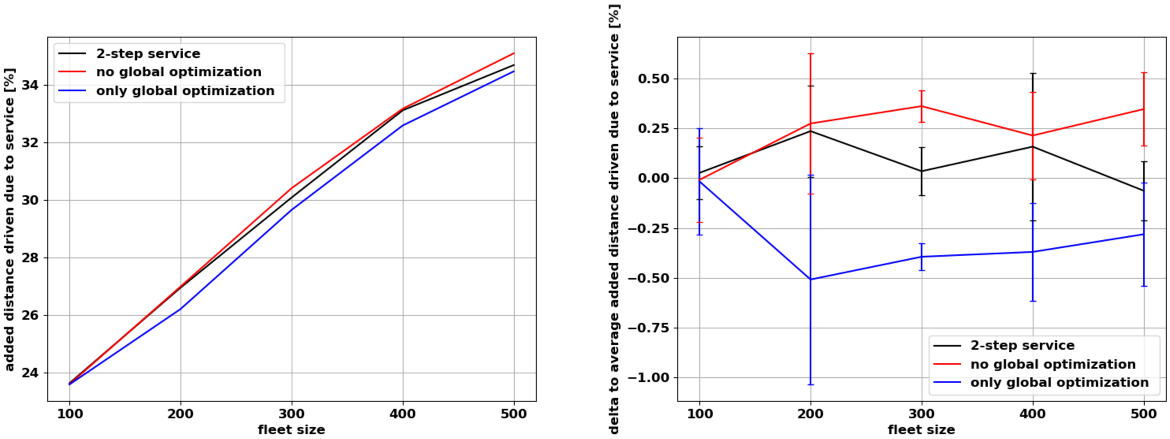


Figure 4.5: Added distances driven due to various service models in the ride hailing use case in percent of total distance of direct user trips with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

Hence, the added distance due to the ride hailing service models evaluated in this section ranges from (23.6 ± 0.3) % in simulations using either of the three service models with 100 vehicles to between (34.5 ± 0.2) % (Service Model 3) to (35.1 ± 0.3) % (Service Model 1) in simulations with fleet sizes of 500 vehicles as depicted in Figure 4.5.

In the evaluation of KPIs from service providers' and city officials' points of view, like service profitability and produced empty mileage, the service model only using global optimization outperforms the other two service models considered in the ride hailing use case. When it comes to the service quality indicators from a user's perspective, though, the picture changes. Figure 4.6 depicts the average waiting times of served customers between requesting a service and being picked up. On the left-hand side, the total values show that the average waiting times decrease with increasing fleet sizes. In scenarios with 100 vehicles, customers are served after 337 s (Service Model 2) to 339 s (Service Model 3). In the 500-vehicle scenarios, the average waiting times are between 134 s and 147 s. In Service Model 2 the waiting times are

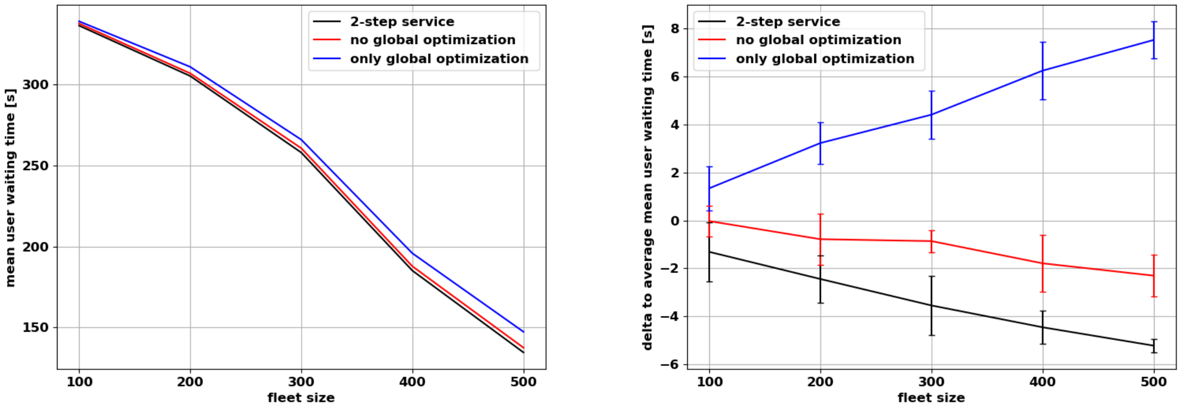


Figure 4.6: User waiting times in seconds with various service models in the ride hailing use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

lowest in simulations with all considered fleet sizes, followed by Service Model 1 and Service Model 3, which can be observed in the delta to average on the right. This figure further reveals the differences between the service models with increasing fleet sizes. While the gap between Service Models 1 and 2 is rather stable between (1.3 ± 1.9) s and (2.9 ± 1.2) s in simulations with fleet sizes from 100 to 500 vehicles, the discrepancy to user waiting times in Service Model 3 clearly grows with increasing fleet sizes. In simulations with 100 vehicles, the 2-step service model serves customers (2.7 ± 2.1) s faster on average than the service model only using global optimization. This difference increases steadily to up to (12.8 ± 1.1) s in simulations with 500 vehicles.

This observation can be explained with the delayed response implied by Service Model 3, which also delays the start of idle vehicles towards the pickup locations of new requests. The more vehicles present, the more likely it is that an idle vehicle initially assigned by the nearest neighbor heuristic is also globally the optimal solution. In Service Model 3, however, there is no heuristic making the initial decision to send a vehicle immediately, instead this decision is made in the next global optimization taking place after another $\frac{t_\text{o}}{2}$ seconds on average. Therefore, the difference in average user waiting time between Service Models 1 and 2 compared to Service Model 3 seems to converge to this value, which is 15 s in the simulations considered in this section because $t_\text{o} = 30$ s in the standard parameter set.

Besides the average waiting time, the reliability of projected pickup times is another important performance indicator of service models when it comes to the received quality from a user's perspective. The KPI used to measure this is the pickup-in-time-window rate, as explained in Section 3.4.3. In Figure 4.7, the pickup-in-time-window rates of each service model is presented. On the left, one of the typical features of Service Model 1 can be observed: since there is no reoptimization of initial assignments, the pickup always happens at the time it was originally projected and communicated to the respective user. The pickup-in-time-window rate is therefore 100 % in all simulations with that service model.

In both of the other service models, assignments can be changed after the initial pickup

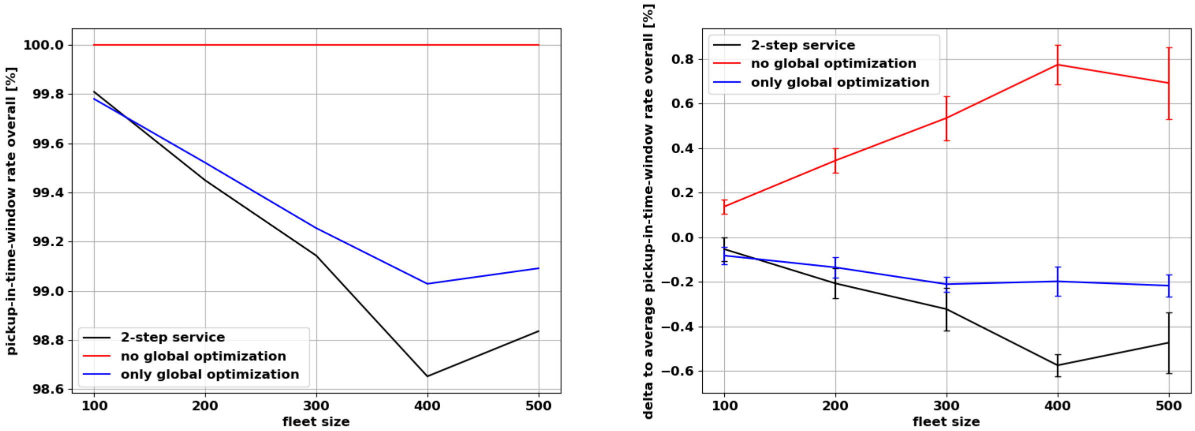


Figure 4.7: Pickup-in-time-window rate in percent for various service models in the ride hailing use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

time window is defined. Even though solutions with assignments outside of the associated pickup time window are penalized as described in Section 3.2, such solutions can be found to be globally better if driven distances or the total user waiting times are reduced, or – in the case of Service Model 3 – if more requests can be accepted. Hence, the pickup-in-time-window rate deviates from 100 %, as can be seen on the left of Figure 4.7. In Service Models 2 and 3, the rate of correctly predicted pickup time windows is highest in scenarios with small fleet sizes (approximately 99.8 % with 100 vehicles) and reaches its minimum in simulations with 400 vehicles, which coincides with the simulation scenarios in which the generated profit is maximized. However, in these instances, the 2-step service model performs slightly worse than the service model only using global optimization, achieving a pickup-in-time-window rate of below 98.7 % compared to over 99 % in Service Model 3.

The right-hand side provides further insight into the variations between the service models. Especially the standard deviation of the delta to average shows that even though the differences in total percentages seem small, the performance gaps of the three service models are significant in the sense that they are greater than the respective standard deviations, represented by the error bars, at least in simulations with larger fleet sizes. The error bars associated with the pickup-in-time-window rate of Service Model 1 reflect the fact that even though the variation of that KPI in simulations of this specific service model is zero, the delta to the average of all three service models varies for each simulation date, resulting in the shown standard deviation.

To summarize the comparison of Service Model 1 (“no global optimization”), Service Model 2 (“2-step service”) and Service Model 3 (“only global optimization”) in the ride hailing use case, the main results are the following:

- The profit generated with Service Model 3 is up to 3 % higher than in simulations of Service Model 1 and is higher than the profits generated with each of the other service models throughout simulations with all considered fleet sizes.
- The maximum daily profit of around $30 000 is reached in scenarios with 400 vehicles,

in which the differences between the three service models is insignificant.

- Service Model 3 is capable of serving the most requests in scenarios with 300 vehicles or less. In simulation scenarios with larger fleets, the differences between the service models is negligible.

- The average computation time of simulations of Service Model 1 is the shortest in scenarios with all considered fleet sizes. With increasing fleet sizes, the computation times of simulations of all service models grow steadily.

- The total percentage of empty mileage produced by the various service models increases with growing fleet sizes. This is due to the steeply increasing distance driven during the repositioning of idle vehicles which outweighs the reduction in pickup trip distances due to more available vehicles.

- Between the service models considered, Service Model 3 produces the shortest emptily driven distance, followed by Service Model 2, which performs better than Service Model 1, especially in saving mileage during pickup trips. The gaps in empty mileages due to pickup trips between the service models grows with increasing fleet sizes.

- The average waiting times drop with increasing fleet sizes, ranging from almost 340 s to below 150 s in simulation scenarios with 100 to 500 vehicles respectively.

- Service Models 1 and 2 serve customers significantly faster than Service Model 3, while this gap widens with increasing fleet sizes. Service Model 2 outperforms Service Model 1 by up to (2.9 ± 1.2) s and Service Model 3 by up to (12.8 ± 1.1) s in simulations with 500 vehicles.

- While the pickup-in-time-window rate of Service Model 1 is 100 % by design, in both of the other models, the minimum rate is reached in scenarios with 400 vehicles in which the most profit is generated. Service Model 2 achieves a pickup-in-time-window rate of approximately 98.7 % in these instances, while Service Model 3 never drops below 99.0 %.

4.2.2 Evaluation of 2-Step Service Model Parameters

After evaluating three different versions of ride hailing service models, the following section focuses on the 2-step service model. A PSA is conducted, providing insight into dependencies of certain model parameters. This section focuses on the most important and most sensitive KPIs for each evaluated parameter. Besides each investigated parameter, the standard parameter set is used. The shown simulation scenarios are run with a fleet size of 300 vehicles. As described in Section 3.4.2, the figures presented in this section include an overview of the results of each individual simulation date to provide insight into the variance of the respective KPI in scenarios with varying demand, as well as the average of all simulation dates for each value of the evaluated parameter. This average is also presented separately, which allows to identify variations in direct relation to the evaluated parameter values in more detail.

Optimization Period

The first parameter of the 2-step service model examined in more detail is the optimization period t_o, which defines the intervals in which the global optimization is executed. The standard value of t_o is set to 30 s in the simulations runs during the prior section. In order to test the impact of shorter and longer periods between global optimizations, the range of evaluated optimization periods is set from 10 s to 60 s in steps of 10 s.

As the profitability of a service is crucial for providers, the generated profit in relation to t_o is presented first. The optimization period (also referred to as "batch time" in literature) affects the performance and profitability of service models that respond to requests after the first global optimization is executed, as shown in [Syed, Kaltenhäuser, et al., 2019]. In the 2-step service model, however, the frequency of global optimization does not have a direct impact on the received service quality for users, because both offers communicated to the customer are sent independently of t_o.

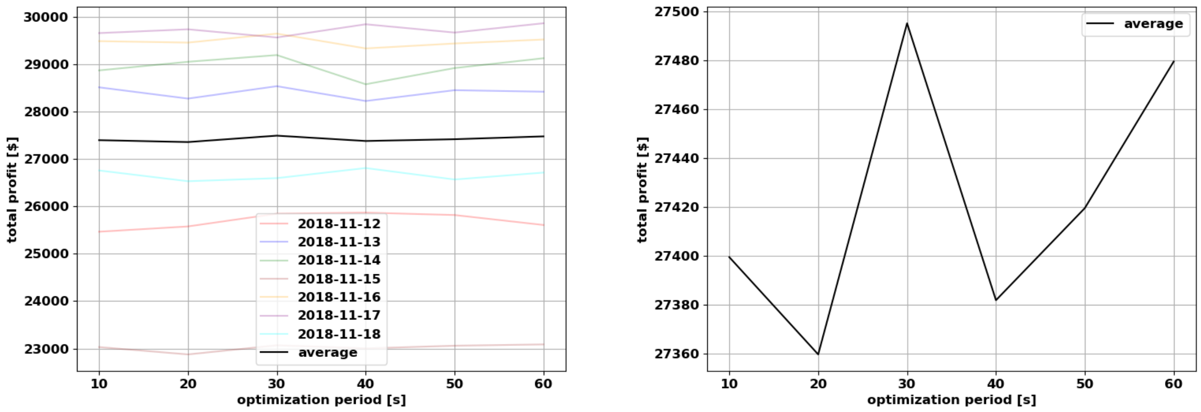


Figure 4.8: Profit in US-Dollars in simulations of the 2-step service model with 300 vehicles in the ride hailing use case with values of the optimization period t_o between 10 s and 60 s. Left: values for all simulation dates and the average value. Right: average value in detail.

Figure 4.8 shows the total profit generated in relation to varying values of t_o. Clearly, the variance between the different simulation dates is significantly higher than between the various optimization periods. That is because the demand during the considered simulation dates varies drastically (see Figure 3.5), which affects the potential profit for an ODM service immensely. Nevertheless, as can be seen on the right of Figure 4.8, the average profitability also changes with varying optimization periods. The standard value of 30 s is found to produce the best results ($27 495), which is 0.5 % more than the average profit generated with $t_o = 20$ s ($27 360). Overall, no clear correlation to t_o can be observed and the service's profitability is found to be rather insensitive to the optimization period used in the 2-step service model.

Even though the impact of different optimization periods for users of the 2-step service model is not as direct as in other service models, Figure 4.9 reveals a clear correlation between the pickup-in-time-window rate and t_o: the longer the intervals between global optimizations, the

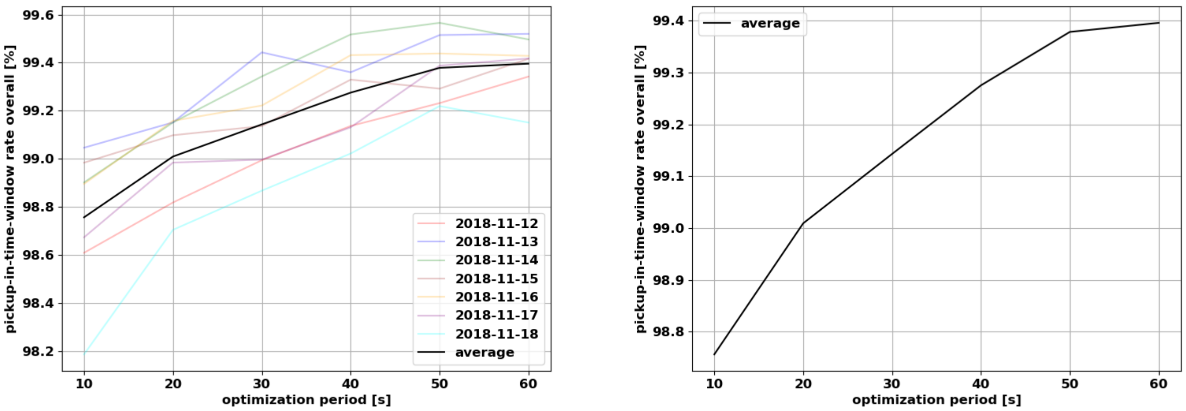


Figure 4.9: Pickup-in-time-window rate in percent of all accepted requests in simulations of the 2-step service model with 300 vehicles in the ride hailing use case with values of the optimization period t_o between 10 s and 60 s. Left: values for all simulation dates and the average value. Right: average value in detail.

higher the precision of pickup time window projections. The average pickup-in-time-window rate rises continuously from 98.76 % in simulations with $t_o = 10$ s to 99.40 % in instances in which $t_o = 60$ s. Part of the added accuracy can be explained by the fact that in scenarios with longer periods, the share of requests that is picked up before they become part of any global optimization rises with increasing t_o. Since such requests are guaranteed to be picked up within their respective time windows, this affects the total pickup-in-time-window rate.

Objective Weight of Distance

The next parameter analyzed in this section is the weight α given to the driven distance associated with a solution in the objective function presented in Equation 3.1. This parameter regulates the focus between the overall driven mileage and the total user waiting time that is caused by the assignments implied by a solution. In other words, it changes the optimization objective from focusing on minimizing the distances driven by service vehicles (high α) to the minimization of average user waiting times (low α). The value range in this PSA spans multiple magnitudes: the lowest α considered is $6 \times 10^{-4}\,\mathrm{m}^{-1}$, the highest value is $6 \times 10^{1}\,\mathrm{m}^{-1}$, while the standard parameter value is $\alpha = 6 \times 10^{-2}\,\mathrm{m}^{-1}$. To facilitate the readability of results, the x-axis is set to a logarithmic scale for this parameter.

The effective reduction of empty mileage presented in Figure 4.10 directly correlates with the emphasis on minimizing the total distance driven in the objective function. Because each assignments of a user to a service vehicle is rewarded with a very high value by the objective function, increasing α does not necessarily lead to fewer requests served in total and therefore no reduction in trip distances with customers on board. Instead, it results in a net reduction of total empty mileage due to saved mileage during pickup trips. The distance driven during repositioning trips, however, is not clearly correlated to α, as the repositioning algorithm is independent of this parameter.

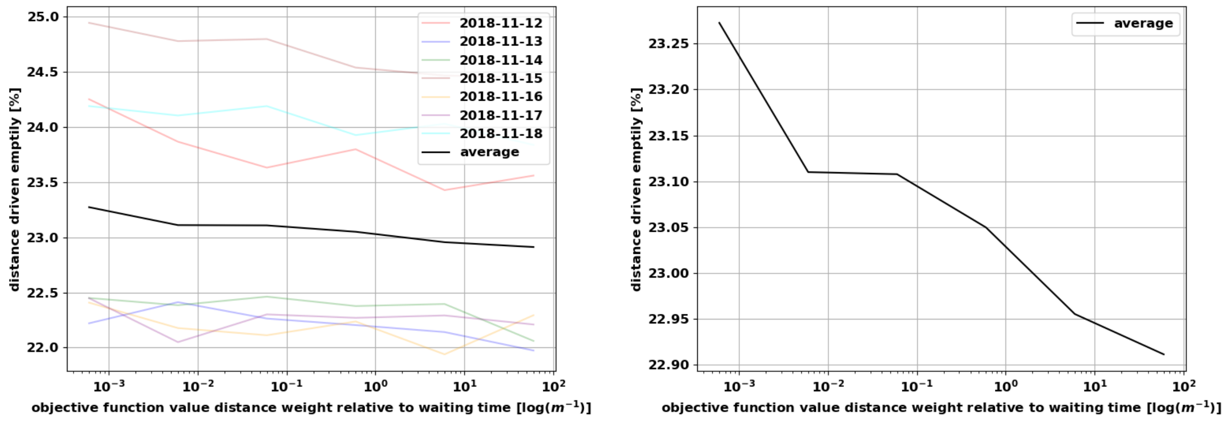


Figure 4.10: Distance driven emptily in percent of total driven distance in simulations of the 2-step service model with 300 vehicles in the ride hailing use case with values of the objective weight of distance α between $6 \times 10^{-4}\,\mathrm{m}^{-1}$ and $6 \times 10^{1}\,\mathrm{m}^{-1}$. Left: values for all simulation dates and the average value. Right: average value in detail.

With increasing α, intuitively, the average user waiting time is expected to increase as well, as the global optimization focuses more and more on saving mileage in favor of avoiding long waiting times for service users. Figure 4.11 shows that such a correlation cannot be observed in the simulations evaluated in this work, though. In fact, the longest average waiting times are measured in simulations in which α is set to $6 \times 10^{-3}\,\mathrm{m}^{-1}$ with an average value of 260.1 s. The shortest mean user waiting time of 258.1 s is reached in scenarios with $\alpha = 6 \times 10^{-2}\,\mathrm{m}^{-1}$.

These counter-intuitive results can be explained by considering the consequences of assignments that favor short waiting times over saved mileage: imagine two vehicles v_1, v_2 and two requests r_1, r_2 which can both be served within their respective maximum waiting time by both vehicles, even one after another. In this example, if r_2 would be picked up by v_1 after serving r_1, the waiting time of r_2 would be significantly higher than if v_2 is assigned, even though the total distance driven by both vehicles to fulfill their respective assignments would be longer in that case. Regardless, if α is set to a very low value relative to β, the latter solution would be considered best, because shorter waiting times would be prioritized higher than shorter travel distances, resulting in two vehicles being occupied for a time, which is in total longer than if v_1 would be assigned to both r_1 and r_2, which means later requests are served later. This example demonstrates how during the course of a 24 h-simulation, the average user waiting time is higher when using a parameter set originally designed to minimize it.

Assignment Lock Time

Unlike the optimization period and the objective weight of distance during global optimization, the assignment lock time t_{lock} directly affects the service experience of users. If t_{lock} is set to high values, customers receive a binding assignment confirmation earlier. Hence, the service offers a better predictability and improves the perceived service quality compared to models

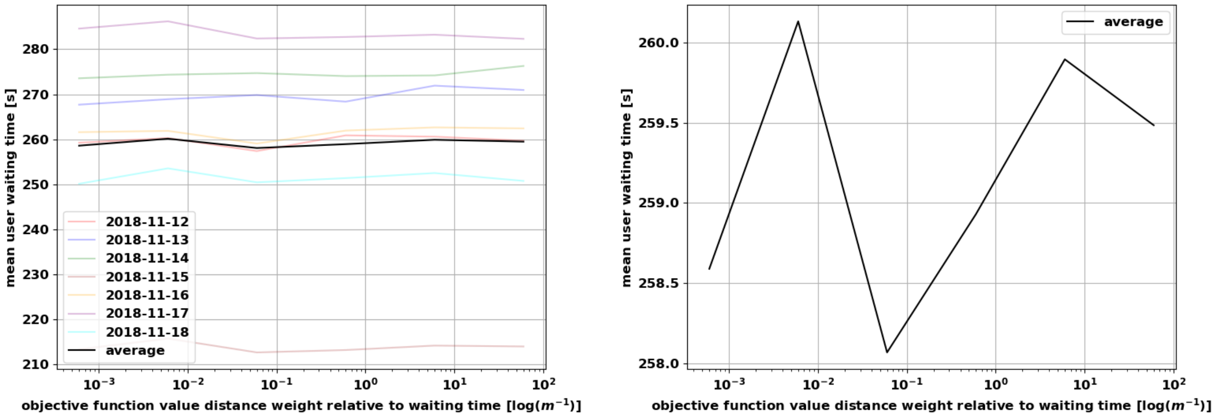


Figure 4.11: User waiting time in seconds in simulations of the 2-step service model with 300 vehicles in the ride hailing use case with values of the objective weight of distance α between $6 \times 10^{-4}\,\mathrm{m}^{-1}$ and $6 \times 10^{1}\,\mathrm{m}^{-1}$. Left: values for all simulation dates and the average value. Right: average value in detail.

with shorter t_{lock}, in which customers receive the details of their assignment – like the exact pickup time and vehicle ID – only on very short notice before the pickup takes place, potentially resulting in stress for the customer or delays in the pickup process. On the other hand, a shorter t_{lock} implies that requests are more often subject of global optimization, increasing the chance to potentially being reassigned to another vehicle in order to improve the overall system performance.

It is therefore not trivial for the service operator to choose suitable values for t_{lock}. In the following PSA, the assignment lock times range from 0 s, which means requests can potentially be reassigned until a pickup eventually takes place and customers are only notified when their assigned vehicle arrives at the pickup location, to up to 300 s (5 min). The standard value of t_{lock} is 120 s (2 min).

Once more, the generated profit is investigated first. As the delta to average reveals, the profitability of the 2-step service does not depend on t_{lock}. While the maximum profit is reached in the standard scenario with $t_{\text{lock}} = 120$ s (2 min), the worst system performance is produced in simulations in which t_{lock} is set to 180 s (3 min). In simulation scenarios in which $t_{\text{lock}} = 0$ s, the average profit is even lower than in scenarios with the longest assignment lock time of 300 s (5 min), which directly contradicts the assumption that the optimization potential decreases with higher values of t_{lock}.

Another KPI that is affected by varying assignment lock times is the pickup-in-time-window rate. Figure 4.13 illustrates that the accuracy of pickup time window projections is lowest (99.04 %) in simulations in which t_{lock} is set to 60 s (1 min) and rises with increasing assignment lock times to 99.71 %. There is also a small but consistent uptick of the pickup-in-time-window rate in simulations with $t_{\text{lock}} = 0$ s, implying that in these scenarios, a significant number of requests is reassigned within the last minute before the scheduled pickup in order to avoid the penalty in the objective function.

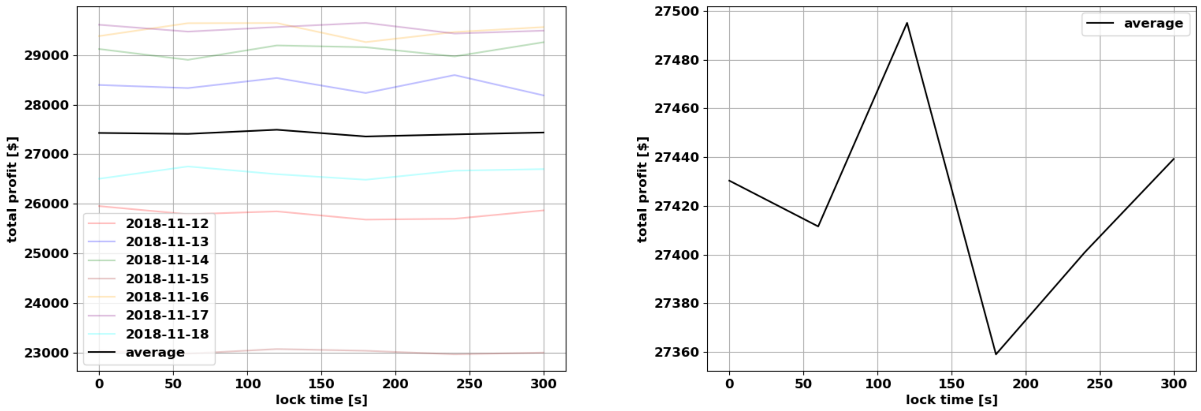


Figure 4.12: Profit in US-Dollars in simulations of the 2-step service model with 300 vehicles in the ride hailing use case with values of assignment lock time t_{lock} between 0 min and 5 min. Left: values for all simulation dates and the average value. Right: average value in detail.

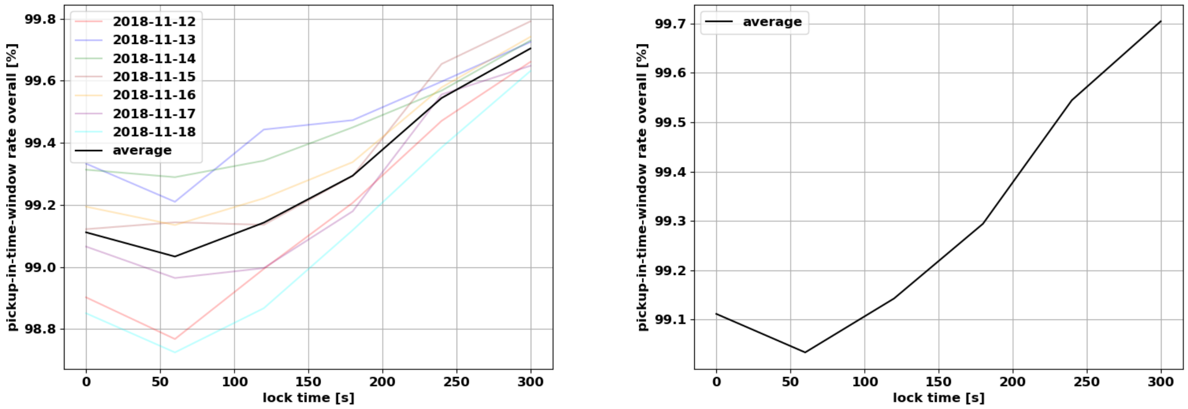


Figure 4.13: Pickup-in-time-window rate in percent in simulations of the 2-step service model with 300 vehicles in the ride hailing use case with values of assignment lock time t_{lock} between 0 min and 5 min. Left: values for all simulation dates and the average value. Right: average value in detail.

Pickup Time Window Length

The last parameter evaluated in this section is the pickup time window length t_{twl}. Similar to the assignment lock time, this parameter is an integral element of the perceived service for customers, which also affects the predictability of the service for users. A longer t_{twl} causes uncertainty on the side of customers about when exactly they are going to be picked up and hence when they will arrive at their respective destination. Shorter time windows, on the other hand, allow customers to be able to plan their time until pickup more precisely, e.g., when exactly to leave the house to avoid waiting outside in the rain or in the dark. The shorter the pickup time windows, though, the fewer opportunities for ODM operators to reassign requests without taking a penalty during global optimization, which potentially limits the optimization potential. The range of values of t_{twl} evaluated in this section spans from 0 s to 300 s (5 min), the standard value is 120 s (2 min).

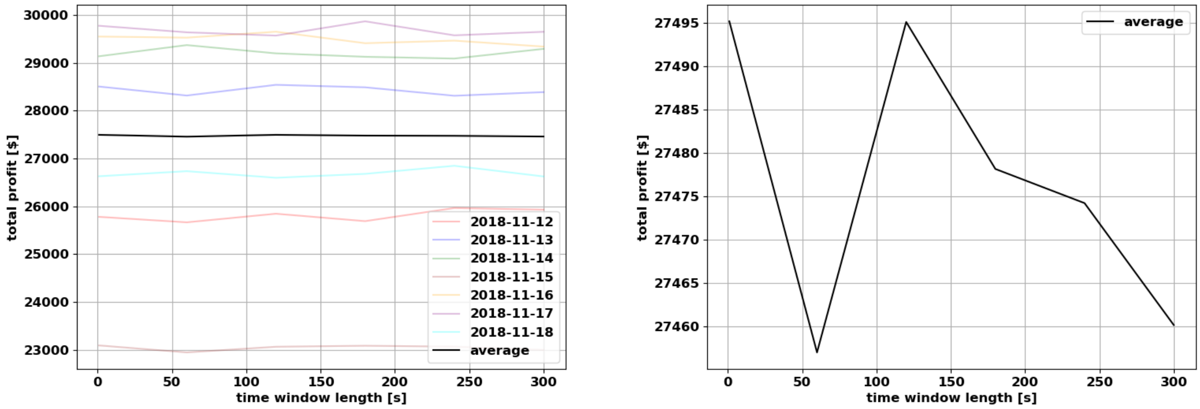


Figure 4.14: Profit in US-Dollars in simulations of the 2-step service model with 300 vehicles in the ride hailing use case with values of pickup time window lengths t_{twl} between 0 min and 5 min. Left: values for all simulation dates and the average value. Right: average value in detail.

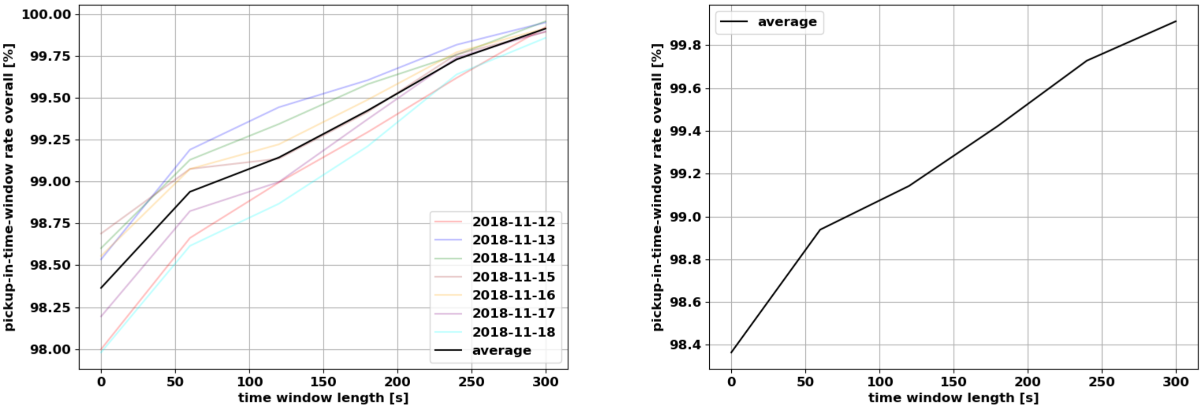


Figure 4.15: Pickup-in-time-window rate in percent of all accepted requests in simulations of the 2-step service model with 300 vehicles in the ride hailing use case with values of pickup time window lengths t_{twl} between 0 min and 5 min. Left: values for all simulation dates and the average value. Right: average value in detail.

Figure 4.14 shows the profit generated using varying values of t_{twl}. The scale of changes of the average value on the right is considerably smaller compared to the other PSAs, showing that the profitability of services with all considered values of t_{twl} ranges from 27 457 to 27 495, within a relative span of 0.1 %. Therefore, apparently distinct peaks and trends of the average profit in relation to the time window length need to be treated as only small variations, leading to the conclusion that t_{twl} does not have a significant impact on the profitability of a 2-step ride hailing service model.

A KPI that clearly correlates with t_{twl} is the pickup-in-time-window rate, which is depicted in Figure 4.15. This observation is rather a validation of the simulation and the service model than an actual finding. However, the percentage of correctly predicted pickup time windows even for small values of t_{twl} is worth noticing. Even in scenarios in which $t_{\text{twl}} = 0\,\text{s}$, the pickup-in-time-window rate never drops below 98 % in any simulation, implying that less frequent reassignments of requests do not negatively impact the system performance as a whole.

Chapter 5

Ride Pooling Use Case

In comparison to the ride hailing use case, in which each vehicle only serves one group of passengers at a time, the ride pooling use case increases the complexity of the assignment problem and the service model drastically. Between an individual's pickup and drop-off, other tasks can be assigned to vehicles, which potentially leads to detours. In theory, this allows service providers to increase the efficiency of their fleet and hence the profitability of their service. However, service users are not willing to make unlimited detours in favor of the provider's profit. Hence, it is reasonable to introduce the maximum detour time as a new model parameter and to measure the average user detour time in the evaluated scenarios as an additional system performance indicator.

Besides the added layer in the model's complexity, ride pooling also necessitates other methods to find optimized assignments of requests to vehicles. In the ride hailing use case, in which the user pickup is always succeeded by the same user's drop-off, the number of feasible combinations of pickup and drop-off tasks within one vehicle's task queue is relatively small, even if three or more requests are considered. The assignment approaches introduced in the former chapter make use of this constrained solution space to efficiently search for optimal assignments. However, in the ride pooling use case these limitations cease and the assignment techniques need to be adjusted.

This chapter describes the methods used to find initial assignments as well as the global optimization algorithm. It also introduces additional model parameters and indicators to evaluate the system performance of ride pooling services. In the concluding section, the results of three different ride pooling service models are presented, a parameter sensitivity analysis is conducted for the 2-step service model and the performances of ride hailing and ride pooling services are examined and compared.

5.1 Use Case Specific Performance Indicators and Parameters

One of the goals of this work is to compare the performances of ride hailing and ride pooling services within one simulation framework, using the same case study and model parameters. However, because of the intrinsic differences between the two use cases, there are aspects of the system performance that are difficult to compare, if at all. This section introduces some of these aspects and describes how they are measured, evaluated and compared.

The property that separates ride pooling from ride hailing is the option for the service operator to combine trips of multiple users, such that more than one request can be served simultaneously by one vehicle. This means, that the operator needs to take into account the capacity of each vehicle in the fleet when assigning requests. Even though the capacity constraint is also part of optimization in the ride hailing use case, as described in 3.2, it plays a much more vital role in the ride pooling use case. When offering a ride hailing service, it basically limits the amount of passengers that can be served with each request. This can be communicated to the service users, hence larger user groups can be generally excluded from the service and the capacity constraint does not limit the solution space for the assignment problem in any way.

In the ride pooling use case, however, the capacity constraint affects the assignment process significantly, because the operator aims to maximize the efficiency of the ODM fleet by pooling as many user requests as possible. In principle, the more requests can be served by one vehicle, the more vehicles are available for later requests, which translates to more served requests overall, and potentially more profit for the service provider and also fewer miles traveled, which improves the traffic situation in the service area. Therefore, one of the most important KPIs for ride pooling services is the average occupancy of vehicles in the fleet. A high average occupancy is beneficial for the service provider. It indicates that many rides are shared between users and the fleet efficiency is high. On the flip side, this has consequences for customers of the service, not all of which are favorable.

One positive aspect of ride pooling services compared to ride hailing offers for users is the increased efficiency and availability of vehicles, which in principle translates to a higher percentage of accepted requests and/or shorter pickup waiting times. Additionally, real-world service providers often offer ride pooling services at cheaper conditions than ride hailing rides because of the increased fleet efficiency. This pricing aspect of ODM services is not in the scope of this work, however, and prices are kept constant between both use cases for comparability.

One downside for users of ride pooling services is closely connected to the reason why it is more profitable for the provider. Every time a vehicle serves another request while having one or more customers on board, it needs to make a detour from the direct route to its current destination. This detour directly delays the drop-off time of each of the passengers. Delayed drop-offs of course do not contribute to a well-perceived service experience, even more so when they are hard to predict beforehand.

In order to avoid extensively long delays of drop-offs, operators of ride pooling services need to limit the detours a vehicle is allowed to make with customers on board. Such limited detour times can be implemented in at least two different ways. The first option is to allow a fixed amount of time that a drop-off might be scheduled later than expected if the vehicle would travel directly from the pickup location to the destination of a request. This implies that a customer that requests a short trip could face a detour that is actually longer than the originally planned ride. To prevent such experiences, another way to set a maximum detour time $t_{\text{det,max}}$ of a request is to define it relative to its direct travel time between pickup and drop-off. The latter is used in this work, in which $t_{\text{det,max}}$ is set to 50 % of the direct trip duration.

Both the mean occupancy of service vehicles and the percentage of the average user detour times relative to direct trips are important KPIs for ride pooling service providers. The negative consequences for users are compensated by the potential of a higher service availability.The quantitative differences in requests that can be served with varying service models are evaluated in this work.

Dynamic pricing strategies of ODM service providers are not in the scope of this work. For ride hailing and ride pooling services the fares are assumed to be the same. Note that the distance-related fare p_{dist} refers to the shortest possible distance between pickup and drop-off location of a request and does therefore not increase with detours.

Besides service providers and users, the third stakeholder in ODM services investigated in this work are cities the service is provided in. Ride pooling services are well-suited to improve urban traffic. The best KPI to measure the positive impact of this transportation mode compared to scenarios where all trips are made with self-owned vehicles or ride hailing, is the added distance due to the service D_{add}. This metric represents the percentage of additional mileage that is produced by service vehicles compared to the sum of all distances of direct trips from pickup to drop-off locations of served customers. If the value is negative, the service provides a net reduction of traffic in the business area, assuming that all trips would otherwise be made with vehicles owned privately. Positive values on the other hand indicate that a service increases the amount of traffic, typically due to empty trips.

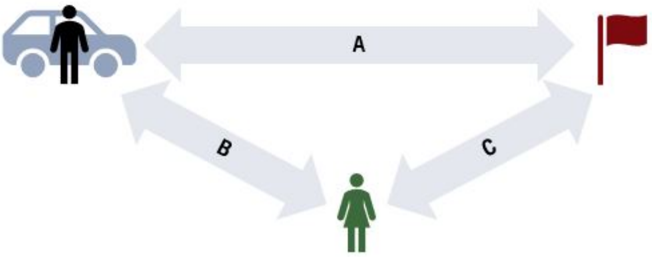


Figure 5.1: Supplemental illustration for the added distance due to service D_{add}.

A conceptional illustration of how the added distance due to a service is measured is presented in Figure 5.1 . Both users in the example share the same destination, indicated by the red flag. In a scenario without any ODM services, both would need to travel their respective direct path's "A" and "C". This distance $d_1 = d_A + d_C$ is the reference the "added distance" D_{add} relates to.

In a second scenario, both customers are served by a ride hailing service. The only available vehicle is at the pickup location of one of the service users. The shortest possible way to serve both requests is for the vehicle to pick up the first user, travel via path "A", drop off the passenger at the destination, go to the pickup location of the second user via "C", pick up that customer and return to the destination, again along path "C". In total, the distance covered therefore is $d_2 = d_A + 2d_C > d_1$ and

$$D_{\text{add}} = \frac{d_2 - d_1}{d_1} > 0. \tag{5.1}$$

The third and final example assumes that both customers are served by a ride pooling service and they can share their rides. The service vehicle can therefore take another route: after the picking up the first customer it can follow path "B" to the second customer, who also gets picked up and both users are transported to the destination via path "C" and dropped off at arrival. Assuming a geometry similar to the situation depicted in Figure 5.1, the distance between the service users is shorter than the direct path of the first user to the drop-off location. Hence, $d_3 = d_B + d_C < d_A + d_C = d_1$ and

$$D_{\text{add}} = \frac{d_3 - d_1}{d_1} < 0. \tag{5.2}$$

Note that D_{add} is not guaranteed to be negative for all routes that include shared rides of ride pooling service users. If the detour from one or more direct paths of customers on-board becomes too long, it is possible that the service produces additional net mileage compared to the scenario in which the customers use their own vehicles to get to their destinations instead. However, the maximum detour time constraint mitigates the number of feasible routes for which this holds.

5.2 Assignment Approaches

The optimal assignment of user requests to vehicles in the ride pooling use case is crucial, because of the sheer number of possible combinations of tasks within each vehicle's task queue. Unlike the ride hailing use case described in Chapter 4, in the ride pooling use case the maximum number of requests served within one task queue is limited mainly by the capacity of the vehicle and the maximum user detour time rather than the average trip time. That is because several nearby users can be picked up sequentially, before they are collectively driven in the direction of their destinations and dropped off.

In order to account for this enlarged solution space, two methods to search for the best vehicle-request assignments are presented in this section. First, the heuristic approach for initial assignments, followed by the global optimization algorithm to periodically improve the solution quality.

5.2.1 Insertion Heuristic

The initial assignment of a new request $j \in J$ to a service vehicle $i \in I$ and its respective task queue ξ_i is made by a heuristic as in the ride hailing use case. However, instead of only considering the time and location of a vehicle i when it is idle (in case $\xi_i \neq \emptyset$ after the last task in ξ_i), the heuristic used in the ride pooling use case needs to take into account every slot in ξ_i to search for a feasible pickup option. Also, each position of the respective drop-off in the sequence of the task queue is eligible, as long as the associated pickup precedes it.

The search space grows rapidly with the number of requests served within one task queue ξ_i. In order to efficiently eliminate infeasible sequences of tasks, it is not sufficient to check if the maximum waiting time constraint is fulfilled for the new request $j \in J$. Since the waiting times as well as the detour times of the other users $k \subset J$ in the task queue ξ_{ik}^{new} potentially change when inserting a new task before their respective pickup or drop-off, a feasibility check is necessary for all subsequent tasks. In order to not increase the computational complexity further, the sequence of already assigned tasks in ξ_{ik}^{old} remains unchanged. Only if the entire set of users $k \subset J$, including the new request j, can be served according to the constraints formulated in Sections 3.2 and 5.1, ξ_{ik}^{new} is considered feasible.

For each vehicle $i \in I$ there is potentially more than one sequence of tasks ξ_{ik}^{new} considered feasible for a given subset of users $k \subset J$. For vehicle $i \in I$ the set of all feasible combinations of tasks associated with the set of requests $k \subset J$ is denoted as Ξ_{ik}. In a pre-processing step before the selection of a vehicle that should serve a new request, the best task queue $\xi_{ik}^{\text{new}} \in \Xi_{ik}$ is identified for each vehicle $i \in I$ according to the control function F_{con} in Equation 3.1.

After the best task queue $\xi_{ik}^{\text{new}} \in \Xi_{ik}$ for vehicle $i \in I$ is found, the difference between the values of $F_{\text{con}}(\xi_{ik}^{\text{new}})$ and $F_{\text{con}}(\xi_{ik}^{\text{old}})$ is calculated.

$$\Delta F_{\text{con}} = F_{\text{con}}(\xi_{ik}^{\text{new}}) - F_{\text{con}}(\xi_{ik}^{\text{old}}) \tag{5.3}$$

If at least one vehicle can serve the new request $j \in J$, the assignment that implies the smallest ΔF_{con} is offered to the user.

The insertion heuristic for the ride pooling use case is considerably more sophisticated than its counterpart in the ride hailing use case in order to account for the increased complexity of the optimization problem. Its methodology can be described in three steps: i) search for feasible solutions, ii) identify the best solution for each vehicle individually, and iii) select the best assignment. This approach allows to efficiently find an initial assignment for incoming requests. Note that this heuristic procedure allows changes of the waiting and detour time of already assigned requests, albeit in the limits set by pickup time windows, maximum waiting and detour time constraints. This is a fundamental difference to the nearest neighbor policy used for initial assignments in the ride hailing use case, which is intrinsically unable to affect already made assignments.

5.2.2 Anytime Optimal Algorithm

Global reoptimization considers possible assignments of all users that had not yet been picked up to all vehicles that are able to serve the requests. Hence, two sets of assigned requests need to be considered separately: unlocked requests and locked requests. Locked requests need to be served by the vehicle they are already in or locked to. The only degree of freedom left to optimize is the time of their respective drop-offs, limited by the maximum detour time. However, unlocked requests have not yet been locked to a vehicle and are therefore free to be reassigned to any vehicle that is able to serve it within the constraints.

This work follows the approach of [Alonso-Mora, Samaranayake, et al., 2017] and [Engelhardt et al., 2019]. The idea of this method, which is referred to as "anytime optimal algorithm", is to eliminate most combinations of tasks within a given task queue ξ_{ik} before explicitly check feasibility and calculate values of $F_{\text{con}}(\xi_{ik})$. The concept is based on an added layer between requests and vehicles, denoted as bundle layer. The optimization problem is thereby effectively split in two: which request bundles $k \subset J$ can be served together and which vehicles $i \in I$ can serve the request bundles optimally, taking into account the users that are already locked.

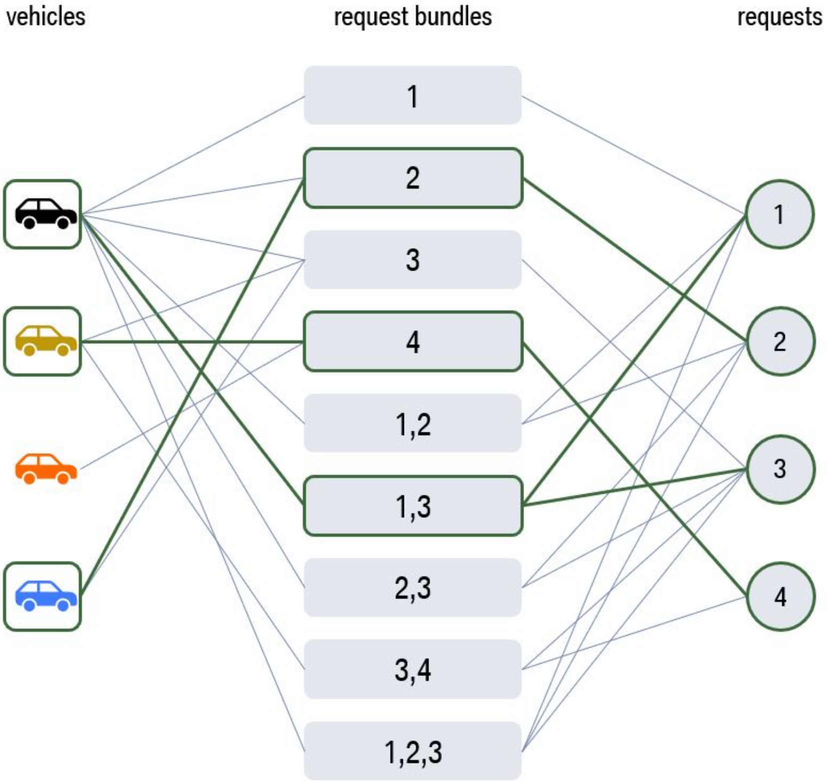


Figure 5.2: Exemplary graph formulation of the V2RB assignment problem. The edges highlighted in green indicate a possible optimal solution.

This three-layer assignment problem can be formulated as a graph problem, connecting vehicles to single requests (RV), requests with other requests (RR) and vehicles to request bundles (V2RB). Figure 5.2 presents an example of such a graph problem formulation. An edge on the left side of this graph between vehicle $i \in I$ and request bundle $k \subset J$ is denoted as V2RB (i, ξ_{ik}) and its cost is equivalent to the value of the control function F_{con} in Equation 3.1 for the best task queue ξ_{ik} of vehicle $i \in I$ that could be used to serve the requests in $k \subset J$. Two important remarks are that all requests locked to vehicle $i \in I$ must be in $k \subset J$ served by task queues ξ_{ik} connected to $i \in I$ and the task queue ξ_{ik}^{old} that was assigned to $i \in I$ before the global reoptimization is always connected to $i \in I$ to guarantee that the new solution is at least as good as the old one.

On the right-hand side, edges between bundles $k \subset J$ and requests $j \in J$ are referred to as RR (j, k). The set of edges connecting feasible request bundles that include request $j \in J$ may contain many subsets of requests, but always contains $(j, \{j\})$.

A key property of a request bundle $k \subset J$ is its rank r_k, which is the number of requests $j \in k$ contained by it. The graph representation of the problem together with definition of ranks allows the following conclusion: a V2RB (i, ξ_{ik}) of rank r_k cannot be feasible if one or more of the r_k V2RBs of rank $r_{k'} = r_k - 1$, with $k' \subset k$, is infeasible. This realization drastically reduces the number of potential solutions, because with increasing ranks the probability of finding feasible V2RBs decreases because of the maximum waiting time and detour constraints.

Consider an example with one vehicle $i \in I$ and three request j_1, j_2, $j_3 \in J$. Starting at $r_k = 1$, which represents the RV-edges in the graph, all assignments of single-request bundles to vehicles are checked for feasibility. In this example, the vehicle is able to serve each request in time if it moves directly to the respective request, so (i, j_1), (i, j_2) and (i, j_3) are all feasible RV-connections. Continuing with $r_k = 2$, RR-connections need to be considered as well. Since in this example all RV-connections are feasible, all six RR-connections with two requests $(j_1, \{j_1, j_2\})$, $(j_1, \{j_1, j_3\})$, $(j_2, \{j_1, j_2\})$, $(j_2, \{j_2, j_3\})$, $(j_3, \{j_1, j_3\})$ and $(j_3, \{j_2, j_3\})$ are checked for feasibility. If for one of these, e.g., $(j_1, \{j_1, j_3\})$ no task queue $\xi_{i\{j_1,j_3\}}$ is found that fulfills the feasibility constraints, the corresponding V2RB $(i, \xi_{i\{j_1,j_3\}})$ is considered infeasible. Coming to $r_k = 3$, we can make use of the fact, that there cannot be a feasible V2RB of rank $r_k = 3$ if there is an infeasible V2RB of rank $r_{k'} = 2$, such as $(i, \xi_{i\{j_1,j_3\}})$, with $k' \subset k$. Therefore, the V2RB $(i, \xi_{i\{j_1,j_2,j_3\}})$ is known to be infeasible without explicitly checking. Following this scheme, the graph-building process becomes manageable even for large fleets of vehicles and many unlocked requests.

With the complete graph built, the actual assignment problem is to find assignments of request bundles to vehicles such that all constraints are satisfied and the control function F_{con} is minimized. The anytime optimal algorithm used in this work is well-suited to globally optimize the assignments of ride pooling requests to vehicles in large-scale scenarios. This allows the comparison of service models in the ride hailing use case, which is intrinsically less complex and can therefore be simulated with problem sizes and model parameters close to real-world ODM services in case studies, like the one introduced in Section 3.4.

5.3 Results

This section is split in three parts. First, the system performances of three service models for the ride pooling use case, described in Section 3.1, are compared. In the subsequent part, a PSA is conducted for the 2-step service model and the variable parameters introduced in Section 3.4.2. At the end of this section, the results of the 2-step service models are compared between the ride hailing and the ride pooling use cases.

5.3.1 Evaluation of Service Models

Like in the evaluation of the ride hailing service models in Section 4.2, this section presents a comparison of the three service models described in Section 3.1, but for the ride pooling use case. The heuristic used in Service Models 1 ("no global optimization") and 2 ("2-step service") is presented in Section 5.2.1. The global optimization technique that is used in the latter as well as in Service Model 3 ("only global optimization") is described in Section 5.2.2.

In addition to the KPIs evaluated in the ride hailing use case, which are described in Section 3.4.3, this section provides insight into the performance indicators presented in Section 5.1, which are of special importance in the evaluation of ride pooling services.

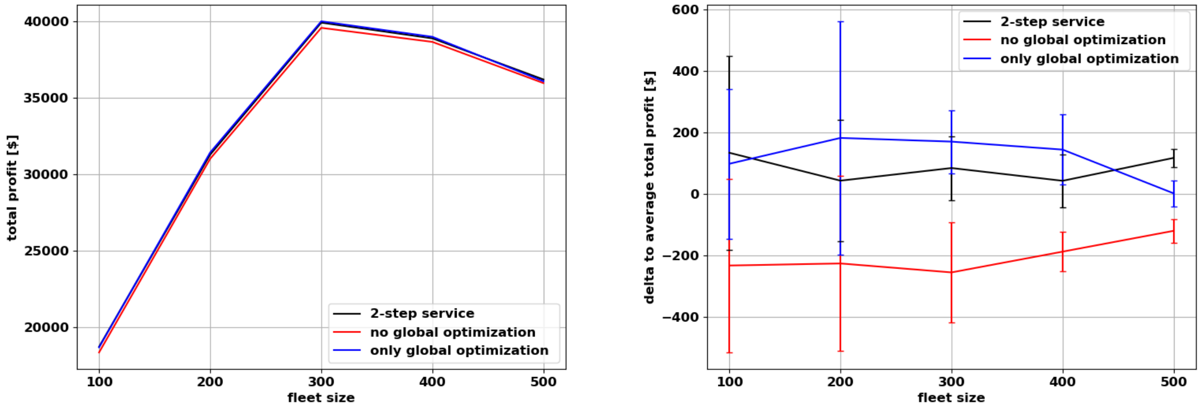


Figure 5.3: Profit in US-Dollars generated with various service models in the ride pooling use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

Figure 5.3 presents the average profit in US-Dollars generated per day. The maximum value for all the service models of approximately \$40 000 is produced in scenarios with fleet sizes of 300 vehicles. The delta to average on the right shows that Service Model 1 is outperformed by both of the others by a margin between (\$121 ± 81) compared to Service Model 3 in scenarios with 500 vehicles and (\$426 ± 265) in relation to Service Model 2 in scenarios with 300 vehicles. Expressed as a percentage, the deficit of Service Model 1 relative to the other models is between (0.3 ± 0.2) % (500 vehicles) and (2.0 ± 2.6) % (100 vehicles), while the difference between Service Models 2 and 3 is smaller than the standard deviation of the results in most of the scenarios.

In Figure 5.4, the percentage of customer requests served by each of the services is shown. In all scenarios with 400 or more vehicles, more than 99 % of requests are served with all ride pooling service models, which makes it unprofitable to further increase the fleet size, as seen in Figure 5.3. In Service Model 3, the percentage of served customers is highest on average in most scenarios. In the most profitable instances with fleet sizes of 300 vehicles, it outperforms the other service models by margins of (0.48 ± 0.46) % (Service Model 1) and (0.45 ± 0.41) % (Service Model 2), respectively.

An exact evaluation and comparison of computation times is difficult due to inevitable variance of performance of the hardware when it comes to long simulations like these. Nevertheless, the gap between the service models seems to be significant. As shown in Figure 5.5, simulations of Service Model 1 are computed the fastest, which is not surprising, because the time consuming global optimization of assignments is only performed in the other two service models. The average total computation time in Service Model 1 increases steadily with growing fleet sizes from 2385 s (ca. 40 min) in scenarios with 100 vehicles to 5259 s (ca. 88 min) in 500-vehicle scenarios. While the computation times for Service Model 3 grow similarly,

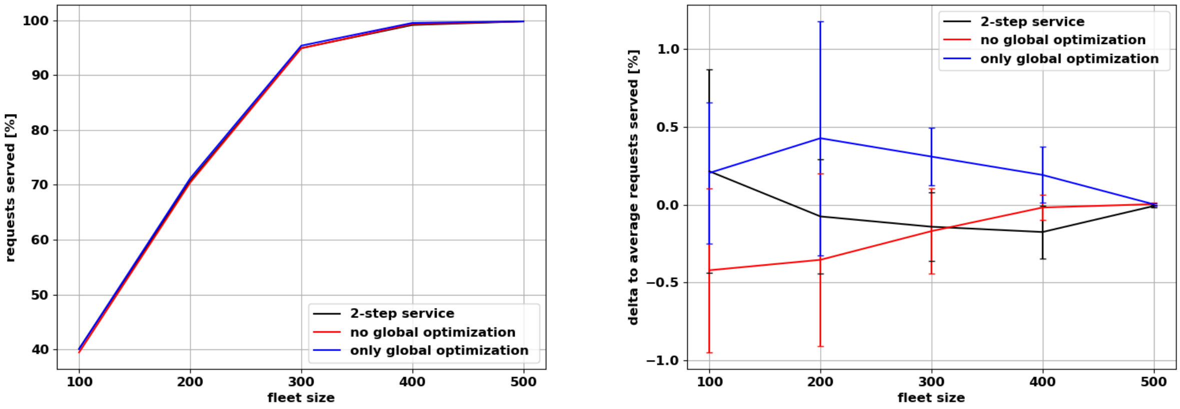


Figure 5.4: Percentage of requests served with various service models in the ride pooling use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

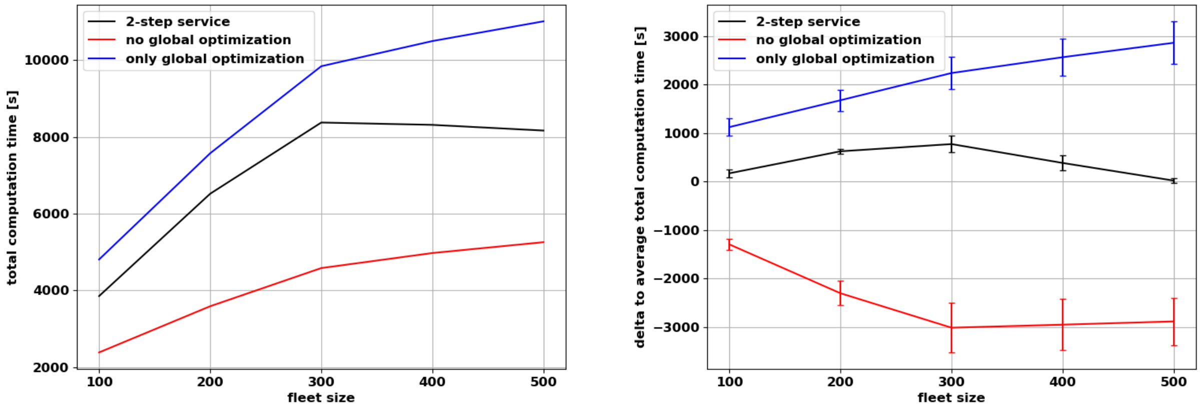
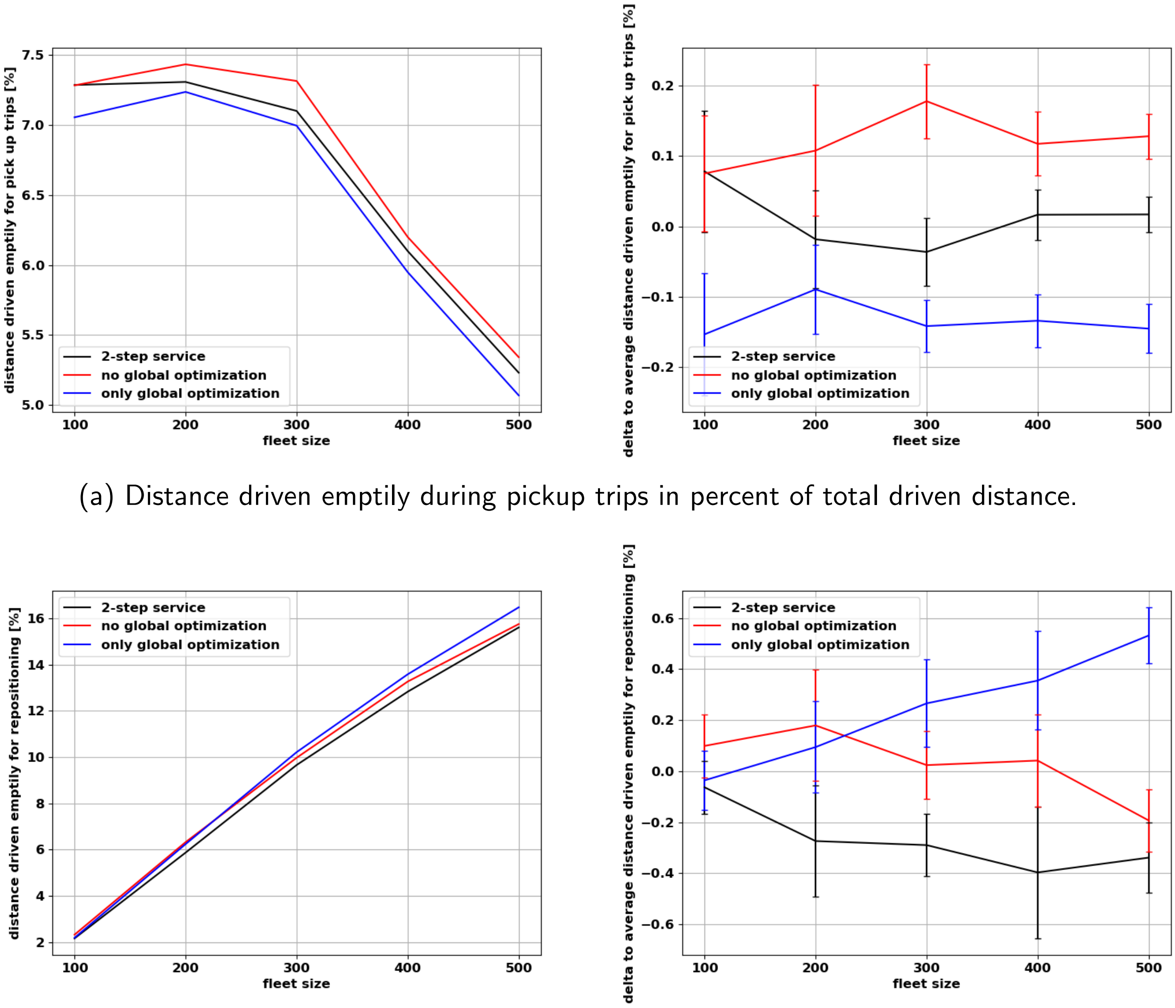


Figure 5.5: Computation times in seconds for simulations of various service models in the ride pooling use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

although with values more than twice as large as in Service Model 1 (between 4808 s and 11 014 s, or approximately 80 min and 184 min), the same cannot be said for Service Model 2. Instead, the maximum computation time of 8376 s (ca. 140 min) on average is measured in scenarios with fleet sizes of 300 vehicles which corresponds with the most profitable instances.

In general, the computation times in simulations of the 2-step service model are clearly faster than in the simulations of the service model only using global optimization. This observation can be explained by the reduction of the solution space in Service Model 2 by selecting the customers that are served with a heuristic instead of a global optimization algorithm. Optimizing the assignments in terms of which vehicle serves which request is clearly a less computational challenging task than including the search for an optimal selection of accepted customers.



(a) Distance driven emptily during pickup trips in percent of total driven distance.

(b) Distance driven emptily during repositioning trips in percent of total driven distance.

Figure 5.6: Distances driven emptily in percent of total mileage for various service models in the ride pooling use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

The distance driven emptily as percentage of the total distance traveled by the vehicles of the various services is presented in Figure 5.6, split up into its two components: empty trips to pickup locations (Figure 5.6a) and repositioning trips (Figure 5.6b). The total share of empty trips rises in all three service models with increasing fleet size. The empty pickup trips account for 7.05 % in Service Model 3 to 7.44 % of the total distance in Service Model 1 in scenarios with fleet sizes of 300 vehicles or less. In scenarios with larger fleets, the share decreases to 5.07 % (Service Model 3) to 5.34 % (Service Model 1) in scenarios with 500 vehicles. This makes sense, because with more vehicles in the fleet the average distance to the next available one becomes smaller.

On the other hand, the percentage of trips that are made during repositioning increases constantly from scenarios with 100 vehicles (between 2.16 % in Service Model 2 and 2.32 % in Service Model 1) to scenarios with 500 vehicles (between 15.76 % in Service Model 2 and 16.48 % in Service Model 3). Again, this is reasonable because in scenarios with larger fleets, more vehicles are idle and therefore eligible for repositioning, while in small-fleet scenarios most vehicles are needed to serve the demand.

Note that a high share of repositioning trips also implies that more vehicle are moved to areas with higher demand, which in turn potentially increases the chances to be able to serve upcoming request. So, while empty mileage is always to be avoided from the perspective of the city the business area is in, a high percentage of repositioning trips can be desirable for service providers to increase the percentage of served requests.

When comparing the three service models, Service Model 3 produces the fewest empty mileage due to pickup trips. This is reasonable because the model is based on the concept to wait a certain period of time before making the choice which requests should be served. This selection is based on the optimization of an objective function that includes the minimization of user waiting times and trip distances. Hence, it is expected that this model performs well in this regard. In all scenarios it produces between (0.09 ± 0.13) % (compared to Service Model 2 in scenarios with fleet sizes of 200 vehicles) and (0.32 ± 0.09) % (compared to Service Model 1 in scenarios with fleet sizes of 300 vehicles) less empty mileage due to pickup trips.

Another observation can be made when comparing the distances traveled during repositioning trips. While all service models perform relatively similar in scenarios with small fleet sizes, the larger the fleet sizes the clearer becomes the gap between Service Model 3 and Service Models 1 and 2. In scenarios with fleet sizes of 500 vehicles, the service model only using global optimization produces (0.72 ± 0.23) % more empty mileage due to repositioning than Service Model 1 and even (0.87 ± 0.25) % more than Service Model 2.

Putting both components together, the total distance driven emptily is dominated by the pickup trip distance in scenarios with small fleet sizes and by the distance traveled due to repositioning in large-fleet instances. Hence, even though Service Model 3 outperforms the other service models with regard to pickup trips, Service Model 2 produces the smallest total empty mileage for scenarios with 200 vehicles or more.

As mentioned before, though, the empty mileage is not the only metric to measure the impact of each service model on the traffic in the business area of the service. Figure 5.7 presents the added distance driven due to the service, a KPI introduced in detail in Section 5.1.

On the left side, the total values reveal that in all scenarios the pooling service models evaluated in this work have a positive impact on the traffic by reducing the net mileage driven in the system (negative added mileage). The reduction is highest in small-fleet scenarios, in which the driven distance is reduced by up to 18.9 % and becomes smaller with increasing fleet sizes. In scenarios with 500 vehicles, the distance saved by the service amounts to between 0.6 % (Service Model 1) and 2.1 % (Service Model 2).

The right-hand side shows the relative performances of the three service models in more detail. Independent of the fleet sizes in the scenarios, Service Model 2 outperforms the other service models. The gap to Service Model 3 remains relatively constant for all fleet sizes between (0.4 ± 0.5) % and (0.7 ± 0.3) %. The difference between Service Models 1 and 2 however is largest in scenarios with fleet sizes of 300 vehicles, in which the reduction is (2.0 ± 0.3) % higher in Service Model 2.

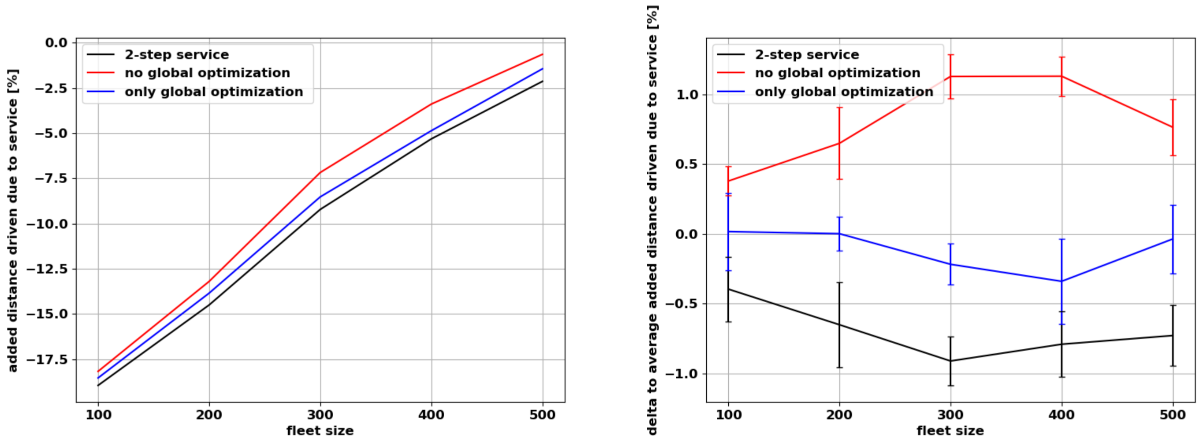


Figure 5.7: Added distances driven due to the service in percent of total mileage for various service models in the ride pooling use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

Another KPI that measures the efficiency of a ride pooling service, which is closely related to the implied mileage reduction is the mean occupancy of service vehicles. Driven distances can only be reduced if rides with a similar destination are shared and the more rides are shared, the higher is the occupancy throughout the fleet on average.

Therefore, it is not surprising to see in Figure 5.8 that the mean occupancy drops from between 1.87 and 1.89 passengers in scenarios with 100 vehicles to below 1.50 in scenarios with fleet sizes of 500. This observation corresponds with the reduction of distances driven due to the services, shown in Figure 5.7 as negative added mileage.

The same is true for the comparison between the three service models. Like observed before, Service Model 1 with no global optimization of the request assignments performs the worst in all scenarios considered. And again, the gap to the best-performing Service Model 2 is largest in 300-vehicle scenarios with a difference of 0.03 ± 0.01 passengers per vehicle, which is equivalent to $(1.9 \pm 0.5)\,\%$ of the mean occupancy in these scenarios.

Coming to the more customer-focused KPIs, Figure 5.9 presents the average user waiting time from sending the request to being picked-up. As expected, the average waiting time decreases with increasing fleet sizes due to the rising availability level of nearby service vehicles. In scenarios with 100 vehicles, the average waiting time is between 276 s (Service Model 2) and 285 s (Service Model 3), in the largest scenarios with 500 vehicles it comes down to 97 s to 109 s.

The direct comparison of the deltas on the right shows the clear gap between the service models. Service Model 2 picks up customers (0.6 ± 1.1) s to (5.1 ± 1.1) s faster than Service Model 1 depending on the fleet sizes. The clearest gap, however, is between Service Model 3 and the other two service models. The concept of this service model includes a period of time after each user request in which the operator does not directly react to the request. This period can be between 0 s and 30 s (the optimization period t_o) in these scenarios, depending on the timing of the request relative to the next global optimization.

This additional waiting time until pickup cannot be compensated by the enlarged solution

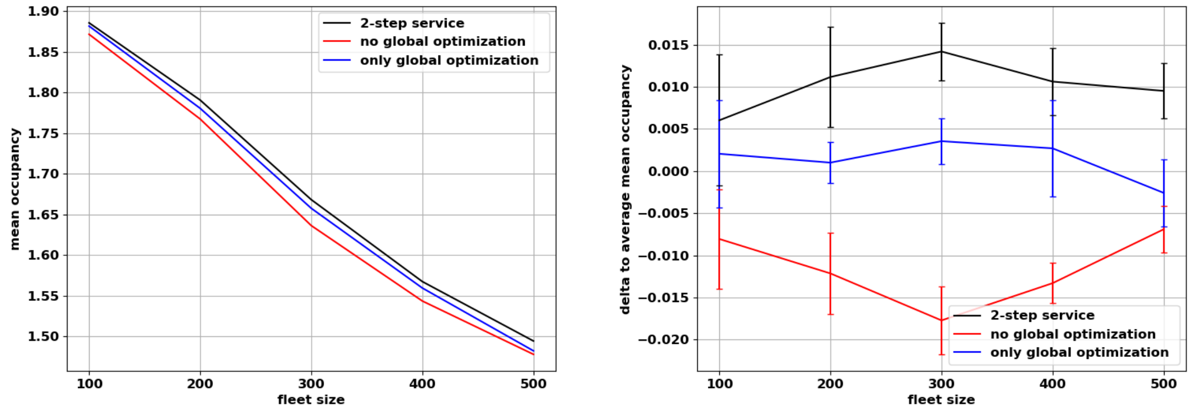


Figure 5.8: Occupancy in average passengers per vehicle for various service models in the ride pooling use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

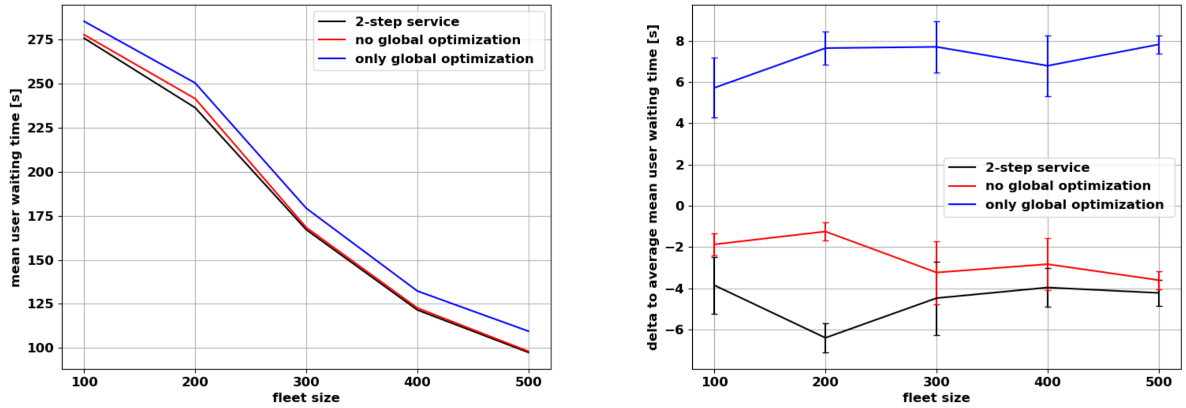


Figure 5.9: User waiting times in seconds with various service models in the ride pooling use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

space of this service model and can clearly be observed on the right side of Figure 5.9. The gap to Service Model 2 amounts to between (9.6 ± 2.8) s and (14.0 ± 1.5) s.

Another important indicator for the quality of ride pooling services perceived by users is the detour time. In Figure 5.10, the detour is presented as percentage of the shortest possible travel time between pickup and drop-off location. As stated in Section 5.1, the maximum value for this KPI is set to 50 %.

The left-hand side of the figure shows the mean user detour time over the fleet sizes considered. It clearly decreases with increasing numbers of vehicles in scenarios with all three service models. Starting between 26.7 % (Service Model 1) and 27.1 % (Service Model 3), it drops to values of between 15.3 % (Service Model 3) and 15.8 % (Service Model 1). This

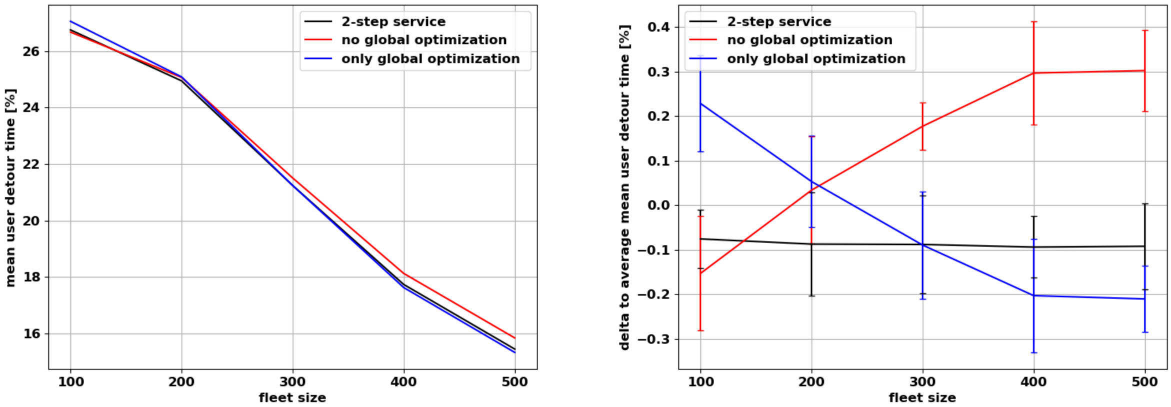


Figure 5.10: User detour times in percent of shortest possible path times with various service models in the ride pooling use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

observation makes sense considering that in scenarios with fewer vehicles, in order to serve as many requests as possible, the service operator has to send vehicles to locations further away from their direct paths compared to when more vehicles are available and chances are higher that another vehicle's detour is shorter or that the vehicle is idle.

The delta of the various service models does not provide such a clear picture, though. While for smaller fleet sizes, the largest percentages of detour times are measured in simulations of Service Model 3, the same service model performs best in this regard for fleet sizes of 300 vehicles and more. The exact opposite is true for Service Model 1: while it outperforms Service Models 2 and 3 by $(0.08 \pm 0.19)\,\%$ and $(0.38 \pm 0.17)\,\%$, respectively, in scenarios with fleet sizes of 100 vehicles, it performs worse and worse relative to the other service models with increasing fleet sizes. In the largest problem instances with 500 vehicles, the percentage of detour time is $(0.39 \pm 0.19)\,\%$ and $(0.51 \pm 0.16)\,\%$ larger than in Service Models 2 and 3 respectively.

The next and final KPI in the evaluation of the ride pooling service models is the pickup-in-time-window rate, which indicates what percentage of users are picked up within the time window initially projected for each request. Figure 5.11 presents the total value on the left and the deltas to the average of all service models' performances on the right.

Because in Service Model 1 the initial assignment is not changed anymore, the percentage of users that are picked up in their respective time window is always 100 %. In Service Models 2 and 3, the rate is lowest in scenarios with 200 vehicles in the fleets, with 97.21 % and 96.58 % respectively. For larger fleet sizes, the percentage increases, up to over 99 % in both service models in scenarios with fleet sizes of 500 vehicles. While Service Model 3 performs slightly worse than Service Model 2 in scenarios with fleet sizes of 100 and 200 vehicles, both reach similar values within their respective standard deviation.

Summing up the evaluation of the three ride pooling service models, the following observations can be made:

- Service Model 1 generates up to 2 % less profit than Service Models 2 and 3, both of

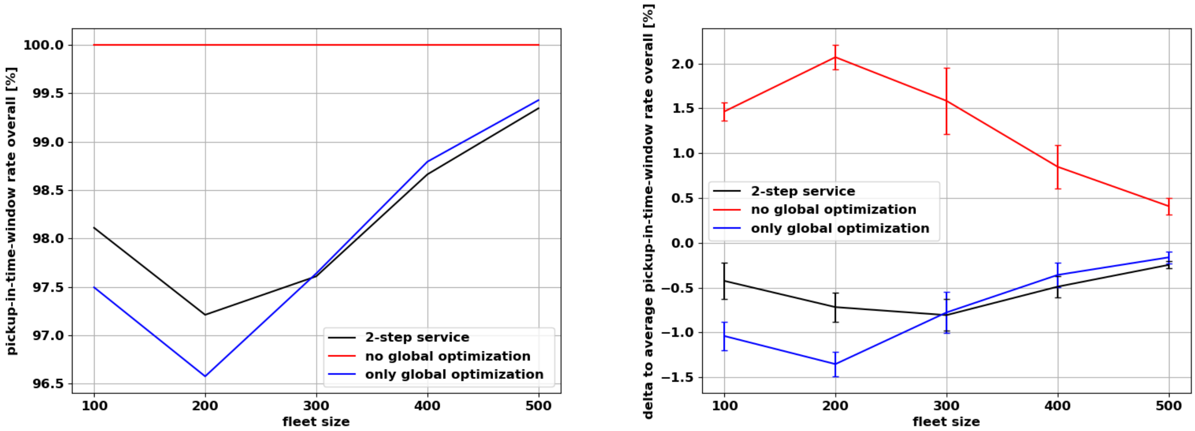


Figure 5.11: Pickup-in-time-window rate in percent for various service models in the ride pooling use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

which perform relatively similar in most scenarios considered. The highest profit per day is produced in scenarios with 300 vehicles with all three service models and amounts to approximately $40 000 in this case study.

- In Service Model 3, the percentage of requests served is highest on average in all scenarios considered. All service models are able to serve more than 99 % of all requests in scenarios with at least 400 vehicles.

- In terms of computation time, simulations of Service Model 1 can be run more than twice as fast as simulations of Service Model 3. While the computation times seem to grow steadily with fleet sizes in simulations of Service Models 1 and 3, when using Service Model 2, the peak is reached in scenarios with fleet sizes of 300 vehicles and the highest profit.

- The empty distances driven during pickup trips is almost constant in scenarios with fleet sizes of 300 vehicles or less with all service models before dropping by up to 2 %, while the share of empty trips due to repositioning increase steadily with increasing fleet sizes from around 2 % to over 15 %.

- Service Model 3 performs best in terms of empty pickup trips, but because Service Model 2 produces fewer mileage due to repositioning, the total empty mileage in scenarios with Service Model 2 is the lowest in scenarios with at least 200 vehicles.

- The added distance due to the service is negative in all scenarios considered, which means the ride pooling service models contribute to a net reduction of vehicle mileage in the business area. The total reduction decreases with increasing fleet sizes. The highest reduction of distances driven in the system independent of the fleet sizes is achieved by Service Model 2, followed by Service Model 3.

- Correlating with the reduced vehicle mileage, the mean occupancy of the vehicles decreases with increasing numbers of vehicles, and Service Model 2 outperforms Service Models 1 and 3.
- The waiting times between request and pickup drop from over 275 s in scenarios with 100 vehicles to below 100 s with 500 vehicles. Due to its model-specific properties, the average user waiting times in Service Model 3 are highest in all scenarios, while Service Model 2 outperforms Service Model 1 by up to more than 5 s.
- The percentage of user detour times relative to the direct path between the respective pickup and drop-off locations decreases steadily with increasing fleet size. In scenarios with fleet sizes of at least 300 vehicles, Service Model 3 produces the least amount of detour, while Service Model 1 generates the most.
- Outside Service Model 1, the pickup-in-time-window rate is lowest in scenarios with 200 vehicles, in which Service Model 2 reaches 97.2 % and Service Model 3 96.6 %. In scenarios with larger fleets, these two service models produce results that only differ within their respective standard deviation.

Comparison to Ride Hailing Use Case

Besides the ride pooling service models evaluated in this section, the same service model concepts are examined in the ride hailing use case in Section 4.2. A high-level comparison between the system performances in both use cases is helpful to understand their respective potentials, strengths and weaknesses, beyond the choice of a service model.

The KPIs in which ride pooling services perform significantly better are important for all three stakeholders of ODM services. The maximum daily profit for the service provider is increased by over 30 % compared to ride hailing services and can be achieved with smaller fleets. The percentage of requests served is higher in scenarios with all considered fleet sizes due to a higher average vehicle occupancy, notably also in scenarios with fleet sizes that produce the highest profitability, which is an indication that the service availability for customers is higher. Ride pooling service users also have shorter average waiting times until they are picked up: compared to ride hailing services with the same fleet size, the waiting time is approximately 1 min shorter. Furthermore, in all considered ride pooling scenarios, the ODM service has a positive effect on the traffic situation in the business area, while the added distances driven due to ride hailing services are between 23 % and 35 %, depending on the fleet size.

On the other hand, ride hailing service offer benefits with regard to two critical KPIs. The computation times of simulations of ride hailing services are only a fraction of the computation times necessary to simulate ride pooling services, which can be directly derived from the increased complexity of the assignment problems that need to be solved in scenarios in which rides can be shared. Longer computation times in real-world applications of ODM services can have a negative impact on the perceived service quality due to increased response and waiting times for customers. Another downside of ride pooling services is the fact that sharing rides unavoidably leads to detours that service users need to be willing to take. In the ride pooling scenarios considered in this work, the average detour relative to the direct travel distance between pickup and drop-off amounts to between 15 % and 27 %, depending on the fleet size.

From the perspective of the overall travel times of customers, these detours undo much of the benefit of shorter average waiting times until pickup.

All in all, the ride pooling use case studied in this work offers some substantial advantages compared to ride hailing. However, its weaknesses need to be addressed in order to provide a high-quality service to customers.

5.3.2 Evaluation of 2-Step Service Model Parameters

The PSA of the ride pooling service includes the parameters described in Section 3.4.2. Like in Section 4.2.2, the 2-step service model is used, fleet sizes of 300 vehicles are considered and outside of the respective parameter investigated, the standard parameter set is applied. Because of the additional KPIs of importance for the ride pooling use case, the system performance is not necessarily compared in terms of the same indicators as in the ride hailing use case.

Optimization Period

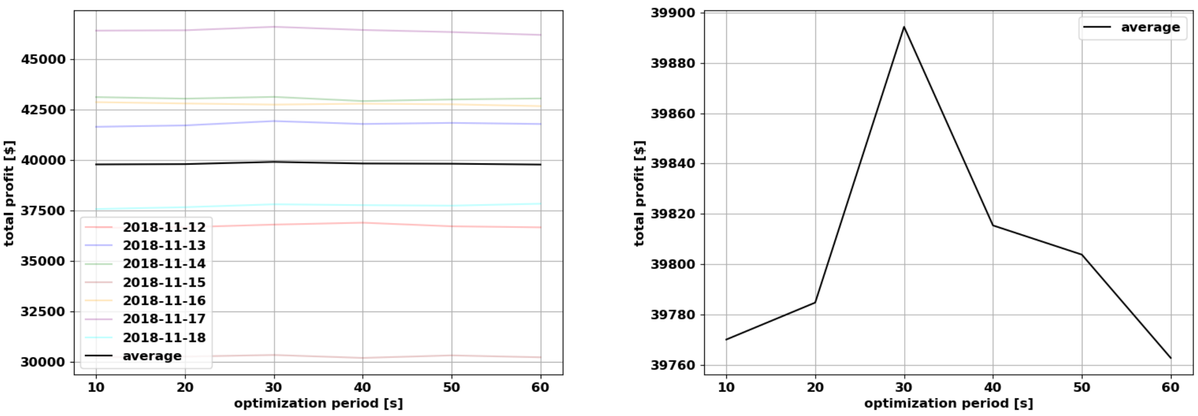


Figure 5.12: Profit in US-Dollars in simulations of the 2-step service model with 300 vehicles in the ride pooling use case with values of the optimization period t_o between 10 s and 60 s. Left: values for all simulation dates and the average value. Right: average value in detail.

The optimization period t_o states how long the intervals are between the global optimization of assignments of vehicles to requests. The investigated range is between 10 s and 60 s. The standard value used in the other evaluations is 30 s.

In Figure 5.12, the generated daily profit is presented in relation to t_o. As in the ride hailing use case, the variance between the simulations of different dates, shown on the left, is significantly higher than the variation of performances because of changing optimization periods. The reason for this is the difference in demand in each of the evaluated dates, which plays a huge role for the performance of the service in terms of almost every KPI. The PSA focuses on the variations in relation to the respective parameter, though.

As shown on the right-hand side of Figure 5.12, the maximum daily profit of \$39 894 is produced in scenarios with optimization periods of 30 s. In scenarios with the shortest and longest optimization periods, the generated profit is the lowest with values of \$39 770 and \$39 763 for t_o =10 s and 60 s respectively, which is approximately 0.3 % less than the maximum profit. It is therefore fair to deduce that the profitability of the service is rather insensitive to the optimization period.

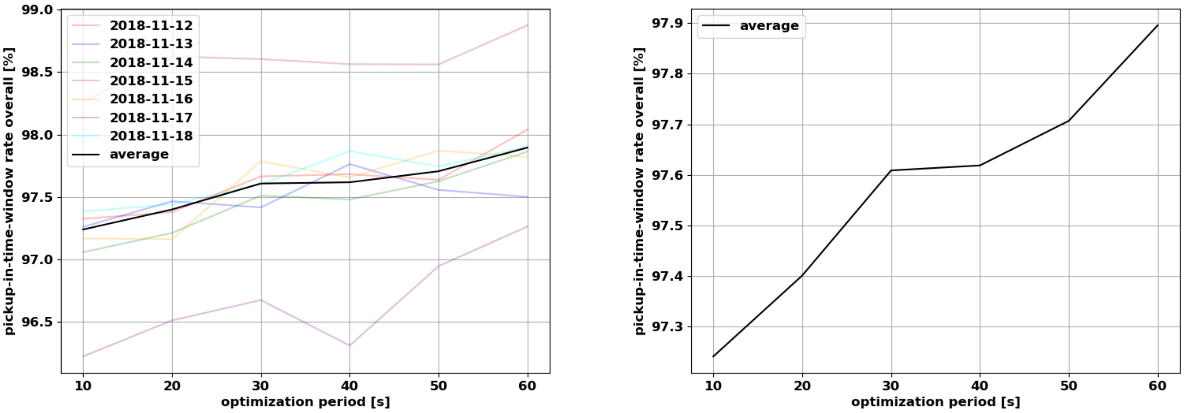


Figure 5.13: Pickup-in-time-window rate in percent of all accepted requests in simulations of the 2-step service model with 300 vehicles in the ride pooling use case with values of the optimization period t_o between 10 s and 60 s. Left: values for all simulation dates and the average value. Right: average value in detail.

The pickup-in-time-window rate shown in Figure 5.13 reveals a steady increase in the accuracy of the predicted pickup time window with increasing length of the optimization period. For t_o =10 s, averaged over all dates 97.24 % of accepted requests are picked up within the time window initially communicated to them. This value increases to 97.90 % in scenarios with optimization periods of 60 s. This trend can be explained by the fact that in scenarios with longer optimization periods more users are picked up before they are part of any global optimization. Such users are all picked up within their pickup time window, which increases the overall ratio.

Objective Weight of Distance

The objective weight of distance α is a parameter in the objective function (see Equation 3.1). With α, the service operator can control the priority of saving travel distances of service vehicles relative to reducing customer waiting times and picking up users within their respective time windows. High values of α imply that the objective function prioritizes solutions with short trip distances, even if that implies longer user waiting times, potentially outside of given pickup time windows. The range of evaluated values for α reaches from $6 \times 10^{-4}\,\mathrm{m}^{-1}$ to $6 \times 10^{1}\,\mathrm{m}^{-1}$. The x-axis is therefore set to a logarithmic scale for this parameter. The standard value is $6 \times 10^{-2}\,\mathrm{m}^{-1}$.

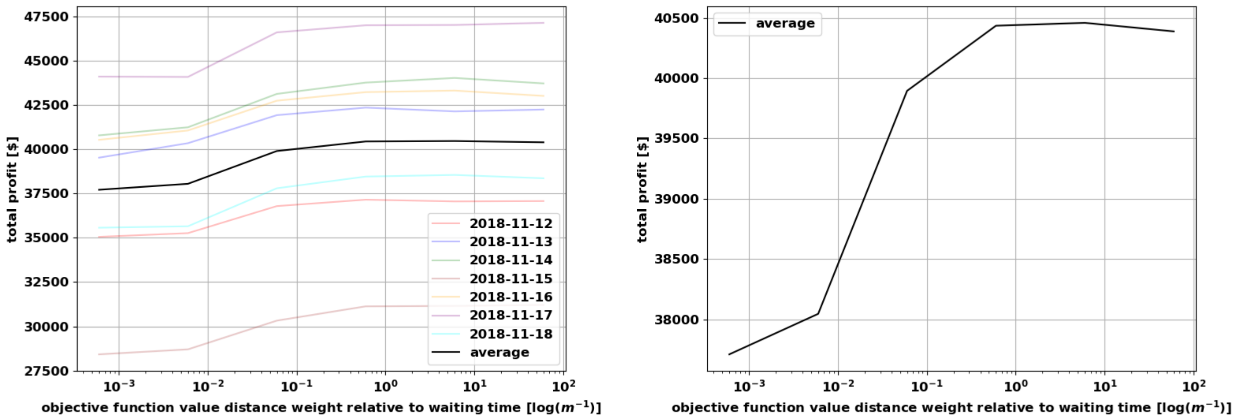


Figure 5.14: Profit in US-Dollars in simulations of the 2-step service model with 300 vehicles in the ride pooling use case with values of the objective weight of distance α between $6 \times 10^{-4}\,\mathrm{m}^{-1}$ and $6 \times 10^{1}\,\mathrm{m}^{-1}$. Left: values for all simulation dates and the average value. Right: average value in detail.

Figure 5.14 presents the daily profit generated in scenarios with varying α. A steep increase in profitability can be observed between scenarios with $\alpha \leq 6 \times 10^{-3}\,\mathrm{m}^{-1}$, in which the average profit does not surpass \$38 045, and scenarios with $\alpha \geq 6 \times 10^{-1}\,\mathrm{m}^{-1}$ in which the profit is between \$40 387 and \$40 458. This gap is equivalent to approximately 6 % to 7 % of the total profit per day. The profit made with the standard value of α is 1.4 % lower than the maximum. The main reason for this considerable gap in profitability between varying values of α is the focus on sharing rides in favor of some service quality parameters.

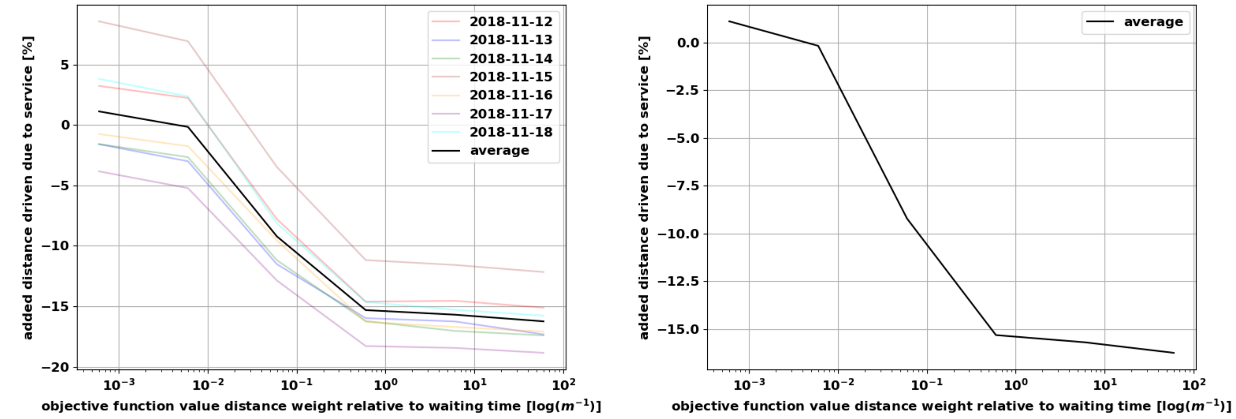


Figure 5.15: Added distances driven due to the service in percent of total mileage in simulations of the 2-step service model with 300 vehicles in the ride pooling use case with values of the objective weight of distance α between $6 \times 10^{-4}\,\mathrm{m}^{-1}$ and $6 \times 10^{1}\,\mathrm{m}^{-1}$. Left: values for all simulation dates and the average value. Right: average value in detail.

This can easily be seen when looking at Figure 5.15, which depicts the added distance due to the service. In scenarios with very small values of α, the service does not have a positive impact on the traffic. The higher the objective weight of distance, the more mileage is saved. In scenarios with $\alpha = 6 \times 10^{-1}\,\mathrm{m}^{-1}$ or higher, the reduction in the distance service vehicles have to travel compared to scenarios in which every customer would use privately-owned vehicles, is more than 15 %.

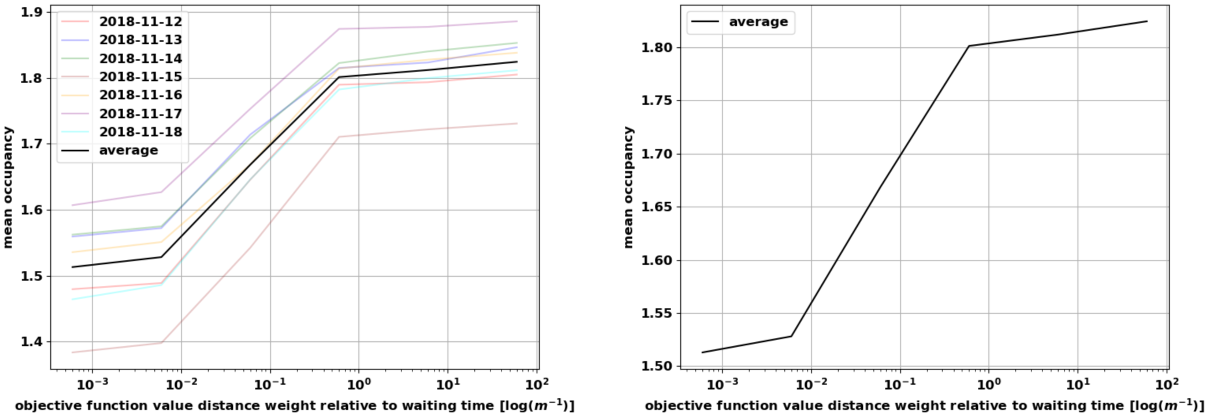


Figure 5.16: Occupancy in average passengers per vehicle in simulations of the 2-step service model with 300 vehicles in the ride pooling use case with values of the objective weight of distance α between $6 \times 10^{-4}\,\mathrm{m}^{-1}$ and $6 \times 10^{1}\,\mathrm{m}^{-1}$. Left: values for all simulation dates and the average value. Right: average value in detail.

This reduction can only be achieved if the mean occupancy per vehicle increases, which can be observed in Figure 5.16. While in scenarios with $\alpha = 6 \times 10^{-4}\,\mathrm{m}^{-1}$ the mean occupancy is 1.51 passengers per vehicle, this value increases to up to 1.82 on average in scenarios with $\alpha = 6 \times 10^{1}\,\mathrm{m}^{-1}$. This means that shifting the focus in the objective function from minimizing the waiting time to minimizing the travel distances, can cause an increase of over 20 % in the average number of passengers per service vehicle.

The choice of α also has a very distinct effect on KPIs reflecting the service experience of users. In Figures 5.17 and 5.18, the mean user waiting and detour times are presented, the latter as percentage of the shortest possible trip durations. Both show a similar dependency on α: for values of $\alpha \leq 6 \times 10^{-2}\,\mathrm{m}^{-1}$ the user experience is good, with short waiting times until pickup (167 s or less) and low percentages of detour times (21.24 % or less). Increasing the focus of the objective function further to minimizing the distances traveled, leads to a jump in both KPIs. The percentage of detour times increases to values between 23.28 % ($\alpha = 6 \times 10^{-1}\,\mathrm{m}^{-1}$) and 23.75 % ($\alpha = 6 \times 10^{1}\,\mathrm{m}^{-1}$). The user waiting time until pickup increases to 225 s in scenarios with $\alpha = 6 \times 10^{-1}\,\mathrm{m}^{-1}$ and even 243 s when $\alpha = 6 \times 10^{1}\,\mathrm{m}^{-1}$, a rise of 35 % and 46 % compared to scenarios with the standard value.

Such a strong effect on the waiting time of customers also has consequences on the pickup-in-time-window rate, as shown in Figure 5.19. While in scenarios with $\alpha \leq 6 \times 10^{-2}\,\mathrm{m}^{-1}$ the percentage of correctly predicted pickup time windows is always higher than 97.5 %, the rate drops significantly to between 81.2 % and 86.1 % for higher values. This implies that the longer

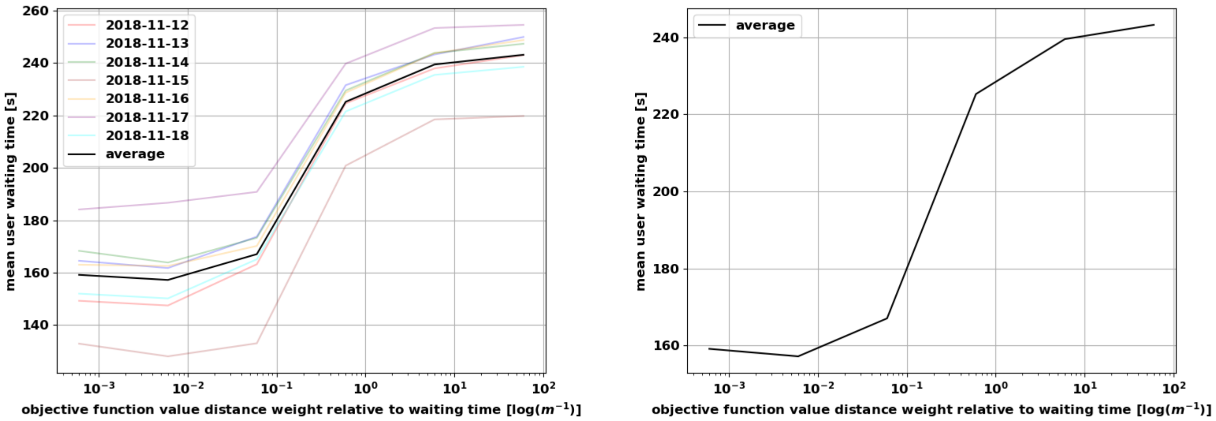


Figure 5.17: User waiting times in percent of shortest possible path times in simulations of the 2-step service model with 300 vehicles in the ride pooling use case with values of the objective weight of distance α between $6 \times 10^{-4}\,\mathrm{m}^{-1}$ and $6 \times 10^{1}\,\mathrm{m}^{-1}$. Left: values for all simulation dates and the average value. Right: average value in detail.

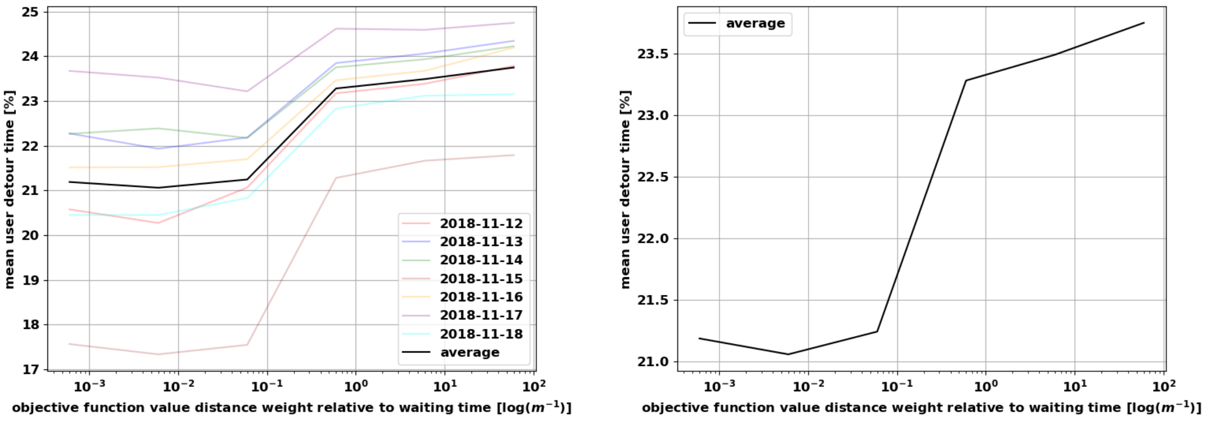


Figure 5.18: User detour times in percent of shortest possible path times in simulations of the 2-step service model with 300 vehicles in the ride pooling use case with values of the objective weight of distance α between $6 \times 10^{-4}\,\mathrm{m}^{-1}$ and $6 \times 10^{1}\,\mathrm{m}^{-1}$. Left: values for all simulation dates and the average value. Right: average value in detail.

waiting times are mainly caused by reassignments of requests after their initial assignment, which also defines the communicated pickup time window. Because of the high value of α relative to the weight of accepted requests γ and the penalty for assignments outside of the pickup time window p_{tw}, the objective function prioritizes solutions with short distances over others with more accurate pickup time predictions.

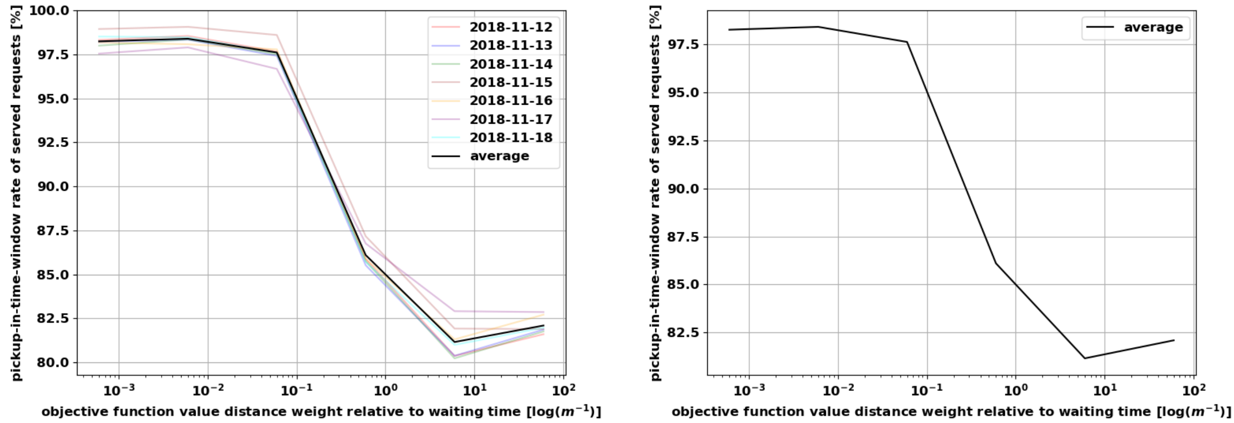


Figure 5.19: Pickup-in-time-window rate in percent in simulations of the 2-step service model with 300 vehicles in the ride pooling use case with values of the objective weight of distance α between 6×10^{-4} m^{-1} and 6×10^{1} m^{-1}. Left: values for all simulation dates and the average value. Right: average value in detail.

Assignment Lock Time

The timing of when assignments are locked plays a big role in the reception of the 2-step service model from a customer's point of view. It defines the time t_{lock} before a projected pickup, after which no changes can be made to the respective assignment, which in this service model coincides with the time when the final assignment is communicated to the service user. The values for t_{lock} examined in this PSA range from 0 s to 300 s (5 min), 120 s (2 min) being the standard value.

The daily profit is not very sensitive to t_{lock}, as can be seen in Figure 5.20. The maximum profit of \$39 894 is reached in scenarios with the standard value $t_{\text{lock}} = 120$ s. In all scenarios the profit per day remains within 0.4 % of this number, with no clear trend to be observed in relation to the assignment lock time.

A clear drawback of shorter assignment lock times can be observed in Figure 5.21, in which the pickup-in-time-window rate is presented. With increasing values of t_{lock}, the rate of correctly predicted pickup time windows rises from 97.7 % to 99.3 %. This observation can be explained by the fact that the number of opportunities for an request to be reassigned decline with longer assignment lock times. Therefore, the chance to be assigned to a vehicle that is not able to pick up the user in his or her respective pickup time window also becomes smaller, ultimately causing higher pickup-in-time-window rates.

Pickup Time Window Length

The pickup time window length t_{twl} is the last parameter examined in this PSA. Variations of t_{twl} have an effect on not only the number of possible solutions for the global optimization problem, but also the user experience. Shorter t_{twl} imply an improved predictability for customers when their pickup will take place. Long pickup time windows on the other hand are associated with a bad user experience, because the uncertainty of the pickup time of service

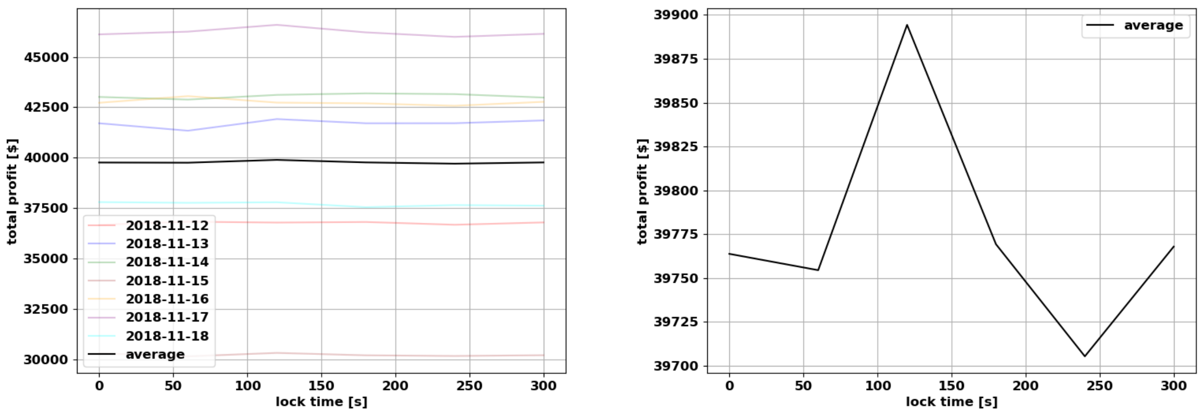


Figure 5.20: Profit in US-Dollars in simulations of the 2-step service model with 300 vehicles in the ride pooling use case with values of assignment lock time t_{lock} between 0 min and 5 min. Left: values for all simulation dates and the average value. Right: average value in detail.

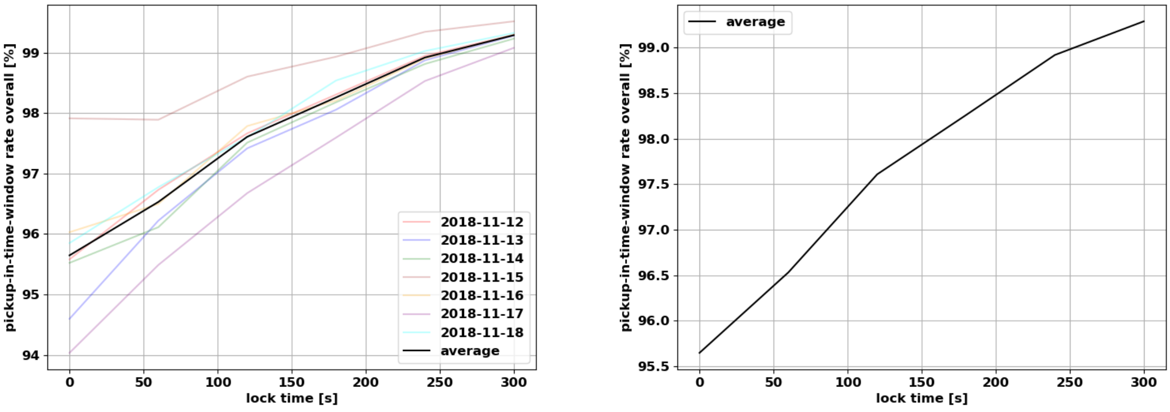


Figure 5.21: Pickup-in-time-window rate in percent in simulations of the 2-step service model with 300 vehicles in the ride pooling use case with values of assignment lock time t_{lock} between 0 min and 5 min. Left: values for all simulation dates and the average value. Right: average value in detail.

users is higher. The PSA is conducted with values of $t_{\text{twl}} = 0\,\text{s}$ to 300 s (5 min) with a standard value of 120 s (2 min).

Like the relation between profit and the assignment lock time, Figure 5.22 reveals that the pickup time window length also does not seem to have a significant impact on the service profitability. In all scenarios considered, the profit ranges between \$39 772 and \$39 914, which is a difference of less than 0.4 %. The insensitivity of this KPI to t_{twl} is also shown by the lack of an apparent trend of the profit in dependence of the length of pickup time windows.

As expected, however, the pickup-in-time-window rate presented in Figure 5.23 correlates

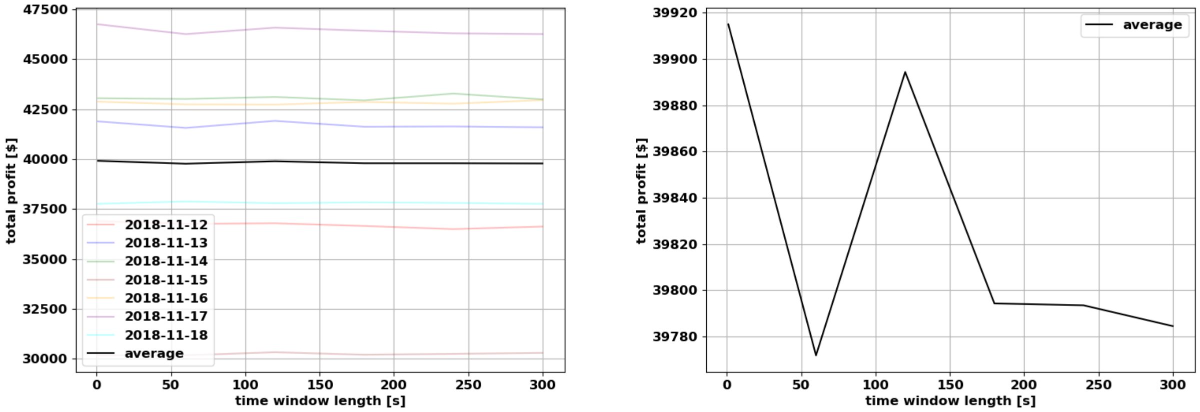


Figure 5.22: Profit in US-Dollars in simulations of the 2-step service model with 300 vehicles in the ride pooling use case with values of pickup time window lengths t_{twl} between 0 min and 5 min. Left: values for all simulation dates and the average value. Right: average value in detail.

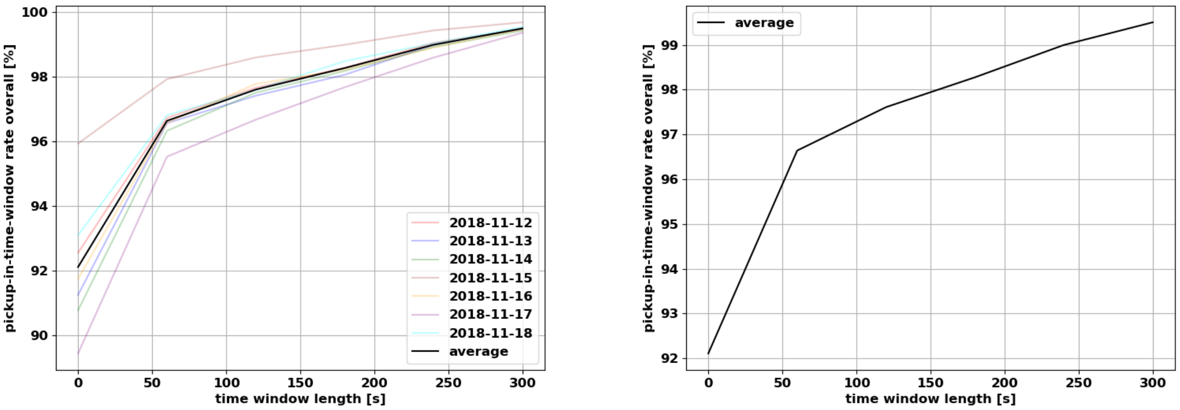


Figure 5.23: Pickup-in-time-window rate in percent in simulations of the 2-step service model with 300 vehicles in the ride pooling use case with values of pickup time window lengths t_{twl} between 0 min and 5 min. Left: values for all simulation dates and the average value. Right: average value in detail.

strongly with t_{twl}. The share of correctly predicted pickup time windows declines almost linearly with decreasing t_{twl}, from 99.5 % in scenarios with $t_{\text{twl}} = 300\,\text{s}$ to 96.7 % when the pickup time window length is 60 s. There is a steep drop in scenarios in which the time window is reduced to a single point in time, though. The average pickup-in-time-window rate in these scenarios is 92.1 %.

Putting together these results, it seems fair to conclude that the system performance is not impaired too much by changing pickup time window lengths, while the accuracy of pickup time predictions remain on a reasonably high level, at least for $t_{\text{twl}} \geq 1\,\text{min}$.

Chapter 6

Diffusion Customer Model

Up to this point, this work focused on the evaluation of ODM service models and their parameters in agent-based simulations. The purpose of such evaluations was to determine the effectiveness and performance of the respective services in order to make conclusions for real-world applications.

The results found in these simulations are comparable between the service models, because the simulation framework is the same for all of them. However, the assumptions made in order to improve the comprehensibility and transparency of the evaluation do not allow a direct transfer of the results to how an exact projection the service model would perform in reality. That is the point of any model: it needs to be simple enough to allow to observe consequences of parameter changes, while being as close a possible to its real-world equivalent.

Many of the assumptions and simplifications of the simulation framework and models used in this work are described in Chapter 3. Most of them facilitate the understanding of the evaluation as a whole or avoid aspects of reality that are very difficult to implement but do not necessarily help to gain insight into the topic that is investigated.

One part of the simulation framework that is often regarded as such a difficult-to-implement aspect, is the customer model. As pointed out in Section 2.3, in the literature there is a lack of such models that can easily be implemented into ODM simulation frameworks, while being able to reflect very basic characteristics of realistic service users.

As described in Section 3.3, the "conventional customer model" (CCM) includes attributes associated with each service request, like the time of request, pickup and drop-off locations and the number of passengers. Outside of these properties, the CCM is very limited in its ability to emulate any human-like behavior. The service offer made by the operator can be arbitrarily bad as long as the waiting time t_{w} until pickup is shorter than the maximum waiting time t_{max} set as model parameter. Such offers will always be accepted immediately by customers in the CCM, which is not realistic but the standard customer model used in literature (see Section 2.3).

This chapter introduces a novel ODM customer model, which aims to be reasonably simple to comprehend and interpret, and to include facets of human behavior that are typically observed in real-world ODM services. The concepts of this model are described in Section 6.1, its parameters introduced in Section 6.2 and its impact on the evaluation of ride hailing and pooling service models are presented in Sections 6.3 and 6.4, respectively.

6.1 Concept and Definition

The concept and design of the ODM customer model introduced in this section is influenced mainly by the "generalized diffusion model for preference and response time" presented in [Yu and Hyland, 2020]. Because of several fundamental alterations and adjustments to the service models used in this work, it will instead be referred to as "diffusion customer model" (DCM), though.

The goal of the DCM is to represent a more realistic and plausible ODM service customer model than the CCM, while being simple to implement, able to generate reproducible results and traceable in its consequences on the system performance. Note that it explicitly is neither designed to nor aimed to generally improve the system performance of ODM services. Instead, the evaluation of service models should be closer to realistic values that would occur in real-world applications of these services.

Differences between CCM and DCM

In the CCM, the decision about accepting or rejecting a request is exclusively made by the service operator. Only if it is not possible to pick up a user within t_{w}, the request is rejected and the deleted from the system. Otherwise, the offer is guaranteed to be accepted by the user who made the request.

In reality, service providers are very unlikely to send rejections as long as there is any chance the offer might be accepted. On the other hand, only a certain percentage of offers will be accepted by customers and the chance of being accepted should depend on the quality of the offer.

Furthermore, the decision-making process requires some time on the customer side. Instead of an immediate reaction like in the CCM, in the DCM each user takes a certain amount of time before responding to the service operator. This makes it more difficult for the operator to manage all users' requests simultaneously, many of them still in the process of confirming offers. The average duration of this process also depends on the offer quality, which takes into account that customers are torn between accepting and rejecting offers that are mediocre.

Since not every service user behaves the same, the DCM also allows a variance of certain model parameters to reflect varying characteristics that have an impact on the decision-making process. Also, the decision is not predetermined to be the same for two users with identical parameter values, facing offers with the same quality. This element of randomness requires the service operator to be more flexible when reacting to unexpected acceptances and rejections, again emphasizing the model's increases realism.

Methodology

The method used to model user behavior in the DCM is a combination of the (binary) decision field theory and the random utility model.

In the binary decision field theory, decision making can be modeled as an dynamic, accumulative process in which two contrasting stimuli stochastically affect the intermediate decision state D_t until a certain decision threshold θ is reached [Busemeyer and Townsend, 1992; Busemeyer and Diederich, 2002]. The binary choices in this theory are a variant of a discrete "random walk model". The random walk, however, is drifting toward one of the two directions implicated by the possible choices in the model. In the DCM, this binary choice is made between "accepting" or "rejecting" the offer, and the drift is defined by the offer quality q. If an offer is perceived as "good", a customer is more likely to accept it. Hence, $q > 0$ and the random walk tends to head towards the decision threshold $+\theta$, associated with accepting the offer. On the other hand, if the offer is "bad", the offer quality is negative and the drift tilts the way of the random walk towards $-\theta$, which is equivalent with rejecting the offer.

The way q is defined in the DCM has a significant impact on the model itself. This work derives q exclusively from the waiting time until pickup that is implied by the offer associated with q. There are potentially many more parameters that affect the offer quality perceived by service users, which remains outside the scope of this work. The waiting time t_{w} is one of the most crucial ones, though, which motivates this parameter's design choice.

The value of q is limited to the range $[-1, 1]$, negative values being associated with offers that are perceived worse than offers with positive values of q. The way q is calculated depends on two variables, $t_{\text{w},1}$ and $t_{\text{w},2}$, which are referred to as the first and second decisive waiting times respectively.

$$q(t_{\text{w}}, t_{\text{w},1}, t_{\text{w},2}) = \begin{cases} 1 & \forall t_{\text{w}} \in [0, t_{\text{w},1}] \\ 1 - 2\frac{t_{\text{w}} - t_{\text{w},1}}{t_{\text{w},2} - t_{\text{w},1}} & \forall t_{\text{w}} \in]t_{\text{w},1}, t_{\text{w},2}] \\ -1 & \text{otherwise} \end{cases} \tag{6.1}$$

This implies, that the offer quality is constantly at its maximum value of 1 if the waiting time is shorter than $t_{\text{w},1}$, then drops linearly to -1 for longer t_{w} until $t_{\text{w}} = t_{\text{w},2}$. In the DCM, the second decisive waiting time of the customer $t_{\text{w},2}$ is equivalent with the maximum waiting time t_{max}. The operator is assumed to be aware of this threshold and rejects user requests that can only be served with pickup waiting times $t_{\text{pu}} > t_{\text{max}}$.

Note that the offer quality is calculated separately for each offer that is sent to the customer. The initial offer's quality is referred to as q_1. If another offer is sent to a user, the associated quality is q_2. The waiting time t_{w} used to calculate both q_1 and q_2 is the difference between the estimated pickup time t_{pu} and the time of request t_{req}.

The decision-making process can be formulated as follows. When the service operator sends an offer, the user starts the decision-making process after an initial reaction time t_{re}. This period is reserved for perceiving the offer and interactions with the service applications.

Then, after initializing the decision state $D_{t=0} = D_0$, at any point in time t, the decision state D_t is updated based on the valence V and the feedback rate S.

$$D_{t+\Delta t} = SD_t + V(t) \tag{6.2}$$

The time step size Δt of the random walk is predefined as a model parameter. The feedback rate quantifies how much the next decision state is influenced by the current value of D_t. It reflects the decisiveness and short-term memory of a customer, in the sense that low values of $S \ll 1$ imply decision-making processes that do not evolve further away from the initial decision state D_0. If $S = 1$, the decision-making process is not hindered by setbacks like these, implying that the random walk of the decision-making process reaches greater highs and lows faster.

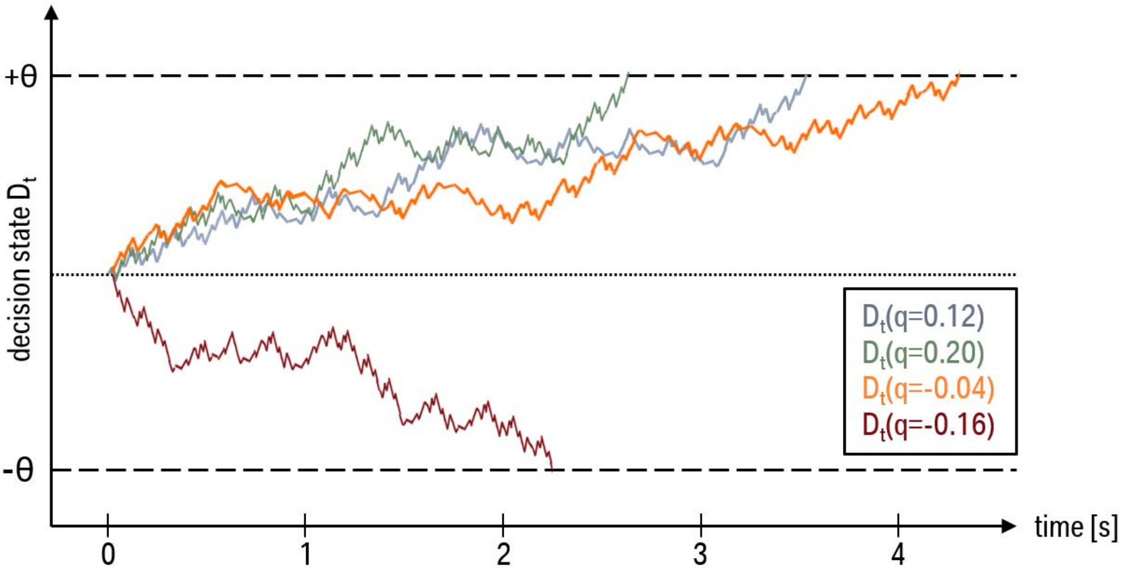


Figure 6.1: Examples of four decision-making processes with varying offer qualities in the DCM.

The valence V represents the sum of the perceived utility of an option (offer quality q) and a random term ϵ. The offer quality q is determined by the expected waiting time associated with it and remains constant throughout each individual decision-making process. In this work, the randomness of ϵ represents the stochastic element of cognitive processes, which is one of two common interpretations of this term. It was originally stated in [Thurstone, 1927] as part of the random utility model, in contrast to the interpretation that it represents nothing but the lack of information about the individual decision maker, advocated by e.g., [Train, 2001]. The mean value of ϵ is zero, its variance σ^2 is equivalent with the diffusion rate ϕ in the context of the DCM. The distribution of ϵ can be approximated with the normal distribution for small values of Δt [Busemeyer and Townsend, 1992].

The process of updating D_t is repeated iteratively until one of the following conditions is met: (a) one of the decision thresholds $\pm\theta$ is reached, (b) the maximum decision duration $t_{\text{d,max}}$ is exceeded or (c) the assigned vehicle arrives at the pickup location of the user that is still in the decision-making process. In case of (a), the offer is either accepted (if $D \geq +\theta$) or rejected (if $D \leq -\theta$). If (b), the offer is accepted if $D \geq 0$ and rejected otherwise. And in case of (c) the customer's decision-making process is stopped, the offer is accepted and the boarding process initiated.

Figure 6.1 illustrates four exemplary decision-making processes with varying offer qualities. Note that the process time in this illustration starts after t_{re} during which the decision states of all four instances is constantly zero and $D_0 = 0$.

The random nature of the process can be recognized in the erratic way each decision state takes. In the cases of the processes depicted in gray and green, the offer qualities are positive, which in the DCM means they are perceived by the service user as good offers. The operator therefore expects that these offers are accepted, which they eventually are. The decision-making process takes slightly longer in the case of $q = 0.12$ compared to the scenario in which $q = 0.20$, which is also typical for the DCM. Because the drift of the random walk is smaller,

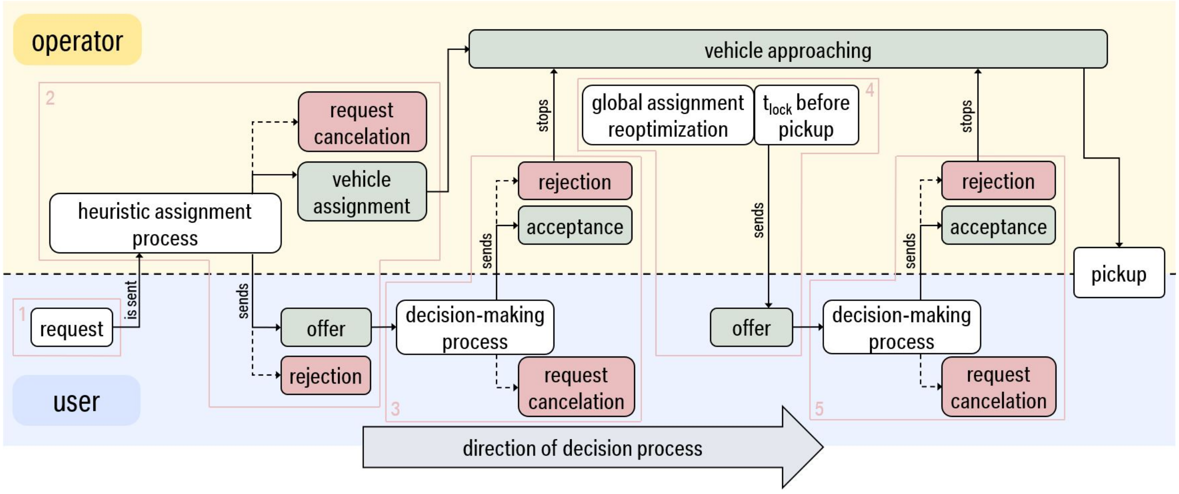


Figure 6.2: Interactive process between service operator and user using the DCM.

the average number of steps necessary to reach θ increases because the random element of the valence dominates.

In the examples with negative values of q, the outcome of the decision-making process varies. In the yellow instance, in which $q = -0.04$, the final decision by the user is to accept the offer, even though it is normally perceived as bad. Such a decision is categorized as "unexpected acceptance". Because of its small absolute value of q, this decision-making process takes the longest, as anticipated.

The only example of a rejection is shown in red. Here, the offer quality is the lowest between all shown examples, which implies that the probability of the offer being rejected is the highest from the start.

Decision-making process Embedded in Simulation Framework

Figure 6.2 shows how the DCM is implemented in the simulation framework. Unlike the CCM, in the DCM the decision-making process is split between the service operator (upper part of the figure in yellow) and the user (bottom part in blue). The general direction of the process illustration is from left to right.

The process begins at the user's side with sending the service request to the operator (box 1). The operator then starts a heuristic to find a vehicle that can serve the request (box 2). If this search is successful and the customer is projected to be picked up within the maximum waiting time, so before the second definitive waiting time $t_{\text{w},2}$ is reached, the request is accepted. In this case, the operator assigns a vehicle of the fleet to the request and sends an offer to the customer. This initial offer contains a pickup time window and is associated with an offer quality q_1. If the heuristic does not find a feasible solution, the request is instead rejected. It is deleted from the system and the customer is sent a notification of the rejection. This terminates the process.

If an offer is sent to the customer, a decision-making process is kicked off as described above (box 3). Depending on the offer quality, this process can take up to the maximum decision duration $t_{\text{d,max}}$. A positive decision by the customer results in an acceptance notification being sent to the service operator. If the decision-making process ends with the customer rejecting the offer, the operator is informed about this decision and deletes the request from the system. This also means, that the request is removed from the task queue of the currently assigned vehicle, which consequently will instead head to the location of the next task in the list or become idle if no task is left.

During the time the decision-making process is running, the operator is not inactive in regard of this exemplary request. The vehicle that was originally assigned to the request based on the solution found with the heuristic is executing its task queue and eventually heads towards the pickup location of the user. In the meantime, after each optimization period t_p, a global optimization of all assignments takes place, including the example request (box 4). As a result, the assignment of the request can change once or multiple times, before the request is locked. This event is triggered by the operator at the point in time when the projected pickup time is exactly t_lock ahead, as described in 3.1. The changes of assignments between the time the request is sent and locked are not communicated to the user. Instead, an offer is sent when the assignment is locked, including the final pickup time projection, associated with the new offer quality q_2.

Another decision-making process begins, in which the customer chooses to either accept or reject this final offer (box 5). Again, the decision is sent to the operator and the request is handled accordingly. If the offer is accepted, the assigned vehicle approaches the pickup location and picks up the waiting customer, which is the end of the interactive decision-making process in the DCM.

Note that the operator assumes all offers made to users are accepted. This may result in an overestimation of the number of requests that need to be served, and is therefore considered to be a conservative approach by the operator when it comes to assignment strategies. This can lead to situations in which a subsequent offer to a customer is made with a shorter waiting time than the initial one, because another customer that would have been served before, rejected their offer in the meantime. If the final offer is already made to and accepted by a user, and the assigned vehicle arrives at the pickup location earlier than offered, the simulation model assumes the vehicle to wait until the planned pickup time.

6.2 Model Parameters and Key Performance Indicators

In the DCM, the number of parameters, variables and KPIs increases compared to simulations with the CCM. The constant model parameters are the same throughout all evaluations with the DCM. They are either determined empirically or based on values from the literature.

Variable model parameters vary between simulations in order to examine their respective impact on the system performance and KPIs. In addition to the performance indicators introduced in Section 3.4.3, in the DCM a number of new KPIs are evaluated to measure the differences during the decision-making process.

Constant Parameters

Values of model parameters that are constant in all simulations with the DCM are chosen to represent an average customer as closely as possible. Because of the complexity of human decision-making processes and the huge variety of personal preferences and characteristics of ODM service users, these values need to be treated as best efforts to simplify the customer model in a reasonable way.

The reaction time t_{re} and the maximum decision duration $t_{\mathsf{d,max}}$ are both set to constant values that have a direct impact on the average decision duration t_{d}. The time necessary for service users to comprehend the offer and interact with the device they use to communicate with the service provider is summarized under t_{re} and set to 2 s. This amount of time is plausible for most customers who are assumed to be most likely to use the service. In the model, this time passes before the actual decision-making process is started as described in Section 6.1.

The value of $t_{\mathsf{d,max}}$ is set to 60 s. This implies that any decision-making process that is not finished after one minute, is stopped and the user decides if the offer is accepted or not depending on the current value of D_t. This prevents the customer model to end up in an infinite decision-making process and can easily be imagined to be implemented in the service application as a countdown that encourage the user to make a decision in a timely manner.

During the decision-making process modeled with the DCM, the central variable is the decision state D_t. Its progression is randomized. However, the stochastic behavior is determined by some constant model parameters.

First, the initial value D_0 at the beginning of each decision-making process is defined to be zero, independent of any previous decisions made by the same customer. This choice of D_0 implies that every customer is absolutely unbiased with respect to historic decisions and motivates each decision solely on the current offer's quality.

The step size Δt of the random walk states the granularity of the decision-making process. The smaller the values of Δt, the closer the model represents a continuous decision-making process. A smaller Δt also implies more steps per simulation step t_{step}, which means that the computation time tends to grow with decreasing Δt. Finding a suitable value for the step size of the random walk therefore is a trade-off between the realism and the applicability of the DCM. In this work, Δt is adopted from [Yu and Hyland, 2020] and set to 0.01 s.

Another parameter that affects D_t is the feedback rate S. It determines the impact of D_t on $D_{t+\Delta t}$ and therefore represents the decisiveness of service users during the decision-making process. High values of S imply that each individual step in the random walk is very much depending on the recent history of preceding steps made in the same decision-making process. On the other hand, if S is very small, the valence V is the dominant element of each iteration of D_t, which mechanically tends to produce random walks meandering around D_0 with rare occasions of high altitudes of D_t in any direction. The value of $S = 0.991$ used in this work is adopted from [Yu and Hyland, 2020] and empirically found to produce reasonable decision durations.

Variable Parameters

The variable parameters of the DCM evaluated in this work can be split in two categories: "global" variables are changed between simulations but are considered constant during each individual scenario. In contrast, "inherent" parameters vary within a certain value range for each individual service user, even in the same instance of simulation.

Global variables of the DCM investigated in this work are the decisive waiting times $(t_{\text{w},1}, t_{\text{w},2})$. This parameter pair determines the calculation of the offer quality q (see Equation 6.1) and is therefore a fundamental part of the DCM.

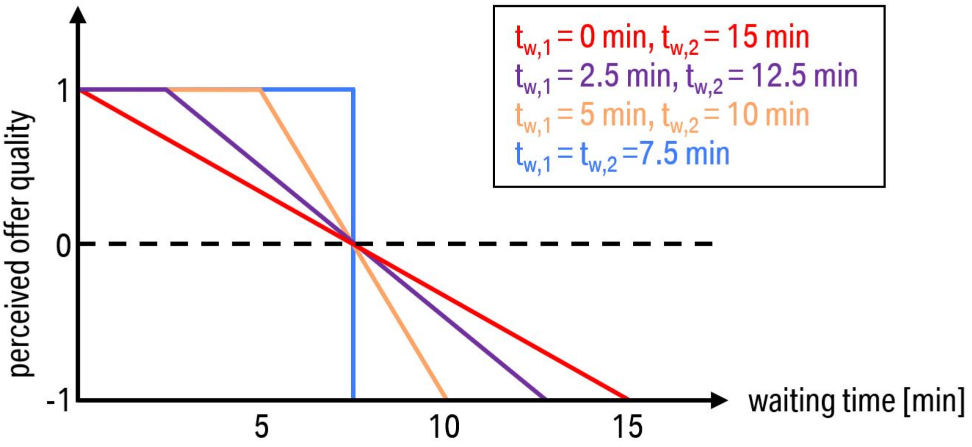


Figure 6.3: Offer quality assessment in DCMs with four different pairs of values for $(t_{\text{w},1}, t_{\text{w},2})$.

Figure 6.3 shows the offer quality assessment with the four different pairs of $(t_{\text{w},1}, t_{\text{w},2})$ evaluated in this work. The version of the DCM with $t_{\text{w},1} = 0$ s and $t_{\text{w},2} = 900$ s (15 min) is depicted in red. Unlike the other versions, there is no plateau of high-quality offers for sufficiently short waiting times. Instead, q drops to -1 linearly with increasing waiting times until $t_{\text{w}} = 15$ min, which is the longest decisive waiting time evaluated.

In the model instance depicted in purple, the offer quality associated with waiting times of 150 s (2.5 min) is set to 1, meaning that users do not make a difference in their assessment of the offer and are very likely to accept it. The linear decrease of q stops at $t_{\text{w},2} = 750$ s (12.5 min).

Similar to the purple example, in the yellow version of the DCM there is a plateau of $q = 1$. Here, $t_{\text{w},1} = 300$ s (5 min), though, and the drop to $q = -1$ is sharper, with $t_{\text{w},2} = 600$ s (10 min). This version of the DCM is used in the PSAs in Sections 6.3.2 and 6.4.2.

The fourth and final version of the DCM is shown in blue. In this version, both decisive waiting times are equal and set to 450 s (7.5 min). This translates to a function of $q(t_{\text{w}})$ that is equivalent with a step function. All waiting times of $t_{\text{w}} \leq 450$ s are associated with an offer quality of $q = 1$, while for $t_{\text{w}} > 450$ s, $q = -1$. This special version of the DCM is also the closest analogy possible with the DCM to the CCM described in Section 3.3. In both, there is no gradual decline of the offer quality with increasing waiting times, but instead effectively one single maximum waiting time that separates offers that are (very likely to be) accepted and those that are rejected by the operator.

All four versions of the DCM share the property that offers implying waiting times of $t_{\text{w}} < 450$ s are associated with positive values of q and therefore perceived positively by customers. The chance for such offers to be accepted is generally higher than to be rejected by the user. Also, the area below each of the four graphs is the same. Since this area can be translated to the average probability of each offer to be accepted in each of the versions of the DCM, a comparison of these versions – together with the base line defined by the CCM – is sound.

Inherent parameters of the DCM are properties that are different for each particular customer. They represent the variety of characteristics service users have as individuals. In order to keep the DCM comprehensible and its parameter changes traceable, this work focuses on two model parameters and their impact on the system performance.

The diffusion rate ϕ determines the variance of the random term ϵ in the valence V during each step of the random walk. High values of ϕ imply a widely spread distribution of random numbers around the mean, which is set to 0. On the other hand, small ϕ increase the probability of ϵ to be very close to the mean.

Service users with high ϕ-values therefore are more often connected to decision-making processes that deviate randomly from the direction implied by the offer quality. Such customers can be described as rather erratic in their behavior and the decision made by them tend to be less closely related to quality of the offer presented to them. The range of possible values of ϕ is set to 2 to 4. The standard value used outside of the PSAs is $\phi = 3$.

The other inherent variable is the decision threshold θ. This parameter specifies the absolute value D_t needs to reach for the customer to make a decision. As soon as $D_t \geq +\theta$, the offer is accepted. If instead $D_t \leq -\theta$, the decision of the service user is to reject the offer. If D_t does not reach any of the decision thresholds before the maximum decision duration, the decision is made based on the decision state at this point $D_{t_{\text{d,max}}}$. If the assigned vehicle reaches the pickup location before a decision is made by the customer, the offer is always accepted and the user boards the vehicle.

High values of θ prevent decisions to be made after only a few steps of the random walk, possibly with very widely spread random numbers involved. Instead, users with such θ-values tend to make thoughtful decisions, very much based on the quality of the offer presented to them. On the other hand, users with small values of θ make more impulsive decisions, that may take less time to make and are more likely to be unrelated to the value of q. In the PSAs of the DCM, θ ranges from 30 to 50. The standard value is set to 50. As an overview, all constant and variable parameters of the DCM are listed in Table 6.1.

Parameter	**Symbol**	**Values**
Decisive waiting times	$(t_{\text{w},1}, t_{\text{w},2})$	(0 s,900 s), (150 s,750 s), (300 s,600 s), (450 s,450 s)
Offer quality	q	Depending on $t_\text{w}, t_{\text{w},1}, t_{\text{w},2}$
Random walk step size	Δt	0.01 s
Initial decision state	D_0	0
Feedback rate	S	0.991
Diffusion rate	ϕ	2-4 (standard: 3)
Random term	ϵ	Normally distributed, with mean 0 and variance ϕ
Valence	V	Depending on q and ϵ
Decision state at $t + \Delta t$	$D_{t+\Delta t}$	Depending on S and V
Decision threshold	θ	30-50 (standard: 50)
Reaction time	t_re	2 s
Max. decision duration	$t_{\text{d,max}}$	60 s

Table 6.1: Parameters and variables of the DCM.

Key Performance Indicators

To evaluate the impact of the inherent parameters on the decision-making processes, new KPIs need to be defined. Note that each of the following KPIs is measured for the two separate decision-making processes considering both the initial offer with offer quality q_1, as well as the second offer associated with q_2. In KPIs related to the second offer, only customers are taken into account who accepted the first offer, so all percentages are relative to this number. KPIs related to the o-th offer are referred to with superscript o (e.g., x^o).

The first DCM-specific KPI is the acceptance rate r_a^o. This KPI differs from the "percentage of requests served" introduced in Section 3.4.3 in several ways. In the DCM, it is not only the operator that decides if a request is accepted. Instead the customer has to accept the offer based on its quality. Hence, r_a^o is an indicator of the average offer quality q_o. Especially the acceptance rate of second offers r_a^2 is of high interest to the operator, because a bad performance in this KPI means many late rejections, which can imply a lot of unnecessary trips and detours on top of the lost paying customer.

Because one of the features of the DCM is the possibility of unexpected decisions by customers, both the rates of unexpected acceptances $r_\mathsf{a,u}^o$ and rejections $r_\mathsf{r,u}^o$ are measured and compared. High values in these categories imply the DCM tends to make more random decisions, while low values are associated with very predetermined decision-making processes.

Besides the fact that decisions about accepting or rejecting requests are no longer made exclusively by the operator, another difference between the CCM and the DCM is the time it takes for such a decision to be made. In the CCM, the operator can be certain that an accepted request will be served immediately after the initial heuristic finds a feasible solution. In the DCM, every decision takes a certain amount of time, referred to as decision duration t_d^o. The longer the average decision-making process takes, the longer the operator is uncertain if an offer is accepted or not, again potentially causing longer trips towards pickup locations of users that eventually might reject the offer.

All these KPIs are relevant in the evaluation of the inherent variable parameters ϕ and θ in Sections 6.3.2 and 6.4.2.

6.3 Evaluation of Diffusion Customer Models in the Ride Hailing Use Case

The impact of the DCM on ODM service models is first evaluated in the ride hailing use case. The service model of choice is the 2-step service model introduced in Section 3.1. The methods for heuristic and globally optimized request assignments are unchanged from those described in Section 4.1. In the first of the upcoming sections, four different versions of the DCM are compared to the CCM presented in Section 3.3. The DCMs vary in their respective decisive waiting times $(t_{\mathsf{w},1}, t_{\mathsf{w},2})$ as shown in Figure 6.3. At the end of the section, a PSA is conducted for the two model-specific parameters ϕ and θ.

6.3.1 Comparison of Customer Models

Like in the comparison of ride hailing service models in Section 4.2.1, the figures presented in the following section are split in two parts. On the left hand side, the average total values of the respective KPI are shown for the simulations run for all dates with five different customer models, four versions of the DCM and the CCM, acting as a base line. On the right of each figure, the delta to average is presented. These values are calculated by first averaging over the differences between the models at each individual simulation date and then averaging over these differences at each date. This method allows to focus on the model-related effects on the KPI instead of the comparably big variations due to changing fleet sizes or demand over various days.

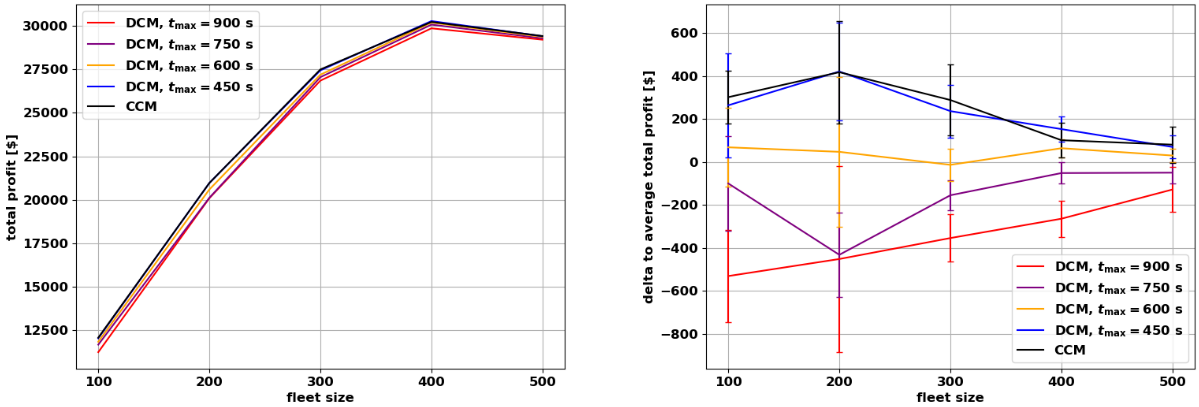
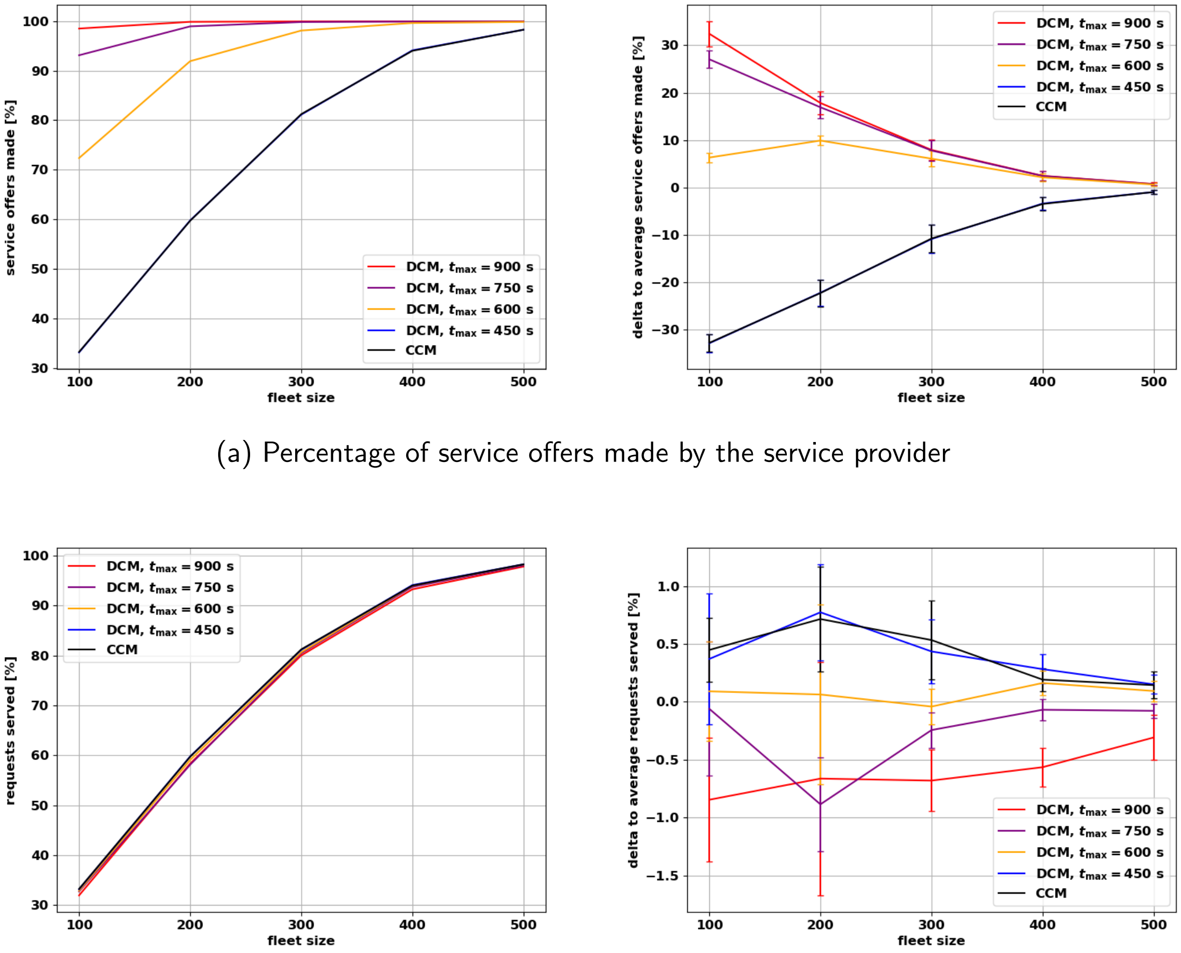


Figure 6.4: Profit in US-Dollars generated in simulations with various diffusion customer model parameters in the ride hailing use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

The first KPI investigated is the profit generated with each of the models. Figure 6.4 shows that the general dependency on the fleet size is the same for all customer models, with a maximum profit of around \$30 000 in scenarios with 400 vehicles. The delta to average reveals that the service profitability decreases with increasing values of t_{max}, which in the DCM is equivalent with the second decisive waiting time $t_{\text{w},2}$. As expected, the difference between the CCM (black) and the DCM with $t_{\text{max}} = 450\,\text{s}$ (blue) is negligible, because the latter is effectively the representation of the former in the context of a diffusion model. The difference between the DCMs with $t_{\text{max}} = 900\,\text{s}$ (depicted in red) and $t_{\text{max}} = 450\,\text{s}$ is largest in scenarios with small fleet sizes. Considering fleet sizes of 100 vehicles, the gap is \$795 ± 456, which means the profit in the red scenarios is up to (6.6 ± 3.8) % smaller than scenarios with the DCM closest to conventional customer models.

How the profitability of a service is connected to the number of service offers made and the percentage of requests served is presented in Figure 6.5. Figure 6.5b shows a similar relation between the customer models as in Figure 6.4. In scenarios with the CCM and its equivalent version of the DCM, the percentage of requests served by the service is highest, independent of the fleet size. The gap to the DCM with $t_{\text{max}} = 900\,\text{s}$ is between (0.46 ± 0.27) % in scenarios



(a) Percentage of service offers made by the service provider

(b) Percentage of requests served by the service provider.

Figure 6.5: Percentage of requests served and service offers made in simulations with various diffusion customer model parameters in the ride hailing use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

with 500 vehicles and $(1.22 \pm 1.10)\,\%$ in scenarios with 100 vehicles.

The decline in percentage of served requests with increasing values of t_{max} is a consequence of the way decisions are made in the DCM and how the operator makes offers to service users. In scenarios with larger differences between the decisive waiting times $t_{\text{w},1}$ and $t_{\text{w},2}$, more requests are accepted and more service offers are made, which can be seen in Figure 6.16a. A large portion of these additional offers is made with negative values of q, as the operator tries to accept as many requests as possible, even if the offer quality is perceived as bad. This means parts of the service fleet are assigned to these requests, reducing the number of vehicles available to serve upcoming requests, and thereby further decreasing the average offer quality. Most of such offers will be rejected, though, and the associated requests end up not being served.

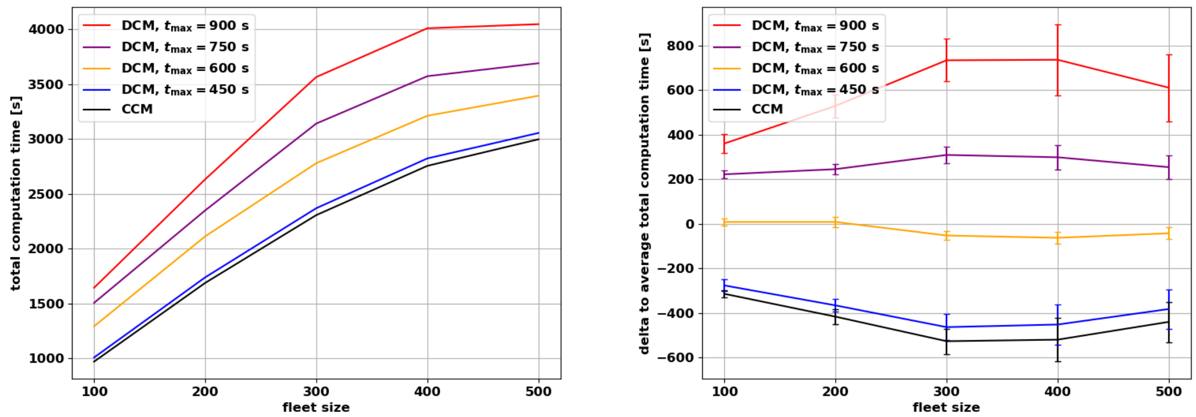
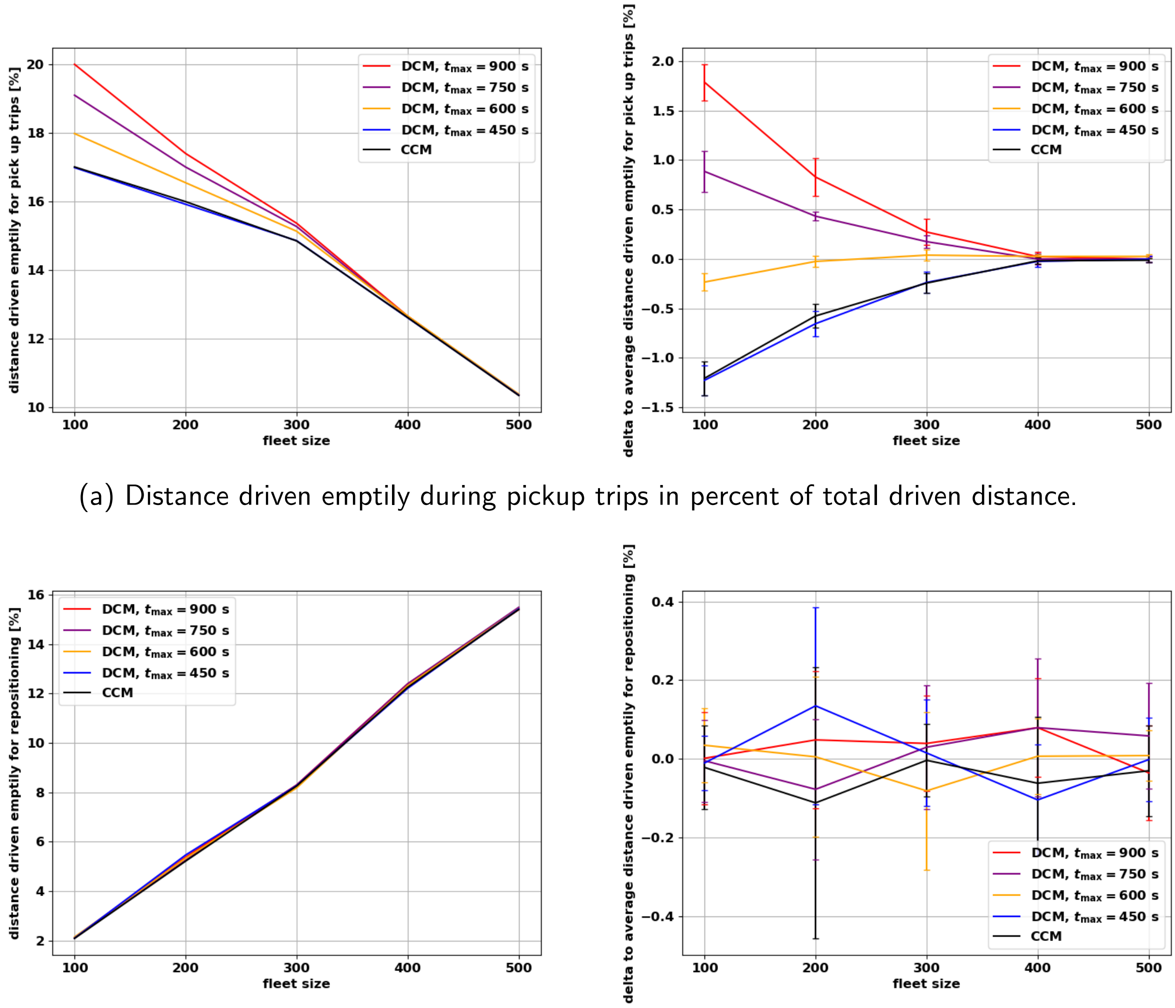


Figure 6.6: Computation time in seconds in simulations with various diffusion customer model parameters in the ride hailing use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

The total computation times shown in Figure 6.6 reveal another clear difference between the versions of the DCM and the CCM. Again, the difference between the models correlates to the value of t_{max} associated with the respective version of the DCM: the larger this parameter, the longer the average computation time to simulate the scenario. This observation can be explained by the added computation time necessary to simulate the decision-making processes of each customer. As explained in Section 6.1, decision-making processes connected to offers with qualities q around 0 tend to take longer than those with higher absolute values. Due to the form of offer quality assessment in each of the four versions of the DCM shown in Figure 6.3, the DCM-version with $t_{\text{max}} = 900\,\text{s}$ (red) includes the most such instances, followed by the one with $t_{\text{max}} = 750\,\text{s}$ (purple) and $t_{\text{max}} = 600\,\text{s}$ (yellow). The version in which both decisive waiting times are equal ($t_{\text{max}} = 450\,\text{s}$, blue) only includes offers with absolute q-values of 1. The decision-making processes made in these scenarios are all very short, therefore the added computation time due to the random walk of the decision-making process is negligible, which can be seen by the narrow offset of computation times in scenarios with this version of the DCM and the CCM.

The implications of the DCM can also be observed in Figure 6.7, in which the empty distances driven during pickup trips (Figure 6.7a) and during repositioning trips (Figure 6.7b) are shown. The differences in distances driven by vehicles on their way to pickup locations are greatest between the versions of the DCM with $t_{\text{max}} = 900\,\text{s}$ and $t_{\text{max}} = 450\,\text{s}$, respectively. It is $(3.01 \pm 0.33)\,\%$ in scenarios with fleet sizes of 100 vehicles and drops to essentially $0\,\%$ in scenarios with 500 vehicles.

This observation is in line with the expectation that the DCM is able to model additional empty mileages that is produced when vehicles on their way to a pickup location stop in their tracks when a customer rejects an offer. Because such events take place more often in scenarios in which the average offer quality is low, small-fleet scenarios show a larger discrepancy in distances driven emptily. The fact, that the empty mileage due to repositioning is virtually the same for all customer models underlines the impact of additional empty mileage



(a) Distance driven emptily during pickup trips in percent of total driven distance.

(b) Distance driven emptily during repositioning trips in percent of total driven distance.

Figure 6.7: Distances driven emptily in percent of total mileage in simulations with various diffusion customer model parameters in the ride hailing use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

during pickup trips and supports this explanation.

The average user waiting times from time of request to pickup are presented in Figure 6.8. The average total waiting times decrease with increasing fleet sizes in simulations with all customer models, dropping from over 300 s in scenarios with 100 vehicles to less than 140 s in the largest instances with 500 vehicles. The delta to average reveals that the difference between the models also declines with growing fleet sizes. The shortest waiting times are reached in scenarios with the version of the DCM in which $t_{\text{max}} = 900\,\text{s}$, which produces waiting times that are (3.3 ± 1.7) s to (27.7 ± 2.3) s shorter than in scenarios with the CCM and fleet sizes of 500 and 100 vehicles, respectively.

Shorter pickup waiting times are a follow-up effect of the DCM's impact on which request are served and which are not. Offers that are received as bad are more likely to be rejected.

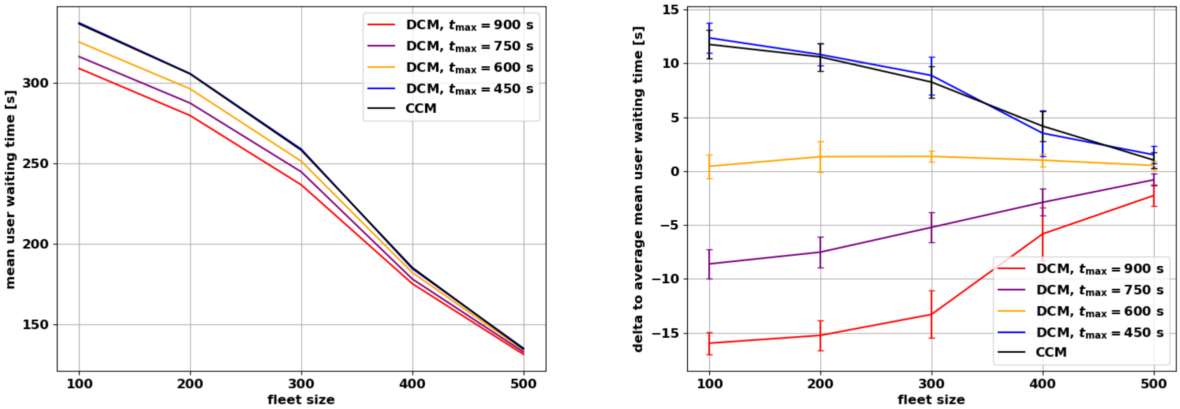


Figure 6.8: User waiting times in seconds in simulations with various diffusion customer model parameters in the ride hailing use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

In scenarios with high demands (small fleet sizes) and large differences between the decisive waiting times in the DCM, this leads to fewer requests served as discussed above. However, the users that are served, on average have a shorter waiting time, because such users accepted two offers, neglecting those who are picked up before their decision-making processes finalizes (which also contribute to shorter average waiting times). This filters bad offers not only once but twice, shifting the average offer quality – and thereby the average pickup waiting time – of served requests towards more favorable values.

6.3.2 Model Parameter Sensitivity Analyses

As explained in Section 6.2, two of the DCM-related parameters are evaluated in more detail. The decision threshold θ and the diffusion rate ϕ are both closely connected to the decision-making process simulated in the DCM. They represent model parameters with an impact on the degree of realism of the customer model. Furthermore, both values reflect descriptive characteristics of individuals using the service, which is an important feature of the DCM compared to other customer models, like the CCM.

The results presented in the following section are conducted during simulations of the dates between November 12 to 18, 2018. The fleet size is fixed to 300 vehicles, and the 2-step service model is used in all instances. The version of the DCM used in these simulations is chosen to be the one with decisive waiting times $t_{\mathrm{w},1}$ =300 s and $t_{\mathrm{w},2}$ =600 s, represented in yellow in Section 6.2. Instead of running individual simulations in which each customer has the same set of parameters (θ, ϕ) that is adjusted after every simulation, the parameters are treated as inherent. This means every single service request is associated with a set of (θ, ϕ), in addition to variables like the time of request and number of passengers. The values of θ and ϕ are randomly chosen within the range of 30 to 50 in integer steps and 2.0 to 4.0 with steps of 0.1, respectively.

This not only allows to significantly reduce the number of simulations needed to evaluate the examined parameter space, but also represents realistic scenarios in which the service operator is confronted with a variety of distinct customer behaviors and characteristics.

As also described in Section 6.2, the KPIs measured in this PSA are the total acceptance rates r^o_{a}, the rates of unexpected acceptances $r^o_{\mathsf{a,u}}$ and rejections $r^o_{\mathsf{r,u}}$ and the average duration of decision-making processes t^o_{d}, measured for the o-th offer made by the service operator, respectively. The figures presented are split in two parts. On the left, the figures relate to the respective KPI measured for first offers, the ones on the right to the same KPI measured for second offers. All shown figures include overviews of the KPI measured at each of the simulation dates, together with the average values depicted as bold black lines.

Decision Threshold

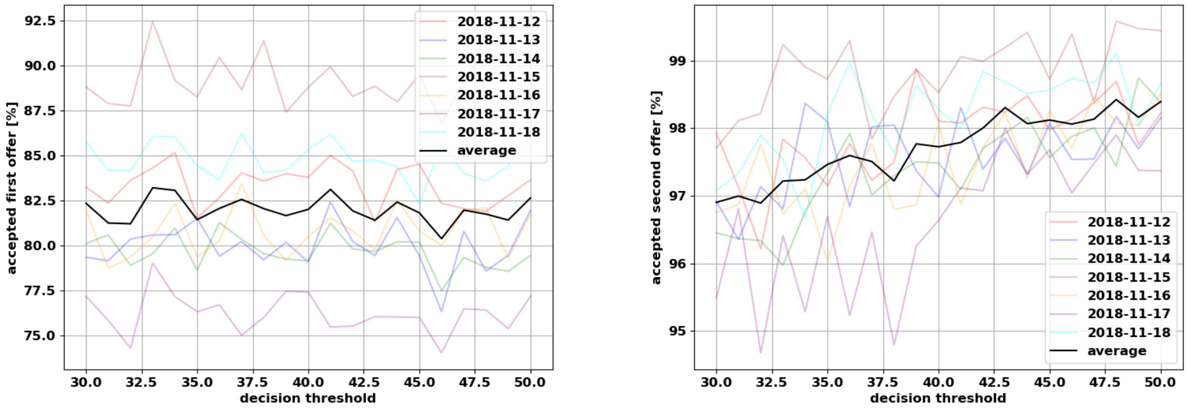
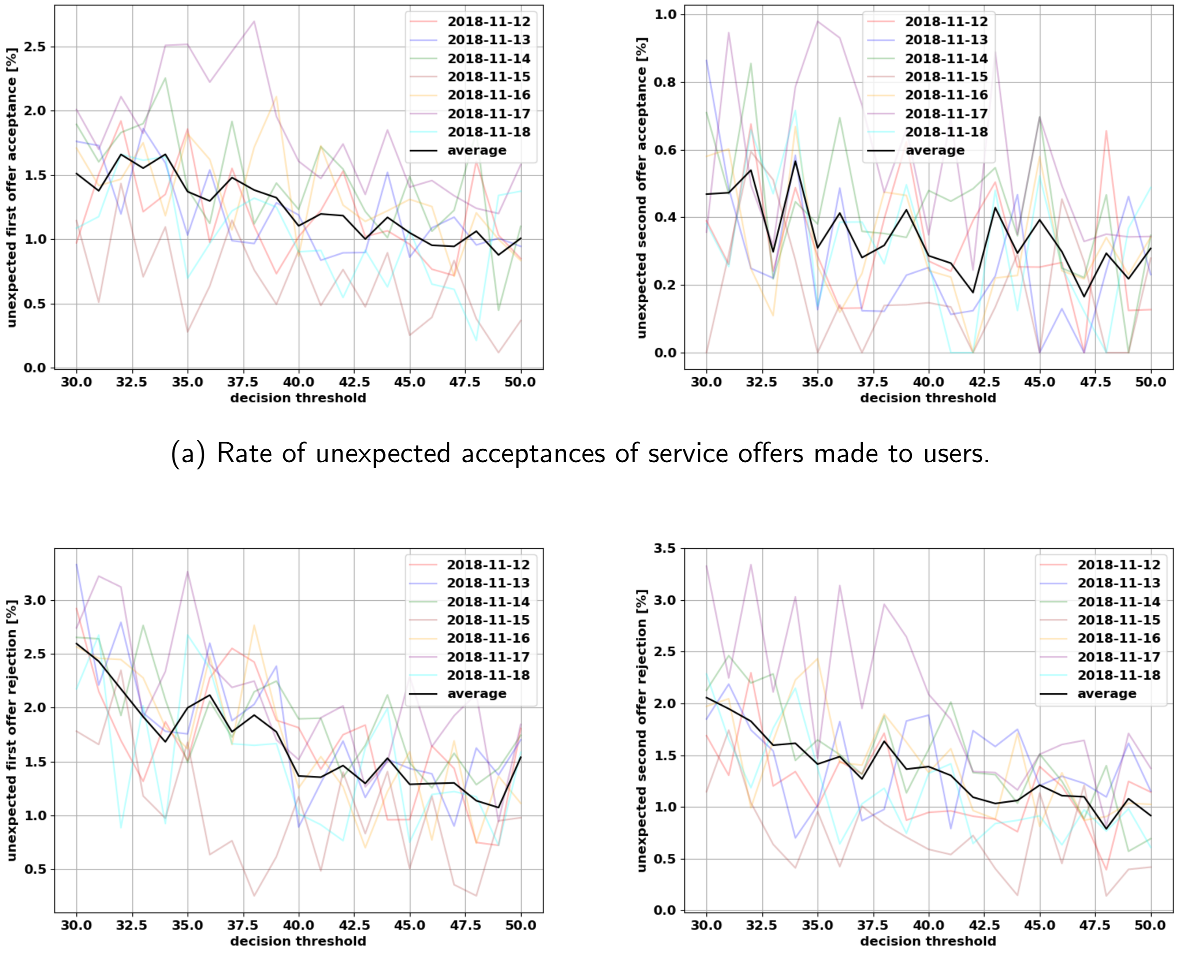


Figure 6.9: Acceptance rates of service offers made to users in percent of all such offers made in simulations with decision thresholds between 30 and 50 in the ride hailing use case. Values for all simulation dates and the average values. Left: related to first offers. Right: related to second offers.

The acceptance rates of first and second offers in relation to customers' decision thresholds are presented in Figure 6.9. As shown on the left, the average value of r^1_{a} fluctuates between values of 80.38 % and 83.21 % with no apparent dependency on θ. Unlike the acceptance rate of first offers, the probability of second offers to be accepted increases with increasing decision thresholds, though. For customers with $\theta = 30$, $r^2_{\mathsf{a}} =$ 96.90 % on average, rather steadily increasing to $r^2_{\mathsf{a}} =$ 98.40 % for customers with $\theta = 50$. Note that this percentages are relative to the number of all customers that accepted the initial offer.

The difference between the average acceptance rates of the two offers can be explained by the difference of the average offer qualities associated with first (average $q_1 = 0.61$) and second offers (average $q_2 = 0.83$). Since q_1 heavily depends on the availability of service vehicles and the offered pickup waiting times, the spectrum of offer qualities is wider, including many offers perceived as bad. The value of the decision threshold hence has a marginal effect on r^1_{a}, because any impact on offers with $q_1 > 0$ is negated by the corresponding inverse effect on decision-making processes about offers with $q_1 < 0$.

On the other hand, subsequent offers, which are only made to users that already accepted the initial one, have a higher quality q_2 on average and fewer offers are perceived as bad, because the first decision-making process effectively filters them. This not only explains the higher total values of r^2_{a}, but also the correlation between the acceptance rate and the decision threshold. The drift of the random walk which simulates the decision-making process in the DCM is determined by the offer quality. If the majority of offer qualities is positive, more random walks drift towards the positive decision threshold $+\theta$ associated with accepting the offer. The higher the value of θ, the lower the probability that random deviations (represented by ϵ in Equation 6.2) from the direction defined by the drift q affect the random walk enough to reach a decision state $D_t \leq -\theta$, which implies a rejection of the offer.



(a) Rate of unexpected acceptances of service offers made to users.

(b) Rate of unexpected rejections of service offers made to users.

Figure 6.10: Rate of unexpected decisions made by users in percent of all such offers made in simulations with decision thresholds between 30 and 50 in the ride hailing use case. Values for all simulation dates and the average values. Left: related to first offers. Right: related to second offers.

This reasoning is supported by the rates of unexpected decisions made by customers, which can be seen in Figure 6.10. In Figure 6.10a, the percentage of unexpected acceptances is presented for first and second offers, Figure 6.10b shows the corresponding rates of unexpected rejections.

While the average rates of unexpected decisions – both acceptances and rejections – drop with increasing decision thresholds, the differences in total percentages of such decisions, also between first and second offers, indicate two things: (1) because the average offer qualities are positive, there is an overweight of unexpected rejections relative to unexpected acceptances, because of the higher number of offers that are expected to be accepted (due to the positive average offer quality). This trend is more obvious in the rates of unexpected responses to second offers, because here the average offer quality is higher than for first offers. (2) The smaller overall scale of $r^2_{\text{a,u}}$ (between 0.17 % and 0.57 %) compared to $r^2_{\text{r,u}}$ (0.79 % to 2.1 %) also implies a lower slope between low and high values of θ.

These two observations combined explain the overall increase of accepted second offers by service users with higher decision thresholds. Unlike first offers, the effects of unexpected acceptances and rejections on the overall percentage of accepted offers do not cancel each other out to a point, at which the difference is smaller than the stochastic fluctuations caused by the randomness of the evaluated customer model. In fact, the majority of rejected second offers (the gap to 100 % on the right side of Figure 6.9) is found to be unexpected. So the impact of the decreasing value of $r^2_{\text{r,u}}$ with increasing θ is more noticeable than for first offers, especially since the inverse effect of $r^2_{\text{a,u}}$ is significantly smaller because of its smaller overall scale.

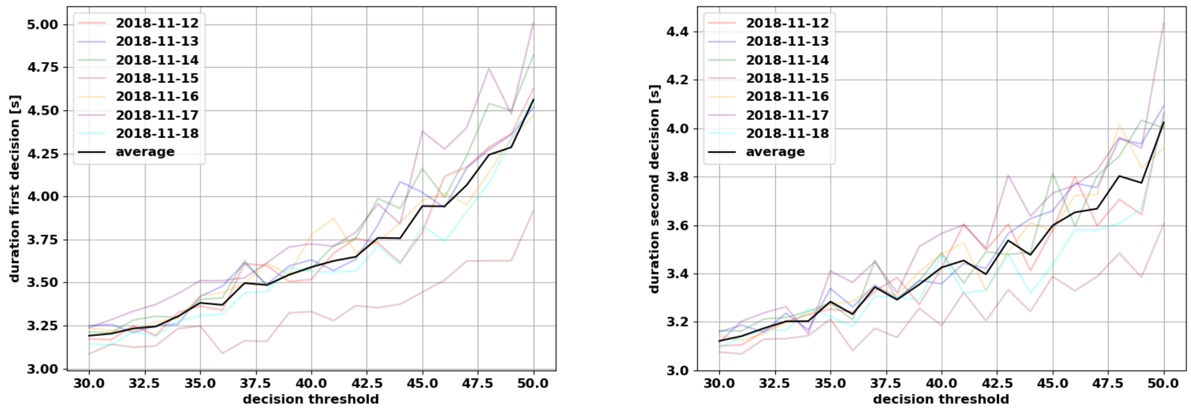


Figure 6.11: Decision duration in seconds in simulations with decision thresholds between 30 and 50 in the ride hailing use case. Values for all simulation dates and the average values. Left: related to first offers. Right: related to second offers.

The relation of the average durations of decision-making processes and decision thresholds is shown in Figure 6.11. As expected, the higher the decision threshold of a user, the longer the average time it takes between receiving an offer and making a decision about it. This observation can be made for both t^1_{d} and t^2_{d}. The average value of t^1_{d} grows from 3.19 s for users with $\theta = 30$ to 4.56 s when $\theta = 50$, while t_{d}^{2} ranges from 3.12 s to 4.02 s, for users with $\theta = 30$ and $\theta = 50$ respectively.

The slightly shorter decision durations of second offers compared to first offers for users of identical θ can again be explained by the higher average offer quality of second offers q_2, which leads to stronger drifts of the random walk involved in the decision-making process.

Diffusion Rate

Like the decision threshold θ, the diffusion rate ϕ directly affects decision-making processes in the DCM. Recall that ϕ is the standard deviation of the random term ϵ in each iteration of the random walk. This means a higher value of ϕ is associated with more random decision-making processes.

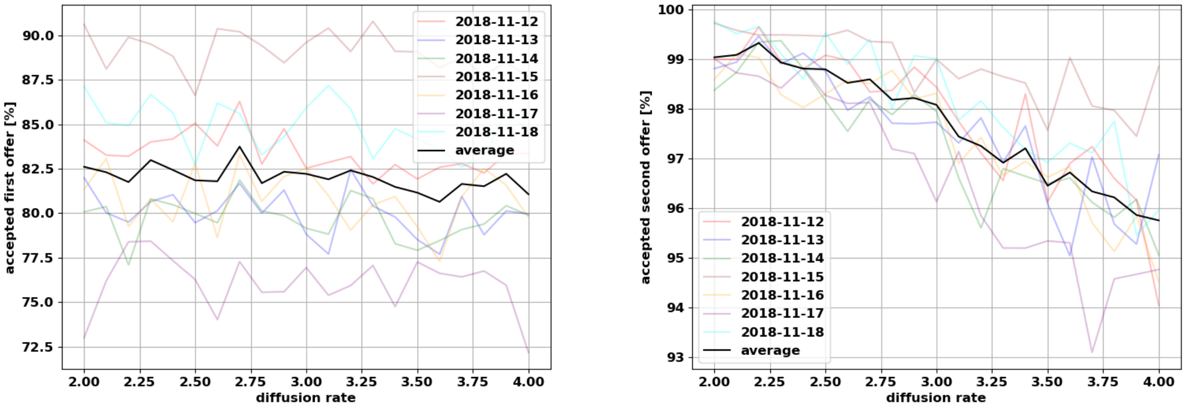
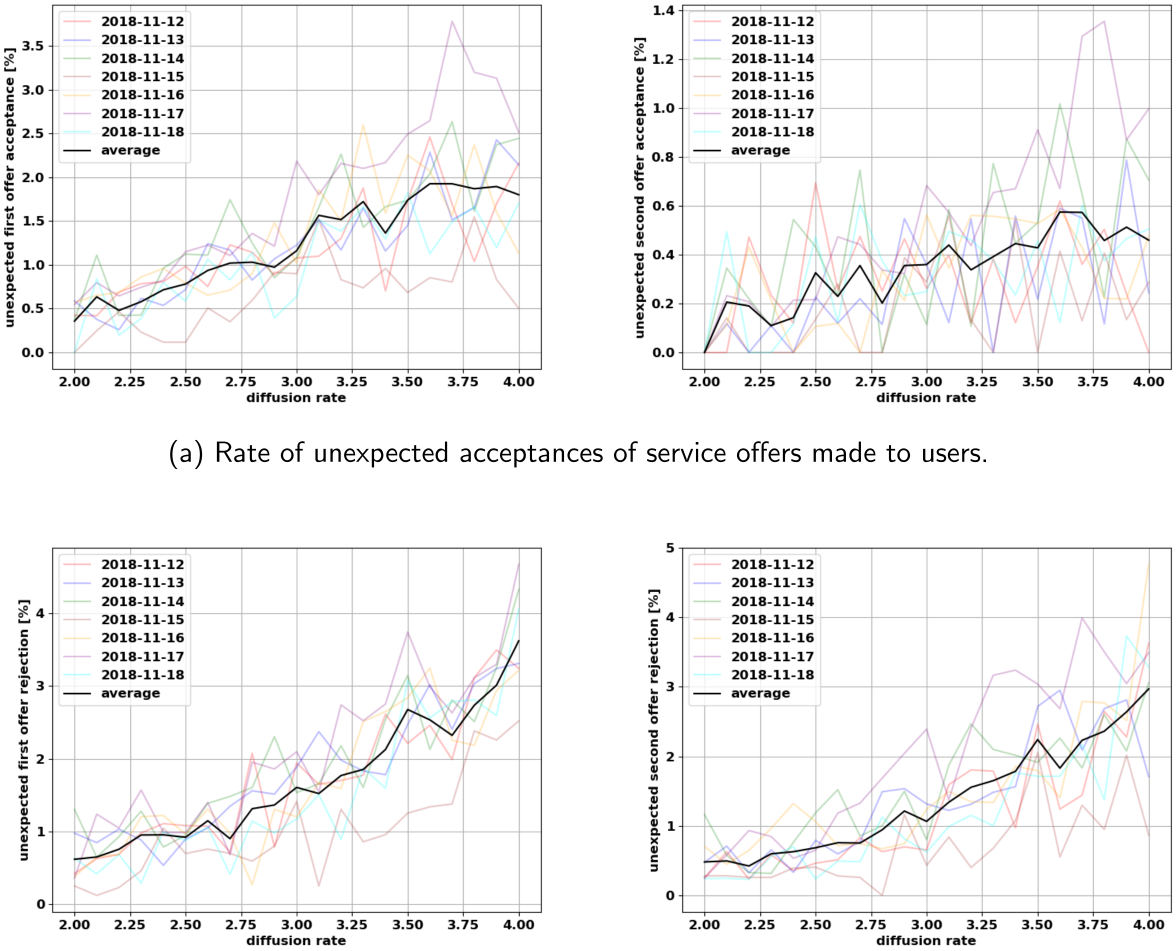


Figure 6.12: Acceptance rates of service offers made to users in percent of all such offers made in simulations with diffusion rates between 2.0 and 4.0 in the ride hailing use case. Values for all simulation dates and the average values. Left: related to first offers. Right: related to second offers.

Figure 6.12 presents the influence on the acceptance rate of first and second offers in dependence to ϕ. On the left, r_{a}^{1} is shown to fluctuate between 80.64 % and 83.74 % without an apparent correlation with the diffusion rate. Similar to the decision threshold, this changes for the acceptance rate of second offers. However, unlike in the case of θ, the highest values of r_{a}^{2} are reached with low diffusion rates (up to 99.32 %). Users with higher diffusion rates have a noticeably lower probability of accepting second offers (r_{a}^{2} =95.75 % with $\phi = 4.0$). This trend can be explained with the same reasoning as above and becomes clearer in Figure 6.13, in which the rates of unexpected decisions are presented for first and second offers.

In Figure 6.13a, the percentages of unexpected acceptances of first and second offers is presented. The value of $r_{\mathrm{a,u}}^{1}$ grows with increasing diffusion rate, from 0.36 % for users with $\phi = 2.0$ up to between 1.80 % and 1.93 % for users with diffusion rates of 3.6 or higher. For second offers, $r_{\mathrm{a,u}}^{2}$ also increases with the diffusion rate, although on a smaller scale (between 0.0 % and 0.57 %.



(a) Rate of unexpected acceptances of service offers made to users.

(b) Rate of unexpected rejections of service offers made to users.

Figure 6.13: Rate of unexpected decisions made by users in percent of all such offers made in simulations with diffusion rates between 2.0 and 4.0 in the ride hailing use case. Values for all simulation dates and the average values. Left: related to first offers. Right: related to second offers.

When comparing these numbers with the rates of unexpected rejections, shown in Figure 6.13b, this gap between first and second offers is smaller. This means the difference between unexpected acceptances and rejections is noticeable larger for second offers than for first ones. Because of the lower gradient of $r^2_{\mathrm{a,u}}$ compared to $r^2_{\mathrm{r,u}}$, this difference further grows with increasing diffusion rates, which directly translates to the trend of r^2_{a} decreasing with increasing ϕ, observed in Figure 6.12.

Figure 6.14 shows the average decision durations for first and second offers in relation to the diffusion rate. The overall difference between the relations of ϕ and t^1_{d} and t^2_{d} is very small, as observed for θ. The correlation is inverse, though, for similar reasons as explained above. With increasing ϕ, t^1_{d} (t^2_{d}) drops from a maximum of 4.68 s (4.08 s) to 3.24 s (3.20 s).

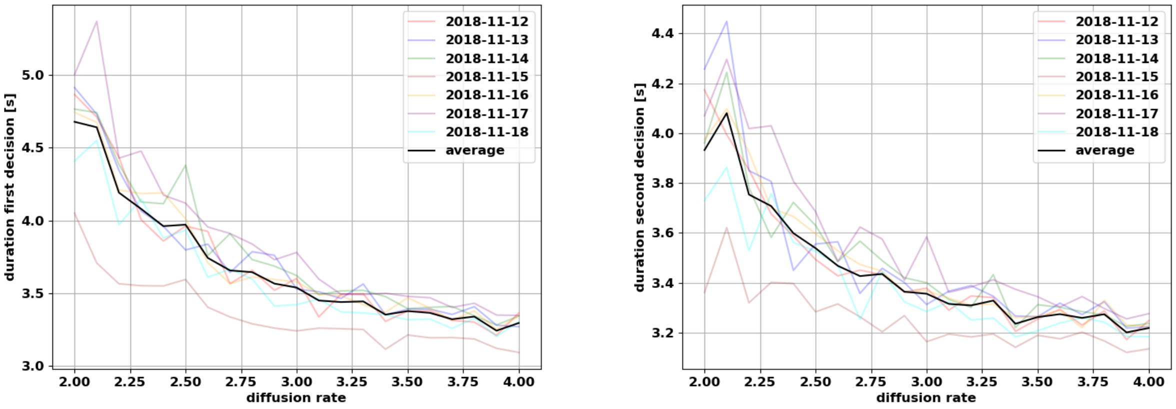


Figure 6.14: Decision duration in seconds in simulations with diffusion rates between 2.0 and 4.0 in the ride hailing use case. Values for all simulation dates and the average values. Left: related to first offers. Right: related to second offers.

The results of this PSA are generally in accordance with the anticipated findings. Users of ride hailing services that are modeled to be more thoughtful and less erratic in their decisions (high θ, low ϕ) are expected to make unexpected acceptances and rejections less frequently. On the other hand, such customers also tend to take a little longer for decisions, which is undesirable for service operators, because it translates to potentially longer empty trips of service vehicles that end up being not needed in case the offer is rejected. The PSA can therefore be considered to validate the model adequately.

6.4 Evaluation of Diffusion Customer Models in the Ride Pooling Use Case

After the evaluation of the ride hailing use case, this section focuses on the impact of the DCM on the system performance in the ride pooling use case. Again, the service model used in this evaluation is the 2-step service model, in which the heuristics and global optimization techniques described in Section 5.2 are used. In Section 6.4.1, a performance comparison of four versions of the DCM with varying decisive waiting times is presented, followed by a PSA of the system for the decision threshold θ and the diffusion rate ϕ.

6.4.1 Comparison of Customer Models

Recall, the four variations of the DCM evaluated in this section are introduced and explained in Section 6.2 and the colors associated with each version are adopted from Figure 6.3: red lines refer to the DCM with a $(t_{\text{w},1}, t_{\text{w},2})$-pair of (0 s, 900 s), purple with (150 s, 750 s), yellow with (300 s, 600 s), and blue with (450 s, 450 s).

The CCM and its representation in the DCM outperform other versions of the DCM with respect to to service profitability as shown in Figure 6.15. On the left, the total profit generated

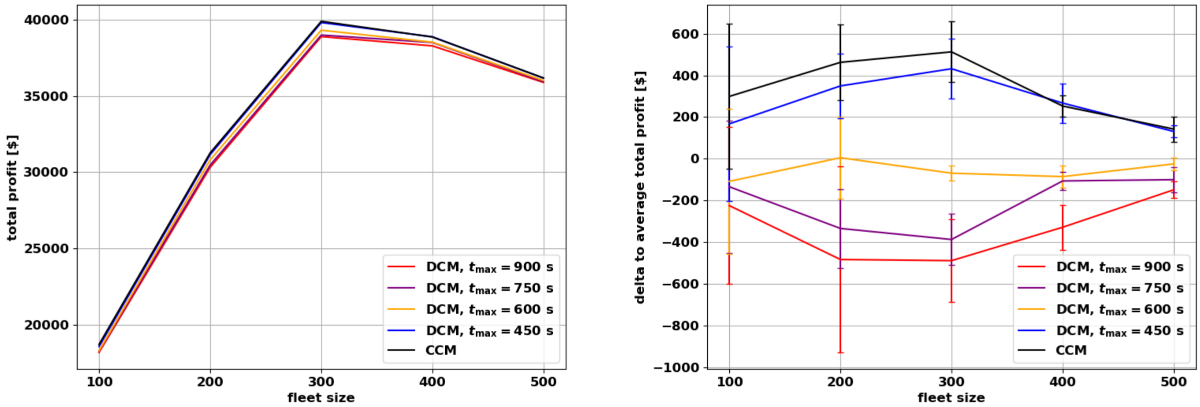


Figure 6.15: Profit in US-Dollars generated in simulations with various diffusion customer model parameters in the ride pooling use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

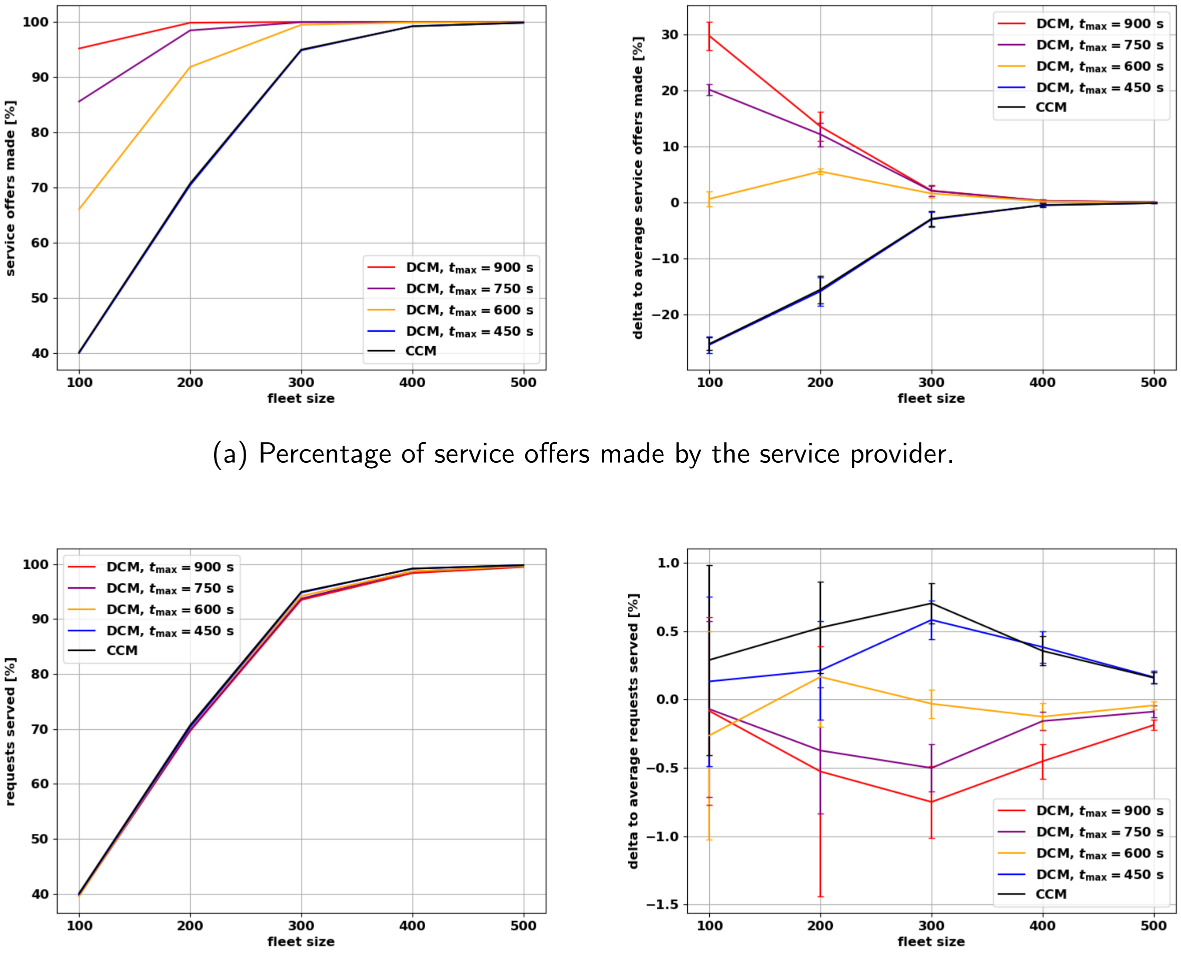
with all versions of the model are shown. The right-hand side presents the delta to average, illustrating the difference between the customer models. In scenarios with 300 vehicles and the maximum generated profit, the difference between the DCM with $t_{\mathrm{w},2} = 900\,\mathrm{s}$ and the CCM is largest and equates to 2.5 % of the total profit generated with the latter.

Figure 6.16 presents the percentages of requests served and initial service offers made by the service provider. The service operator aims to make as many offers as possible. With the examined versions of the DCM, higher values of $t_{\mathrm{w},2}$ allow the operator to initially accept more requests (see Figure 6.16a), at the cost of a decreased average offer quality. This results in more rejected offers overall, but also from users that might have received a better offer if the operator would not have offered the service to more customers, which causes a net decrease of requests served (Figure 6.16b).

This effect is strongest in scenarios with the best coverage of requests by service vehicles. In scenarios in which the demand is too high to be satisfied, vehicles are utilized close to their capacity anyway, so rejected offers are quickly replaced by another. If the fleet size is larger than needed to cover the demand, the average offer quality is higher, independent of the version of the DCM in use, which in turn leads to most offers being accepted by customers.

The differences in emptily driven mileages presented in Figure 6.17 reveal that the choice of the customer model only moderately influences this KPI. The empty mileage due to pickup trips shown in Figure 6.17a differs by around 1 % at maximum in scenarios with fleet sizes of up to 200 vehicles, but fluctuates within the standard deviations of the simulations of larger scenarios. The empty trips due to repositioning do not vary between the models significantly in any of the considered scenarios (Figure 6.17b).

This is plausible, because the DCM allows customers to reject offers after a certain period of time after the user request was accepted by the operator. In this period, a vehicle already started its trip towards the pickup location of the respective user, assuming it was idle at the time of the request. If the offer is rejected however, this trip terminates and produced empty mileage that is avoided in the CCM. This effect is stronger in versions of the DCM with larger



(a) Percentage of service offers made by the service provider.

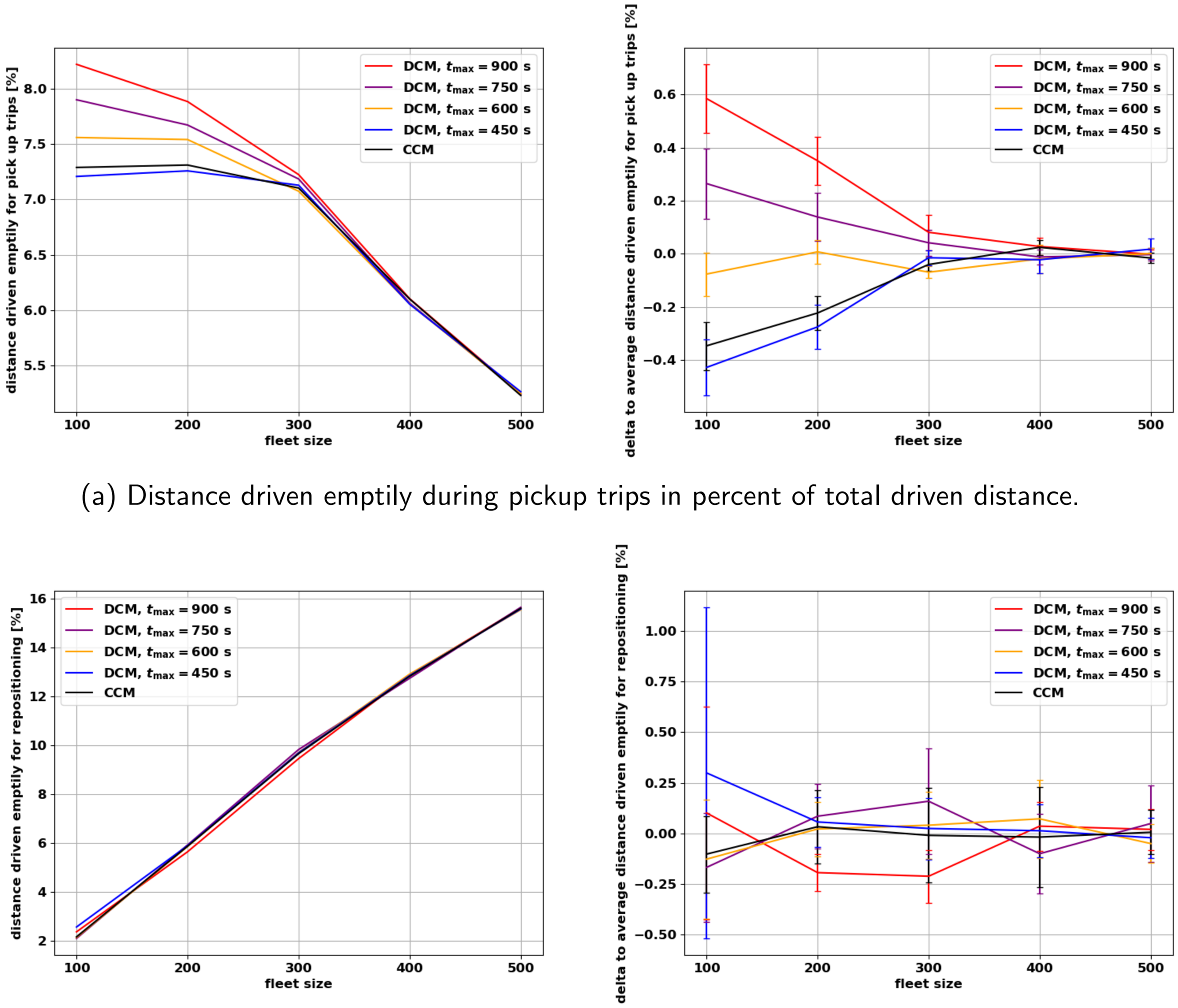
(b) Percentage of requests served by the service provider.

Figure 6.16: Percentage of requests served and service offers made in simulations with various diffusion customer model parameters in the ride pooling use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

differences between $t_{\mathrm{w},1}$ and $t_{\mathrm{w},2}$, because more offers are rejected for the reasons explained above.

Another KPI that is affected by the reduced number of served users is the average occupancy of service vehicles, shown in Figure 6.18. Especially in scenarios with smaller fleets, in which more rides are typically shared, the differences between the versions of the DCM and the CCM are apparent. The average number of passengers in simulations with the DCM is up to 0.06 lower than the average occupancy in the CCM, which equates to approximately 3.3 %. This gap closes in scenarios with more vehicles, in which less rides are shared overall because more service vehicles are available.

In Figure 6.19, the average user waiting time until pickup is presented for customers modeled with the CCM and the various versions of the DCM respectively. Especially in small-fleet



(a) Distance driven emptily during pickup trips in percent of total driven distance.

(b) Distance driven emptily during repositioning trips in percent of total driven distance.

Figure 6.17: Distances driven emptily in percent of total mileage in simulations with various diffusion customer model parameters in the ride pooling use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

scenarios, service users modeled with the DCM clearly have to wait shorter for their pickup on average. The gap is largest in scenarios with 100 vehicles (up to 26.2 s, or 9.5 %) and continuously shrinks with increasing fleet size, until it is virtually the same in scenarios with 500 vehicles.

The reason for the decreased waiting times when using the DCM in scenarios with small service fleets is the implied priority of customers that receive offers with short pickup waiting times in the DCM. Unlike the CCM, in which the offer quality does not change with the waiting time (unless it is longer than t_{max}), in the DCM the probability of customers accepting an offer grows with shorter waiting times, which results in an overweight of served users with shorter waiting times. This effect is amplified by the design of the 2-step service model that is used to evaluate the service model in this work. Because most users have to accept two offers made

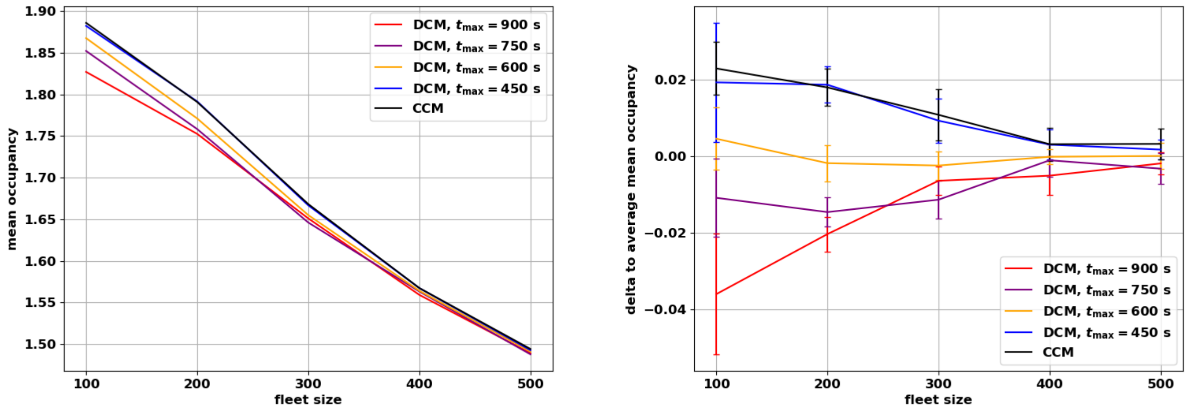


Figure 6.18: Occupancy in average passengers per vehicle in simulations with various diffusion customer model parameters in the ride pooling use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

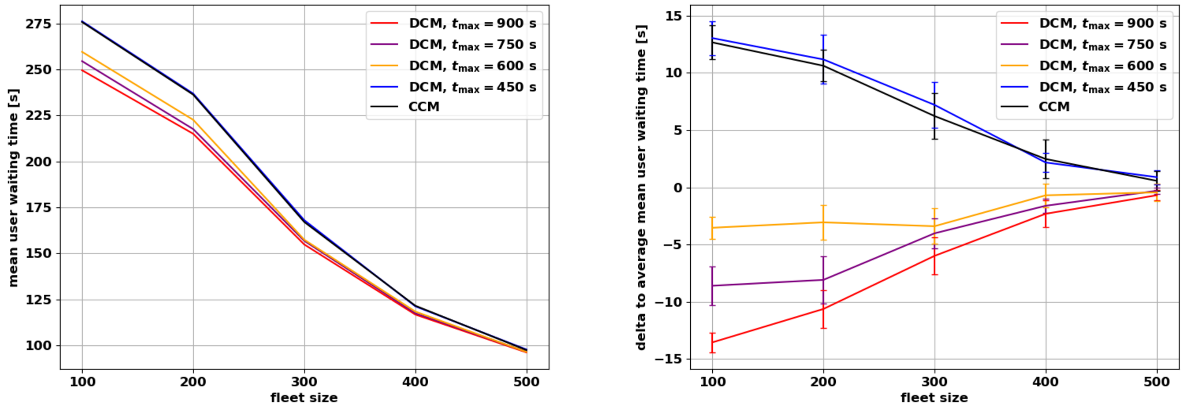


Figure 6.19: User waiting times in seconds in simulations with various diffusion customer model parameters in the ride pooling use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

by the service provider, the probability of customers accepting bad offers with long waiting times is further reduced.

6.4.2 Model Parameter Sensitivity Analyses

Like in Section 6.3.2 for the ride hailing use case, the upcoming section examines the impact of two model parameters of the DCM on the system, this time for the ride pooling use case. Again, the parameters are the decision threshold θ and the diffusion ϕ. The simulations are run with fleet sizes of 300 vehicles, the 2-step service model and over all seven simulation dates between November 12 and 18, 2018. The figures show the respective KPI in relation to one of the evaluated model parameters for each of the dates (light colors) as well as the average over all dates (black line). The left hand side of each figure depicts results associated with first offers, the right-hand side relates to second offers. The used DCM-version has decisive waiting times $t_{\mathrm{w},1} = 300\,\mathrm{s}$ and $t_{\mathrm{w},2} = 600\,\mathrm{s}$. The inherent parameters θ and ϕ range from 30 to 50 and 2.0 to 4.0 respectively.

Decision Threshold

The decision threshold θ determines at what decision states D_t customers accept or reject offers. If the decision-making process described in Section 6.1 results in $D_\mathrm{t} \geq +\theta$, the offer is accepted. In case $D_\mathrm{t} \leq -\theta$ the decision-making process results in a rejection. If the maximum decision time $t_{\mathrm{d,max}}$ is reached before either of these values is reached, the offer is accepted if $D_\mathrm{t} \geq 0$ and rejected otherwise. If the assigned service vehicle reaches the pickup location before a decision is made, the respective user accepts the offer and boards the vehicle immediately.

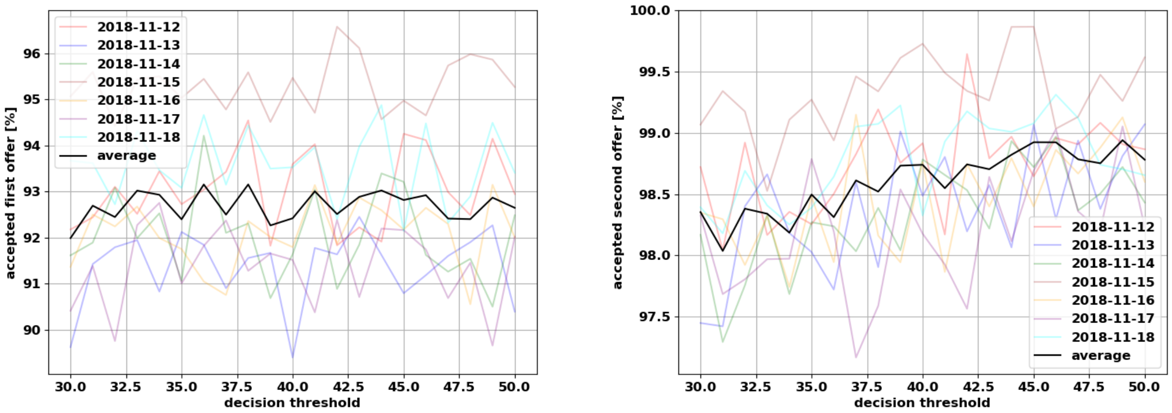
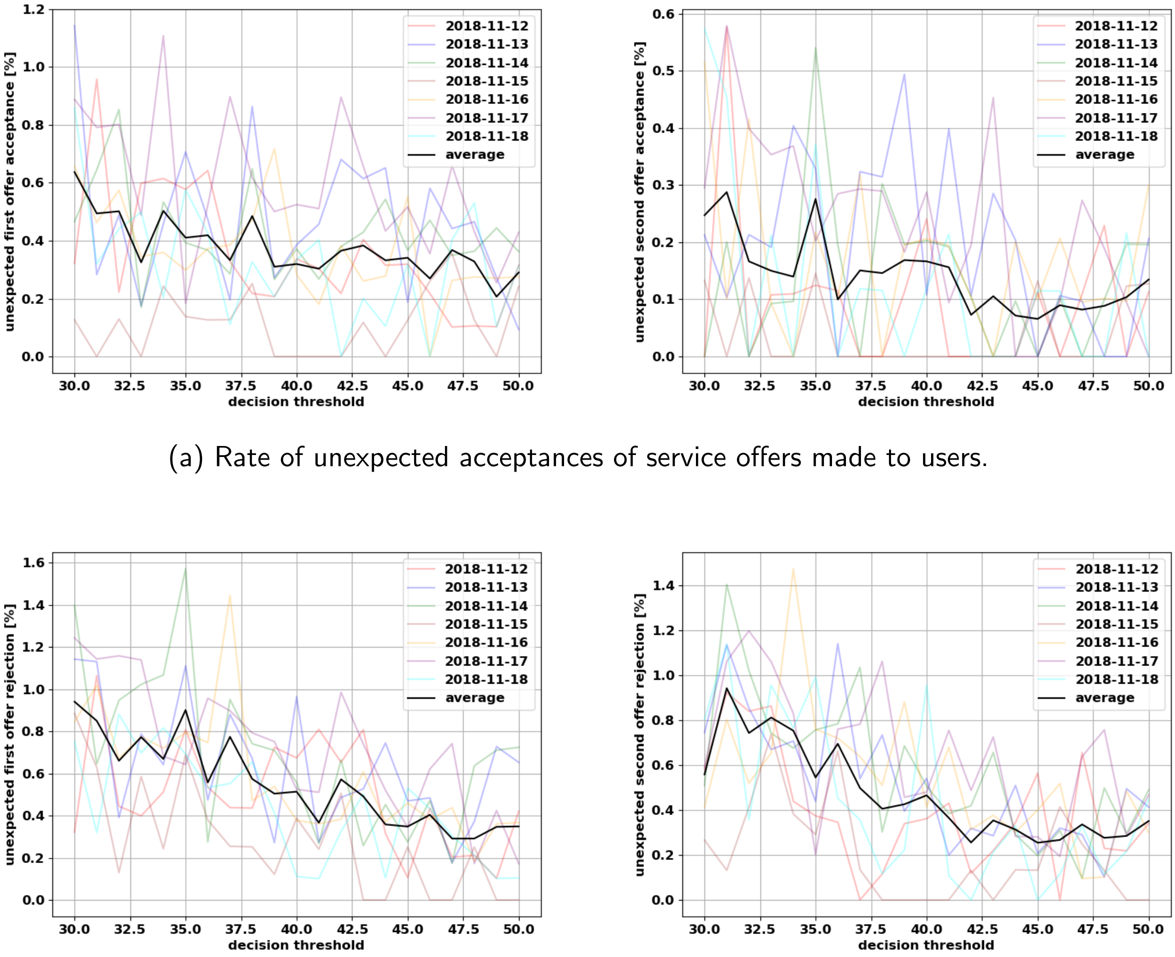


Figure 6.20: Acceptance rates of service offers made to users in percent of all such offers made in simulations with decision thresholds between 30 and 50 in the ride pooling use case. Values for all simulation dates and the average values. Left: related to first offers. Right: related to second offers.

In the DCM, service users with varying decision thresholds differ in their reaction to service offers with similar qualities. Customers with high θ-values tend to make more thoughtful decisions, which take a little longer and are more often in accordance with the expected decision based on the offer quality. On the other hand, low values of θ indicate a more spontaneous decision-making habit, quicker and less reliable in regard of the correlation between offer quality and decision making.

The acceptance rates of first and second offers, depicted in Figure 6.20, are consequences of this model design. The average share of accepted first offers fluctuates between 92.0 % and 93.2 % with no apparent correlation to the decision threshold.



(a) Rate of unexpected acceptances of service offers made to users.

(b) Rate of unexpected rejections of service offers made to users.

Figure 6.21: Rate of unexpected decisions made by users in percent of all such offers made in simulations with decision thresholds between 30 and 50 in the ride pooling use case. Values for all simulation dates and the average values. Left: related to first offers. Right: related to second offers.

The average offer quality is independent of the model parameters of the customers and is measured to be 0.88 for first offers and 0.94. Nevertheless, second offers are more often accepted by customers with high values of θ, as shown in the right part of the figure. While users with decision thresholds of $\theta = 31$ accept approximately 98.0 % of all second offers on average, the maximum share of acceptances is reached by users with $\theta \geq 44$, with values upside of 98.8 %. For second offers, the correlation of acceptance rates and decision threshold is clear, even though the rates are very close to 100 % and the scale of variation is below 1 %.

This trend can be explained by the rates of unexpected acceptances and rejections, presented in Figure 6.21. Because of the very high average offer qualities provided by ride pooling services due to shorter average user waiting times, the total percentages of unexpected decisions are low. In fact, the average rate of unexpected acceptances of first offers $r^1_{\text{a,u}}$ is highest for users with $\theta = 30$ at 0.64 % and decreases to 0.21 % for users with $\theta = 49$. The scale is even smaller for $r^2_{\text{a,u}}$, where the difference between the highest value (0.29 % for users with $\theta = 31$) and the lowest (0.07 % for users with $\theta = 45$) is only 0.22 %.

The average unexpected rejection rates in Figure 6.21b are slightly higher because of the fact that the average offer qualities are close to the maximum of one (for first and second offers) and there are more offers with positive implied qualities that can end up being unexpectedly rejected. Note that unlike the rates of unexpected acceptances, the graphs of $r^1_{\text{r,u}}$ and $r^2_{\text{r,u}}$ for varying values of θ are very similar in scale and slope. Both range from 0.94 % for small values of θ to minima of 0.29 % and 0.25 % respectively.

The discrepancy in scale and scope between unexpected acceptances and rejections of second offers means they do not cancel each other out as in the case of first offers. Hence, the overall acceptance rate of second offers is slightly higher for users with higher decision thresholds.

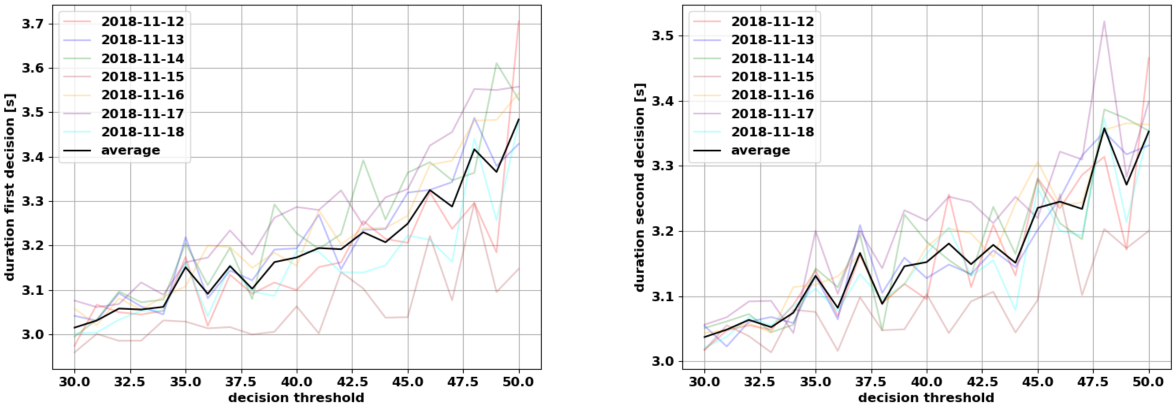


Figure 6.22: Decision duration in seconds in simulations with decision thresholds between 30 and 50 in the ride pooling use case. Values for all simulation dates and the average values. Left: related to first offers. Right: related to second offers.

Figure 6.25 presents the average durations of decision-making processes. The differences between first and second offers are narrow. In both cases, the average time it takes users to decide about an offer is longer if their respective θ is high, ranging from 3.0 s at $\theta = 30$ to 3.5 s and 3.4 s at $\theta = 50$ for first and second offers, respectively.

Again, these findings match the anticipated model behavior. Higher decision thresholds correspond with prolonged decisions, because the drifting random walk used in the simulation of the decision-making process needs more decision steps to reach the critical values of $\pm\theta$. In the ride pooling use case, this correlation is weaker than in the ride hailing use case, since the average offer qualities are higher, which results in larger drifts and ultimately in quicker decision-making processes.

Diffusion Rate

In the DCM, the diffusion rate ϕ determines the standard deviation of the random walk around the drift implied by the offer quality. High values of ϕ are associated with customers whose decision making is rather erratic, because the amplitude of random fluctuations of the decision state D_t is higher, which is expected to not only lead to more unpredictable decisions but also shorter average decision durations.

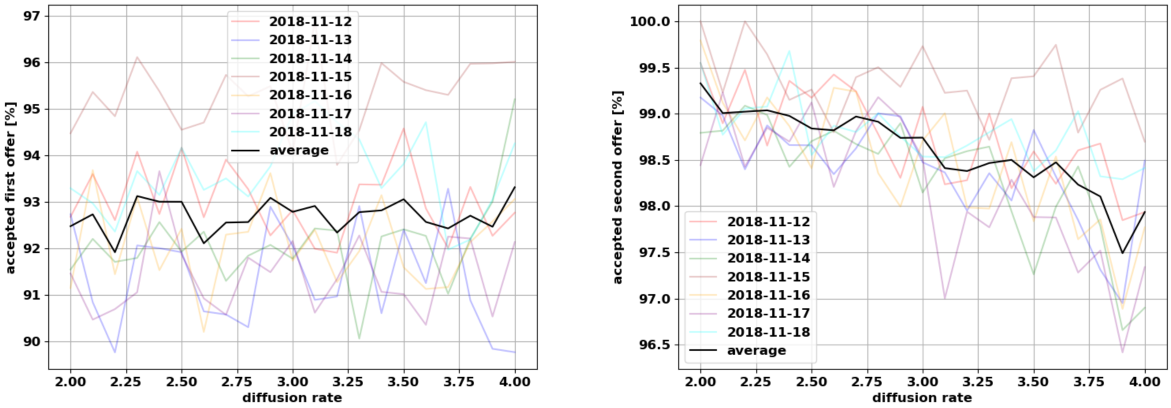
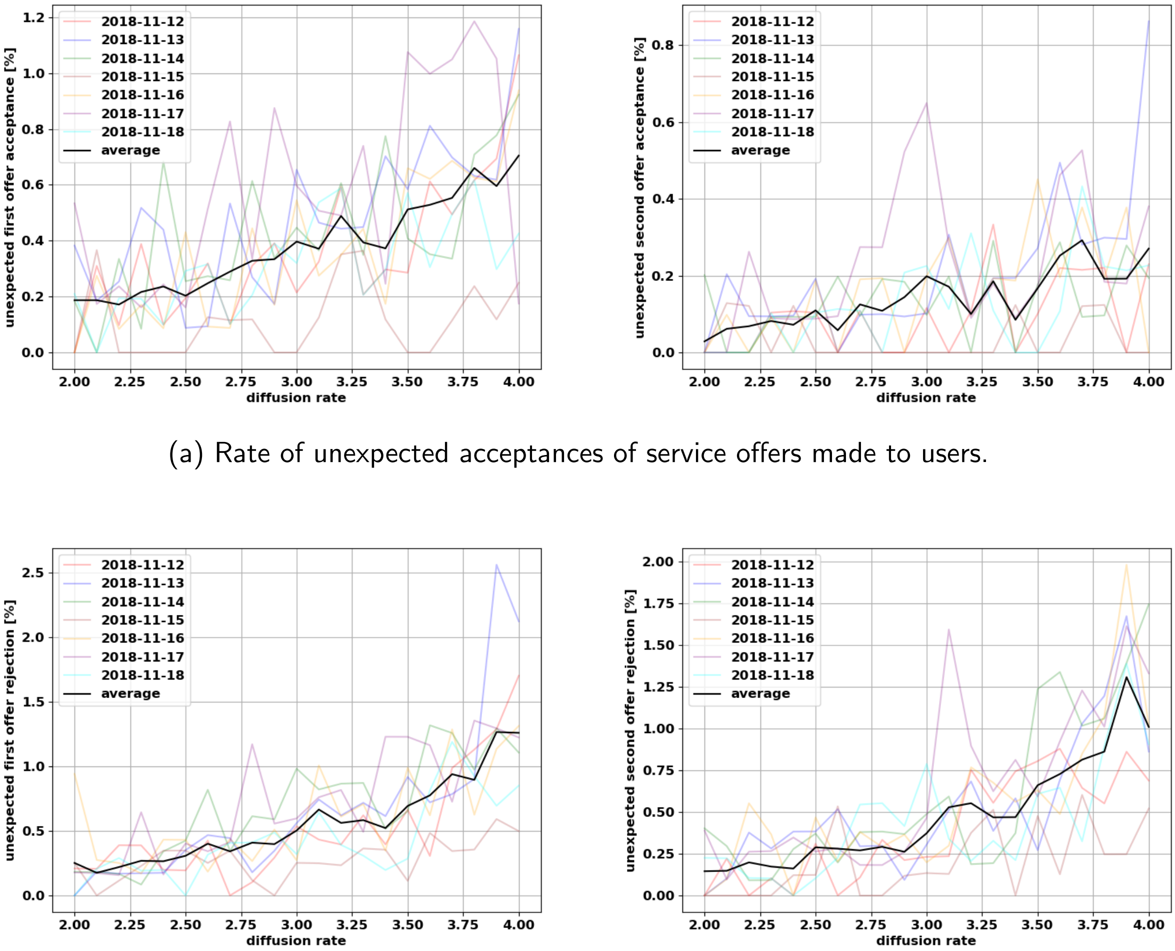


Figure 6.23: Acceptance rates of service offers made to users in percent of all such offers made in simulations with diffusion rates between 2.0 and 4.0 in the ride pooling use case. Values for all simulation dates and the average values. Left: related to first offers. Right: related to second offers.

In Figure 6.23, the acceptance rates of first and second offers in relation to the diffusion rates of customers are presented. While the stochastic fluctuations of the acceptance rate of first offers cover any dependency on ϕ, second offers are clearly accepted more often by customers with lower values of ϕ. The maximum of 99.3 % is reached by users with $\phi = 2.0$, while those with $\phi = 3.9$ have the lowest share of accepted second offers with 97.5 %.

Recall that the offer quality does not depend on the diffusion rate or any other characteristic of a service user, but only on the pickup waiting time that is offered by the provider. The quality of first offers is worse because the assignment methods used by the operator aim to make as many offers as possible, even at the cost of a number of offers being perceived as bad (those with a negative offer quality). The DCM is designed to disadvantage such offers and make it less probable for customers to accept them. Hence, the majority of rejections of first offers are connected to such bad offers.

Thus, the total number of good first offers that are rejected is very low, the total amount of rejected bad first offers relatively high. This not only results in a subset of requests that is presented with a second offer, that can mostly be served quick enough to be perceived as good. The imbalance of offers with positive and negative qualities is even amplified by this effective filter process during the first decisions made by the customers.



(a) Rate of unexpected acceptances of service offers made to users.

(b) Rate of unexpected rejections of service offers made to users.

Figure 6.24: Rate of unexpected decisions made by users in percent of all such offers made in simulations with diffusion rates between 2.0 and 4.0 in the ride pooling use case. Values for all simulation dates and the average values. Left: related to first offers. Right: related to second offers.

This effect can be observed in the rates of unexpected acceptances (Figure 6.24a) and rejections (Figure 6.24b). The percentage of acceptances of first offers that are perceived as bad, is almost as high as the rate of unexpected rejections for all diffusion rates. In the former case the share increases from less than 0.20 % for users with $\phi \leq 2.2$ to 0.70 % when $\phi = 4.0$. In the latter, the range is 0.18 % to 1.27 %. This implies that there is a relatively equal share of first offers with qualities between 0 and 1 and such with qualities between −1 and 0.

However, when considering the right-hand sides of both figures, there is a clear divergence between the shown graphs. While the rate of unexpected acceptances of second offers drops significantly compared to first offers (range between 0.03 % and 0.29 %), the rate of unexpected rejections is virtually the same within stochastic fluctuations (0.25 % to 1.31 %) between both offers. The gap between unexpected second acceptances and rejections grows with increasing diffusion rates, which ultimately results in the observed negative correlation with the overall acceptance rate of second offers and ϕ in Figure 6.23.

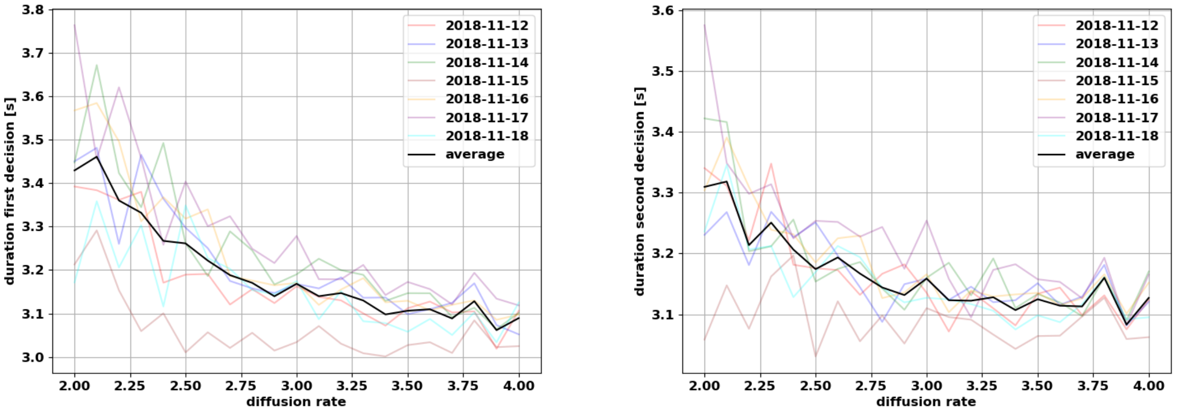


Figure 6.25: Decision duration in seconds in simulations with diffusion rates between 2.0 and 4.0 in the ride pooling use case. Values for all simulation dates and the average values. Left: related to first offers. Right: related to second offers.

The relations between average decision durations and diffusion rates for first and second offers $t_{\mathrm{d}}^{1,2}$ are shown in Figure 6.25. As anticipated for the DCM, customers with higher values of ϕ tend to make quicker decisions than users with lower diffusion rates. The differences between the longest and shortest average decision duration are approximately 0.4 s and 0.3 s for first and second offers respectively.

The small total values of $t_{\mathrm{d}}^{1,2}$ as well as the slim differences in relation to ϕ are a consequence of the high average offer qualities involved. By design, those lead to strong drifts of the random walk used in the DCM, thereby to ascents of the decision states of many of the customers towards the decision threshold and a quick acceptances of offers.

Chapter 7

Discussion and Outlook

The concluding chapter of this work aims to recall the research questions formulated in Chapter 1, give concise answers to each of them, and put them into perspective. A summary of the most significant results and findings is presented alongside an outlook toward future research that could build upon these outcomes.

7.1 Discussion of Results

In this section, the research questions (RQs) of this work are answered concisely and with context, based on the obtained results. The findings are summarized, discussed and interpreted, including an assessment of strengths and weaknesses of the service models and customer models considered in this work.

Recall that this work focuses on two main research questions. The first one (RQ1) is stated as follows:

> *How to design an ODM service model in order to bring together optimal request assignments and quick responses to users, considering both ride hailing and ride pooling use cases?*

To find an answer to RQ1, three different concepts for ODM service models are compared for the use cases of ride hailing and ride pooling:

- Service Model 1 makes use only of heuristic methods and **no global optimization** for assignment of vehicles to service requests. This approach focuses on fast response times rather than optimal assignments.
- Service Model 2 is a **2-step service** model that uses a heuristic algorithm for initial assignments and global optimization methods for periodical improvements of those assignments. It aims to benefit of the advantages of quick responses based on heuristics and the potential of global optimization to improve the system performance.
- Service Model 3 uses **only global optimization** techniques for vehicle-user assignments. Its focus lays on optimality at the cost of longer response and use waiting times.

These service model concepts are presented in detail in Section 3.1. The management of the ODM fleet, including the objective function for the underlying assignment problem, is explained in Section 3.2. The comparison of the service models is conducted by means of an agent-based simulation. A conventional customer model is used (see Section 3.3) in a case study based on real-world data, as explained in Section 3.4.

In the ride hailing use case – covered in Chapter 4 – the heuristic method used in Service Models 1 and 2 is based on the nearest-neighbor policy, while the global optimization in Service Models 2 and 3 is executed by the commercial solution-providing software CPLEX. The relative differences between system performances vary in terms of many KPIs in simulations with all three service models, depending on the fleet sizes considered. Scenarios in which the fleet size is smaller than the optimal fleet size with respect to profitability are considered scenarios with small fleets. Independent of the service model, the optimal fleet size for the ride hailing use case in the case study conducted in this work is found to be 400 vehicles.

In scenarios with small fleets, results indicate that Service Model 3 performs slightly better than the other service models in terms of daily profit, requests served, distance driven, and average vehicle occupancy, all KPIs considered to be service quality parameters for the service provider and the city. Service Model 2 performs best with respect to customer-related KPIs: the request serving rate is not as high as in simulations with Service Model 3 but slightly higher than with Service Model 1, the average pickup waiting time for users is the shortest amongst all three service models, and the average pickup-in-time-window rate in the small-fleet scenarios is higher than 99 %. Out of the three service models considered, Service Model 1 performs worst overall in ride hailing scenarios with small fleet sizes. The profitability for the service provider is the lowest due to the most emptily driven mileage, and the fewest requests served.

In scenarios with fleet sizes that are at least as large as the optimal fleet size, the relative system performances differ from those in scenarios with smaller fleets. The increasing difference in pickup waiting times between Service Model 3 and the other two service models means that from a customer's perspective, the service quality provided with Service Models 1 and 2 is clearly higher. Service Model 3 performs slightly better than Service Model 2 in terms of traffic-related KPIs, but the essentially equal numbers of requests served and daily profit between all three service models in these scenarios lead to the conclusion that overall, Service Model 2 is the best-performing service model in scenarios with large fleets. It should be noted that the system performance with Service Model 1 is very close to the other two, particularly in large-fleet scenarios and that the computational times for these simulations were found to be significantly shorter than for Service Models 2 and 3.

The comparisons between the service models for ride hailing in scenarios with small and large fleet sizes with respect to system performances from the perspective of customers, the city, and the service provider, respectively, are presented in Figure 7.1. A green "+" indicates a good system performance relative to the other service models, an "o" stands for middling system performance, and a red "-" symbolizes a bad system performance. The overall system performance is summarized as the total of the three stakeholder-related symbols. Here, one bad aspect of the system performance ("-") negates one positive ("+"). Note that this is a generalization of the more detailed results from Section 4.2.1, meant to provide an overview of how the evaluated service models performed from the perspectives of the main stakeholders of ODM services.

The parameter sensitivity analysis conducted for Service Model 2 in scenarios with 300 vehicles shows that the model, in general, is resilient to changes in model parameters, such

	Service Model 1	Service Model 2	Service Model 3
Customer	○	⊕	○
City	○	○	⊕
Service Provider	⊖	○	⊕
Total	⊖	⊕	⊕ ⊕

(a) Comparison of system performances in scenarios with small fleets.

	Service Model 1	Service Model 2	Service Model 3
Customer	⊕	⊕	⊖
City	⊖	○	⊕
Service Provider	○	○	○
Total	○	⊕	○

(b) Comparison of system performances in scenarios with large fleets.

Figure 7.1: Summary of system performances in ride hailing simulations with three different service models with respect to three main stakeholders.

as the optimization period, the weight of driven distance in the objective function relative to pickup waiting times, the length of the assignment lock time, and the length of pickup time windows. Effects on KPIs are found to be rarely larger than 1 % of the respective total value of each KPI when using the standard values of each parameter. This means that ride hailing service providers can fine-tune model parameters that are, e.g., critical for the service quality assessment by customers without significant negative effects on service KPIs. For example, longer assignment lock times mean that service users can be informed earlier about their exact pickup time, which can be considered an improvement in service quality. The fact that, according to the conducted analysis of Service Model 2, small changes of these parameters do not negatively affect most of the evaluated KPIs, allows ride hailing service providers to use a very customer-friendly model parameter setup, which further increases the service quality perceived by service users, which is already a strength of this service model, as shown in Figure 7.1.

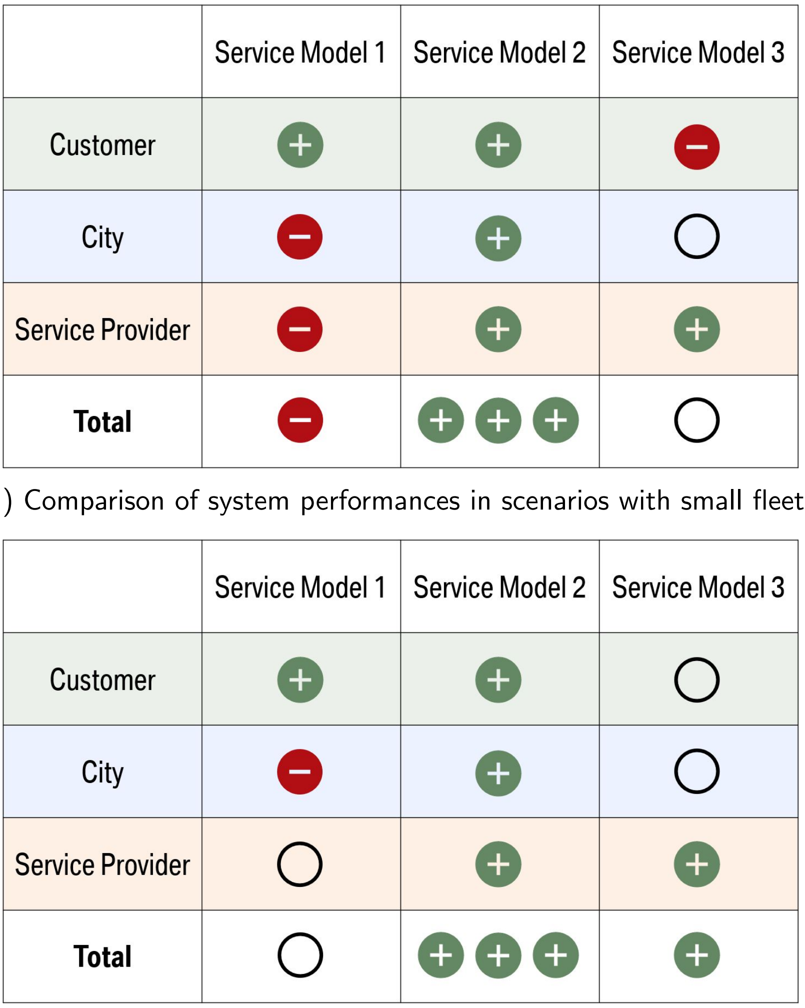
The same three service model concepts are evaluated in the ride pooling use case. However, the heuristic and global optimization methods used are considerably more complex due to the nature of the use case, which allows shared rides between customers. This translates to a significantly larger solution space in every instance of the assignment problem. The insertion heuristic used in Service Models 1 and 2, as well as the anytime optimal algorithm used in Service Models 2 and 3, are described in Section 5.2.

The optimal fleet size in the case study conducted for the ride pooling use case in this work is found to be 300 in all service models considered. Compared to the ride hailing use case, this means fewer vehicles are needed to serve the same number of requests, which is expected due to the possibility of sharing rides and thereby increasing the effectiveness of the service.

Comparing the system performances in scenarios with each of the service models, Service Model 2 clearly outperforms Service Models 1 and 3 with respect to most of the KPIs considered. It is on par with Service Model 3 in terms of profitability for the service provider and at least matches the performance of Service Model 1 in terms of customer-related KPIs, such as average detour and pickup waiting times. It outperforms both when it comes to positive traffic impact, which is most relevant for cities the ODM service is operated in. Service Model 3 performs worst with regard to KPIs that measure the service quality from a customer's perspective, primarily because of long detours and pickup waiting times, especially in scenarios with smaller fleets. Compared to Service Model 2, it generates similar amounts of daily profit for the service provider and serves more requests on average than any of the other service models. From the perspective of the city the service is run in, it cannot match the performance of Service Model 2 in KPIs like empty fleet mileage or average vehicle occupancy but clearly outperforms Service Model 1 in that regard. Service Model 1 competes for the best service quality from a customer's point of view, with waiting times comparable to Service Model 2, especially in large-fleet scenarios, and the 100 % rate of pickups in communicated time windows due to the service model design. It performs worse than both of the other service models regarding KPIs prioritized by cities and service providers, though, which outweighs the benefit of significantly shorter computation times.

Overviews of the system performances of all three service models in the ride pooling use case for scenarios with small and large fleets with respect to the main stakeholders are presented in Figure 7.2. The superiority of Service Model 2 relative to Service Models 1 and 3 is shown very clearly. Also note that this gap is more prominent in scenarios with small fleets and that Service Model 3 consistently outperforms Service Model 1 outside of KPIs that measure the service quality from a customer's perspective, independent of fleet sizes.

The parameter sensitivity analysis conducted for the ride pooling use case shows that the objective weight of driven distance α has an unmistakable impact on the system performance. Unlike ride hailing scenarios, the choice of this parameter affects all measured KPIs significantly, which gives service operators options to actively regulate the focus of the service. Higher values of α lead to higher average vehicle occupancies and more saved mileage relative to the case in which all trips are made with privately-owned vehicles. This also allows service providers to increase the profitability of the service, assuming the prices for customers remain constant. On the other hand, average detour and pickup waiting times are longer, and the reliability of pickups within communicated time windows drops sharply, all of which have a negative effect on the perceived service quality from a customer's point of view. In practice, service



	Service Model 1	Service Model 2	Service Model 3
Customer	⊕	⊕	⊖
City	⊖	⊕	○
Service Provider	⊖	⊕	⊕
Total	⊖	⊕⊕⊕	○

(a) Comparison of system performances in scenarios with small fleets.

	Service Model 1	Service Model 2	Service Model 3
Customer	⊕	⊕	○
City	⊖	⊕	○
Service Provider	○	⊕	⊕
Total	○	⊕⊕⊕	⊕

(b) Comparison of system performances in scenarios with large fleets.

Figure 7.2: Summary of system performances in ride pooling simulations with three different service models with respect to three main stakeholders.

operators could make use of this by adjusting α depending on the current demand or linking it to dynamic pricing strategies. The other evaluated model parameters do not have a similar impact on the system performance, which allows service operators to tune them to values that increase the perceived service quality, similar to the ride hailing use case.

Overall, Service Model 2 is shown to combine strong system performances with respect to KPIs relevant to all three main stakeholders of ODM services with quick response times to service users. This holds for ride hailing and ride pooling use cases, albeit much more so for the latter, in which it clearly outperforms the other service models considered in this work.

It should be noted that the system performance of ride pooling services are better in comparison to ride hailing services regarding many KPIs: more effective, smaller fleets can serve the same number of requests, driving shorter distances, which translates to higher profit potentials for service providers, and less traffic in the business area. The shorter average waiting times until pickup are negated by unavoidable detour times for customers of ride pooling services, though. Together with slightly less reliable pickup time windows, this might justify lower prices for ride pooling services compared to ride hailing offers, which in turn reduces the profitability for providers of such services.

Finding the delicate balance between maximizing the profit for service providers and the service quality perceived by customers is a difficult task itself. This leads to the second research question answered in this work. RQ2 is stated as follows:

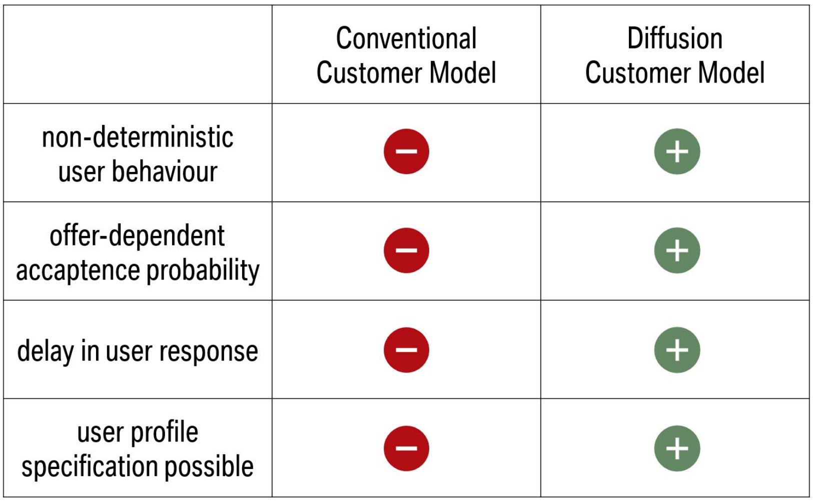
> *How to improve the comparability of system performances of ODM service models to real-world applications and how to measure the impact of customer models on the system performance?*

This work introduces a novel customer model, referred to as the "diffusion customer model" (DCM), based on a random walk process to simulate customer behavior. It is compared to a customer model used in this or similar forms in numerous ODM studies and publications. This "conventional customer model" (CCM) is used in the simulations to answer RQ1 to focus on differences between the evaluated service models rather than the effects of the customer model in use. The CCM is a very simplified model of service users: the decision if a service offer is accepted or not solely depends on the question if the pickup will happen within a set maximum waiting time. If so, the request is immediately accepted, independent of the exact waiting time and without any delay on the side of the customer. In this sense, the CCM is deterministic, and every customer is modeled identically. This allows the comparison of service models in more detail (because every observed effect on system performances can be directly linked to the respective service model) but is also an unrealistic representation of service users in simulations of ODM services.

The DCM, on the other hand, includes features that address these shortcomings of the CCM. As described in Section 6.1, the decision-making process relies on a drifting random walk, which means that this model is non-deterministic. The probability of accepting a given offer depends on the respective offer quality, and therefore the drift of the underlying random walk, and decreases gradually with increasing waiting times. The offer quality also has an impact on the time it takes the customer to make a decision: very good (and very bad) offers tend to be accepted (and rejected) relatively quickly. In contrast, middling offer qualities imply that customers might hesitate to make a decision. With its model parameters, the DCM allows the representation of certain types of customer behaviors, e.g., a higher decision threshold θ means that the random walk has to meet higher total values. Hence, such customers can be interpreted as less impulsive and more thoughtful in their decision making.

A comparative overview of features provided by the CCM and the DCM is presented in Figure 7.3. The DCM is clearly more suited to represent service customers more accurately in ODM service simulations. In order to understand the impact such a more realistic customer model has on the system performance of an ODM service model in the ride hailing and ride pooling use cases, respectively, this work evaluated the 2-step service model in simulations with multiple versions of the DCM in both use cases and compared it to the results found with the CCM. The model parameters and KPIs used for these comparisons are listed and explained in Section 6.2.

The system performance in simulations with the DCM in regard to KPIs related to the service provider and the city is worse compared to simulations with the CCM. With increasing



	Conventional Customer Model	Diffusion Customer Model
non-deterministic user behaviour	−	+
offer-dependent accaptence probability	−	+
delay in user response	−	+
user profile specification possible	−	+

Figure 7.3: Summary of model properties of the CCM and the DCM.

maximum pickup waiting times t_{max} between the evaluated versions of the DCM, the number of sent-out service offers rises while the share of served requests decreases. As described in Sections 6.3 and 6.4, this is because the average offer quality is lower in scenarios with higher t_{max} in combination with the double-filtering effect implied by the DCM in simulations with the 2-step service model. Together with the additional empty mileage due to trips made to pickup locations of customers that eventually reject service offers, this means that in simulations with the DCM, the evaluated service model produces more traffic in the business area while generating less profit for the service provider than in simulations with the CCM. However, the average pickup waiting time of requests that are served in these simulations is significantly shorter compared to the CCM, also a result of the effective filtering of offers that are perceived as bad throughout two separate decision-making processes in the 2-step service model.

The PSAs of one version of the DCM conducted in simulations of the 2-step service model in the ride hailing and ride pooling use case, respectively, validate the model and confirm that the model parameters have the expected impact on measurable KPIs like the acceptance rates of first and second offers and the duration of each of the decision-making processes. Higher values of decision thresholds θ and lower values of diffusion rates ϕ are associated with customers that are more thoughtful in their decision making and therefore do more often accept good offers and decline bad ones. They also take longer on average to decide whether to accept an offer or not. On the other hand, users with low values of θ and high values of ϕ tend to make more decisions that would not be expected based on the respective offer quality. Bad offers are more often accepted, and good offers are more often declined than in the case of the “thoughtful” customer type. This behavior, together with the shorter decision durations, is associated with rather spontaneous and unpredictable customers.

In conclusion, the DCM was found to be able to represent a more realistic customer behavior compared to the CCM, which helps to improve ODM service simulations in terms of comparability between simulated results and real-world system performances, which answers RQ2. The findings of this work suggest that the performance of service models with respect to most of the KPIs relevant to service providers and cities is overestimated in simulations with the CCM. The inclusion of a temporal aspect and an offer-quality depending degree of randomness in the decision-making process in the DCM is essential in simulations of service models that focus on operator-user interactions, like the 2-step service model evaluated in this work. It is shown that delays in user responses and unexpected user decisions have a significant impact on the system performance of such service models in the ride hailing and ride pooling use case. ODM service operators need to consider these effects in the design of their real-world services Otherwise, they could perform worse than expected according to simulations with customer models such as the CCM.

7.2 Outlook and Future Work

This work answers the research questions posed in Chapter 1 and contributes to the field of research about ODM service and customer models, as summarized in the previous section. This section acknowledges this work's limitations, and an outlook is provided on how they could be addressed in the future.

With regard to the simulation framework and problem sizes considered, this work focuses on the comparability between the evaluated service models rather than on scenarios that represent large-scale problem instances that can occur in real-world ODM services. More sophisticated frameworks that include larger fleets of ODM service vehicles, dynamic travel times, advanced repositioning and assignment algorithms or dynamic pricing models could and should be used to test the potential of the three service models that are evaluated in this work, especially the 2-step service model that is found to perform best amongst them. Of particular interest would be a test of the DCM in these simulation frameworks because the system performance of service models used in these simulations is expected to be significantly worse than results conducted with other customer models like the CCM. Candidates for such evaluations can be found in the literature presented in Chapter 2.

Another examination that is out of the scope of this work is the evaluation and validation of the DCM with data from real-world ODM service users. Assumptions about, e.g., average reaction times, maximum waiting times, and perceived offer qualities have been made on the basis of related literature and estimations. Future research should validate these assumptions by employing surveys and data analysis from ODM services. The insights could be used to calibrate the DCM parameters to further improve the model accuracy, for example, in terms of realistic decision-making durations.

The aspect of delayed customer responses has profound implications for the system performance of ODM service models. In this work, the service operator immediately sends idle vehicles to pickup locations of customers that receive their first offer, which results in additional mileage due to trips in the direction of pickup locations of customers that eventually reject an offer. Future studies could examine the system performance in scenarios where the vehicles only start their trips if the initial offer is accepted and if the potential decrease in empty mileage is worth the longer pickup waiting times implied by such a service policy. Another strategy could be to anticipate offer rejections with probabilities depending on the respective offer qualities and plan alternative routes. This creates immense potential for saved mileage and increased service efficiency, especially in the ride pooling use case.

As it is introduced in this work, the DCM also offers plenty of material to build upon with other follow-up studies. Its feature of individual user parameters that are associated with behavior archetypes like spontaneity, decisiveness, and thoughtfulness can be used to define customer groups that typically share these archetypes and specifically target them with certain types of offers or allow service providers to react accordingly. For example, a service operator that knows from preceding service requests about the tendency of a recurring customer to accept relatively good offers very reliably could be willing to send a vehicle to the pickup location immediately. In contrast, customers known to be more erratic in their decision-making process could be served after they eventually accept the offer. This proactive request management potentially increases the profitability of ODM services.

Another strategy that could be tested with the DCM is connected to deliberately over- or underestimating the pickup waiting time communicated to customers, especially in the initial offer in the 2-step service model presented in this work. Such a strategy could work if the perceived quality of the second offer directly depends on the offer quality of the initial offer, i.e., if the probability of accepting a second offer with a given quality is higher if the preceding first offer is bad compared to the acceptance probability of a second offer that follows a good first offer. Such a direct dependency between the qualities of offers is not implemented in the DCM evaluated in this work but could be of interest for ODM service providers because the positive psychological effect of overachieving in terms of service quality could outweigh the risk of additional offer rejections due to pickup waiting times of first offers that are deliberately inflated by service providers, as described in [Yu and Hyland, 2020].

In addition to more detailed studies about the DCM, especially its interactions with the 2-step service model, there are aspects of this service model that can be examined and improved independently of the customer model. In-depth analyses of system parameters like the vehicle capacity in the ride pooling use case could provide insight into what kind of service vehicles should be used to maximize efficiency or if service providers should instead offer a heterogeneous fleet of different vehicle variants. Alternative optimization methods should be explored, heuristics as well as global optimization algorithms, in the ride hailing and the ride pooling use case. This work does not consider the impact of computation times of optimization processes on the system performance, which leaves the opportunity for future research to compare various optimization techniques, particularly regarding their respective computation times and how more time-consuming methods affect the simulation framework and service KPIs.

Future work could also focus on a higher granularity of the evaluation of service models. In this work, KPIs are measured and compared in terms of the average over seven full days. This facilitates general observations and provides robust results but neglects the effects of different service and customer models on smaller time scales. Over the course of 24 hours, not only do traffic conditions change, which could be represented by dynamic travel times in the simulation framework but also demand varies drastically. This leads to periods with a lot of idle vehicles that are mainly used to relocate service vehicles in the business area and other parts of the day, during which many requests cannot be served because the fleet capacity is not high enough. Studying the fleet utilization, average vehicle occupancy, and request acceptance rate during the day would allow operators to adjust the service accordingly, including a more precise estimation of optimal fleet sizes, the potential for dynamic pricing patterns, and indications when to offer incentives for ride sharing during periods of high demand.

Acknowledgments

At the end, I would like to thank a number of people who made this work possible. First, a big thank you to Prof. Dr. Klaus Bogenberger, who not only contributed his ideas and pieces of advice since the beginning of my time as a Ph.D. candidate back in December 2017. Klaus, you are also a great mentor, who always has an open door for everyone in your Chair of Traffic Engineering and Control at the Technical University Munich, and a very open-minded and friendly nature, who supports his staff and students wherever possible.

As the second supervisor of my dissertation, great gratitude goes to Prof. Dr. Michael Hyland from the University of California. Thank you Mike, for the suggestion to lay focus on the diffusion customer model in this dissertation, for every discussion we had about it in the early hours of your days, and the valuable input you gave during these sessions. I also want to thank Jiangbo 'Gabe' Yu, who joint these calls as Mike's co-author of the publication about the diffusion customer model that this dissertation's version is based on. Gabe contributed important insights into the model that helped me to better understand and describe it.

BMW gave me the opportunity to write my dissertation as part of the BMW ProMotion Programme. I am very grateful to everyone in the departments of "Fleet Intelligence" and "Business Line My Journey" who welcomed me as part of the team from day one and helped me with their experience in the field of on-demand mobility services. Special thanks to Dr. Ulrich Fastenrath, who interviewed and convinced me to begin this whole journey. To my supervisors Irina Benkert and Dr. Heidrun Belzner, who both helped me with their great knowledge in technical and conceptual questions, provided guidance to navigate my way in a company as big as BMW, and supported me wherever they could. And, of course, to many other colleagues I had the joy to work with during the years. The time flew by and I enjoyed to be part of your team, learned a lot, and came to know numerous people I consider friends.

I would also like to thank my fellow Ph.D. candidates and colleagues at the Chair of Traffic Engineering and Control at Technical Universtity Munich. First and foremost Dr. Aledia Bilali, Florian Dandl, Roman Engelhardt and Arslan Ali Syed, who not only reviewed this work and gave valuable feedback to help me improve its quality. In countless meetings throughout the years, they also provided their expertise in questions about ride hailing and ride pooling service models, simulation frameworks, customer behavior and optimization algorithms, which was invaluable for this work to be realized. Thanks to all of you, as well as everyone else in the chair. I gained a lot from the exchanges of ideas in regular seminars and informal conversations and am very thankful for the time I spent with all of you.

Finally, I want to thank the people who helped me most, not only, but in particular during the time of this work. Thank you to Kerstin Erdmann, the most loving mother on Earth, for always supporting and believing in me. And thanks to Elisabeth Schuster, for being at my side, listening to my complaints, sharing my struggles and helping me to overcome them.

List of Figures

List of Tables

List of Publications

The following list contains all publications of the author relevant for this work in chronological order.

- Marvin Erdmann, Florian Dandl, Klaus Bogenberger (2019). “Dynamic Car-Passenger Matching based on Tabu Search using Global Optimization with Time Windows”. In: *2019 8th International Conference on Modeling Simulation and Applied Optimization (ICM-SAO)*, IEEE. DOI:10.1109/icmsao.2019.8880293.
- Marvin Erdmann, Florian Dandl, Bernd Kaltenhäuser, Klaus Bogenberger (2020). “Dynamic Car-Passenger Matching of Online and Reservation Requests”. In: *99th Annual Meeting of the Transportation Research Board (TRB 2020)*.
- Marvin Erdmann, 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*. DOI:10.1016/j.trc.2021.103104.

Bibliography

Accenture (2020). *Unlock the Value of Mobility Services*. url: https://www.accenture.com/_acnmedia/PDF-134/Accenture-Unlock-Value-Mobility-Services.pdf. (accessed: February 25, 2022).

Alonso-Mora, Javier; Samitha Samaranayake; Alex Wallar; Emilio Frazzoli; Daniela Rus (2017). "On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment". In: 114.3, pp. 462–467. doi: 10.1073/pnas.1611675114.

Alonso-Mora, Javier; Alex Wallar; Daniela Rus (2017). "Predictive routing for autonomous mobility-on-demand systems with ride-sharing". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. doi: 10.1109/iros.2017.8206203. url: https://doi.org/10.1109%2Firos.2017.8206203.

Arthur D. Little GmbH (2020). *Rethinking on-demand mobility - turning roadblocks into opportunities*. url: https://www.adlittle.de/en/insights/report/rethinking-demand-mobility. (accessed: February 25, 2022).

Attanasio, Andrea; Jean-François Cordeau; Gianpaolo Ghiani; Gilbert Laporte (2004). "Parallel Tabu search heuristics for the dynamic multi-vehicle dial-a-ride problem". In: *Parallel Computing* 30.3, pp. 377–387. doi: 10.1016/j.parco.2003.12.001. url: https://doi.org/10.1016%2Fj.parco.2003.12.001.

Bai, Ruibin; Edmund K. Burke; Michel Gendreau; Graham Kendall; Barry McCollum (2007). "Memory Length in Hyper-heuristics: An Empirical Study". In: *2007 IEEE Symposium on Computational Intelligence in Scheduling*. IEEE. doi: 10.1109/scis.2007.367686. url: https://doi.org/10.1109%2Fscis.2007.367686.

Bai, Ruibin; Xinan Chen; Zhi-Long Chen; Tianxiang Cui; Shuhui Gong; Wentao He; Xiaoping Jiang; Huan Jin; Jiahuan Jin; Graham Kendall; Jiawei Li; Zheng Lu; Jianfeng Ren; Paul Weng; Ning Xue; Huayan Zhang (2021). "Analytics and Machine Learning in Vehicle Routing Research". In: *CoRR* abs/2102.10012. arXiv: 2102.10012. url: https://arxiv.org/abs/2102.10012.

Baldacci, Roberto; Aristide Mingozzi; Roberto Roberti (2011). "New Route Relaxation and Pricing Strategies for the Vehicle Routing Problem". In: *Operations Research* 59.5, pp. 1269–1283. doi: 10.1287/opre.1110.0975. url: https://doi.org/10.1287%2Fopre.1110.0975.

Balinski, M. L.; R. E. Quandt (1964). "On an Integer Program for a Delivery Problem". In: *Operations Research* 12.2, pp. 300–304. doi: 10.1287/opre.12.2.300. url: https://doi.org/10.1287%2Fopre.12.2.300.

Battarra, Maria; Jean-François Cordeau; Manuel Iori (2014). "Chapter 6: Pickup-and-Delivery Problems for Goods Transportation". In: *Vehicle Routing*. Society for Industrial and Applied Mathematics, pp. 161–191. doi: 10.1137/1.9781611973594.ch6. url: https://doi.org/10.1137%2F1.9781611973594.ch6.

Baugh, John W.; Gopala Krishna Reddy Kakivaya; John R. Stone (1998). "Intractability of the dial-a-ride problem and a multiobjective solution using simulated annealing". In: *Engineering Optimization* 30.2, pp. 91–123. doi: 10.1080/03052159808941240. url: https://doi.org/10.1080%2F03052159808941240.

Bektaş, Tolga; Panagiotis P. Repoussis; Christos D. Tarantilis (2014). "Chapter 11: Dynamic Vehicle Routing Problems". In: *Vehicle Routing*. Society for Industrial and Applied Mathematics, pp. 299–347. doi: 10.1137/1.9781611973594.ch11. url: https://doi.org/10.1137%2F1.9781611973594.ch11.

Bent, Russell; Pascal Van Hentenryck (2004). "A Two-Stage Hybrid Local Search for the Vehicle Routing Problem with Time Windows". In: *Transportation Science* 38.4, pp. 515–530. doi: 10.1287/trsc.1030.0049.

Berbeglia, Gerardo; Jean-François Cordeau; Irina Gribkovskaia; Gilbert Laporte (2007). "Rejoinder on: Static pickup and delivery problems: a classification scheme and survey". In: *TOP* 15.1, pp. 45–47. doi: 10.1007/s11750-007-0015-2. url: https://doi.org/10.1007%2Fs11750-007-0015-2.

Berbeglia, Gerardo; Jean-François Cordeau; Gilbert Laporte (2010). "Dynamic pickup and delivery problems". In: *European Journal of Operational Research* 202.1, pp. 8–15. doi: 10.1016/j.ejor.2009.04.024. url: https://doi.org/10.1016%2Fj.ejor.2009.04.024.

Berbeglia, Gerardo; Jean-François Cordeau; Gilbert Laporte (2012). "A Hybrid Tabu Search and Constraint Programming Algorithm for the Dynamic Dial-a-Ride Problem". In: *INFORMS Journal on Computing* 24.3, pp. 343–355. doi: 10.1287/ijoc.1110.0454. url: https://doi.org/10.1287%2Fijoc.1110.0454.

Bertsimas, Dimitris J. (1992). "A Vehicle Routing Problem with Stochastic Demand". In: *Operations Research* 40.3, pp. 574–585. doi: 10.1287/opre.40.3.574. url: https://doi.org/10.1287%2Fopre.40.3.574.

Bertsimas, Dimitris J.; Garrett van Ryzin (1991). "A Stochastic and Dynamic Vehicle Routing Problem in the Euclidean Plane". In: *Operations Research* 39.4, pp. 601–615. doi: 10.1287/opre.39.4.601. url: https://doi.org/10.1287%2Fopre.39.4.601.

Bianchessi, Nicola; Giovanni Righini (2007). "Heuristic algorithms for the vehicle routing problem with simultaneous pick-up and delivery". In: *Computers & Operations Research* 34.2, pp. 578–594. doi: 10.1016/j.cor.2005.03.014. url: https://doi.org/10.1016%2Fj.cor.2005.03.014.

Bilali, Aledia; Florian Dandl; Ulrich Fastenrath; Klaus Bogenberger (2019). "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. doi: 10.1109/itsc.2019.8917280. url: https://doi.org/10.1109%2Fitsc.2019.8917280.

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. doi: 10.1177/0361198120917666. url: https://doi.org/10.1177%2F0361198120917666.

BODIN, Lawrence; Bruce GOLDEN (1981). "Classification in vehicle routing and scheduling". In: *Networks* 11.2, pp. 97–108. DOI: 10.1002/net.3230110204. URL: https://doi.org/10.1002%2Fnet.3230110204.

BRANDÃO, José (2011). "A tabu search algorithm for the heterogeneous fixed fleet vehicle routing problem". In: *Computers & Operations Research* 38.1, pp. 140–151. ISSN: 0305-0548. DOI: https://doi.org/10.1016/j.cor.2010.04.008.

BRÄYSY, Olli; Michel GENDREAU (2002). "Tabu Search heuristics for the Vehicle Routing Problem with Time Windows". In: *Top* 10.2, pp. 211–237. DOI: 10.1007/bf02579017. URL: https://doi.org/10.1007%2Fbf02579017.

BUSEMEYER, Jerome R.; Adele DIEDERICH (2002). "Survey of decision field theory". In: *Mathematical Social Sciences* 43.3, pp. 345–370. DOI: 10.1016/s0165-4896(02)00016-1. URL: https://doi.org/10.1016%2Fs0165-4896%2802%2900016-1.

BUSEMEYER, Jerome R.; James T. TOWNSEND (1992). "Fundamental derivations from decision field theory". In: *Mathematical Social Sciences* 23.3, pp. 255–282. DOI: 10.1016/0165-4896(92)90043-5. URL: https://doi.org/10.1016%2F0165-4896%2892%2990043-5.

CESELLI, Alberto; Ángel FELIPE; M. Teresa ORTUÑO; Giovanni RIGHINI; Gregorio TIRADO (2021). "A Branch-and-Cut-and-Price Algorithm for the Electric Vehicle Routing Problem with Multiple Technologies". In: *SN Operations Research Forum* 2.1. DOI: 10.1007/s43069-020-00052-x. URL: https://doi.org/10.1007%2Fs43069-020-00052-x.

CHAO, I-Ming (2002). "A tabu search method for the truck and trailer routing problem". In: *Computers & Operations Research* 29.1, pp. 33–51. DOI: 10.1016/s0305-0548(00)00056-3. URL: https://doi.org/10.1016%2Fs0305-0548%2800%2900056-3.

CHUXING, Didi (2021). *Milestones*. URL: https://www.didiglobal.com/about-special/milestone. (accessed: February 25, 2022).

CNN (2021). *A History of Lyft, From fuzzy Pink Mustaches to Global Ride Share Giantt*. URL: https://edition.cnn.com/interactive/2019/03/business/lyft-history/index.html. (accessed: February 25, 2022).

CÖMERT, Serap Ercan; Harun Reşit YAZGAN; Irem SERTVURAN; Hanife ŞENGÜL (2017). "A new approach for solution of vehicle routing problem with hard time window: an application in a supermarket chain". In: *Sādhanā* 42.12, pp. 2067–2080. DOI: 10.1007/s12046-017-0754-1. URL: https://doi.org/10.1007%2Fs12046-017-0754-1.

COPPOLA, Pierluigi; Fulvio SILVESTRI (2019). "Autonomous vehicles and future mobility solutions". In: *Autonomous Vehicles and Future Mobility*. Elsevier, pp. 1–15. DOI: 10.1016/b978-0-12-817696-2.00001-9. URL: https://doi.org/10.1016%2Fb978-0-12-817696-2.00001-9.

CORBERÁN, Angel; Christian PRINS (2009). "Recent results on Arc Routing Problems: An annotated bibliography". In: *Networks*, NA–NA. DOI: 10.1002/net.20347. URL: https://doi.org/10.1002%2Fnet.20347.

CORDEAU, Jean-François; Guy DESAULNIERS; Jacques DESROSIERS; Marius M. SOLOMON; François SOUMIS (2002). "7. VRP with Time Windows". In: *The Vehicle Routing Problem*. Chap. 7, pp. 157–193. DOI: 10.1137/1.9780898718515.ch7.

CORDEAU, Jean-François; Michel GENDREAU; Gilbert LAPORTE (1997). "A tabu search heuristic for periodic and multi-depot vehicle routing problems". In: *Networks* 30.2, pp. 105–

119. DOI: https://doi.org/10.1002/(SICI)1097-0037(199709)30:2<105::AID-NET5>3.0.CO;2-G.

CORDEAU, Jean-François; Michel GENDREAU; Gilbert LAPORTE; Jean-Yves POTVIN; Frédéric SEMET (2002). "A Guide to Vehicle Routing Heuristics". In: *The Journal of the Operational Research Society* 53.5, pp. 512–522. ISSN: 01605682, 14769360. URL: http://www.jstor.org/stable/823019.

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: https://doi.org/10.1016/S0191-2615(02)00045-0.

CORDEAU, Jean-François; Gilbert LAPORTE (2005). "Tabu Search Heuristics for the Vehicle Routing Problem". In: *Metaheuristic Optimization via Memory and Evolution: Tabu Search and Scatter Search*. Ed. by Ramesh SHARDA; Stefan VOSS; César REGO; Bahram ALIDAEE. Boston, MA: Springer US, pp. 145–163.

CORDEAU, Jean-François; Gilbert LAPORTE (2006). "Modeling and Optimization of Vehicle Routing and Arc Routing Problems". In: *Handbook on Modelling for Discrete Optimization*. Ed. by Gautam APPA; Leonidas PITSOULIS; H. Paul WILLIAMS. Boston, MA: Springer US, pp. 151–191. ISBN: 978-0-387-32942-0. DOI: 10.1007/0-387-32942-0_6.

CORDEAU, Jean-François; Gilbert LAPORTE; Stefan ROPKE (2007). "Recent Models and Algorithms for One-to-One Pickup and Delivery Problems". In: *Operations Research/Computer Science Interfaces*. Springer US, pp. 327–357. DOI: 10.1007/978-0-387-77778-8_15.

COSTA, Luciano; Claudio CONTARDO; Guy DESAULNIERS (2019). "Exact Branch-Price-and-Cut Algorithms for Vehicle Routing". In: *Transportation Science* 53.4, pp. 946–985. DOI: 10.1287/trsc.2018.0878. URL: https://doi.org/10.1287%2Ftrsc.2018.0878.

DANDL, Florian; Roman ENGELHARDT; Michael HYLAND; Gabriel TILG; Klaus BOGENBERGER; Hani S. MAHMASSANI (2021). "Regulating mobility-on-demand services: Tri-level model and Bayesian optimization solution approach". In: *Transportation Research Part C: Emerging Technologies* 125, p. 103075. DOI: 10.1016/j.trc.2021.103075. URL: https://doi.org/10.1016%2Fj.trc.2021.103075.

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. DOI: 10.1007/s11116-019-10007-9. URL: https://doi.org/10.1007%2Fs11116-019-10007-9.

DANDL, Florian; Michael HYLAND; Klaus BOGENBERGER; Hani S. MAHMASSANI (2020). "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. DOI: 10.1287/mnsc.6.1.80. URL: https://doi.org/10.1287%2Fmnsc.6.1.80.

DESAULNIERS, Guy; Oli B.G. MADSEN; Stefan ROPKE (2014). "Chapter 5: The Vehicle Routing Problem with Time Windows". In: *Vehicle Routing*. Society for Industrial and Applied Mathematics, pp. 119–159. DOI: 10.1137/1.9781611973594.ch5. URL: https://doi.org/10.1137%2F1.9781611973594.ch5.

DESROSIERS, Jacques; Yvan DUMAS; Marius M. SOLOMON; François SOUMIS (1995). "Chapter 2 Time constrained routing and scheduling". In: *Handbooks in Operations Research and Management Science*. Elsevier, pp. 35–139. DOI: 10.1016/s0927-0507(05)80106-9. URL: https://doi.org/10.1016%2Fs0927-0507%2805%2980106-9.

DESROSIERS, Jacques; François SOUMIS; Martin DESROCHERS (1984). "Routing with time windows by column generation". In: *Networks* 14.4, pp. 545–565. DOI: 10.1002/net.3230140406. URL: https://doi.org/10.1002%2Fnet.3230140406.

DIJKSTRA, E. W. (1959). "A note on two problems in connexion with graphs". In: *Numerische Mathematik* 1.1, pp. 269–271. DOI: 10.1007/bf01386390.

DOERNER, Karl F.; Juan-José SALAZAR-GONZÁLEZ (2014). "Chapter 7: Pickup-and-Delivery Problems for People Transportation". In: *Vehicle Routing*. Society for Industrial and Applied Mathematics, pp. 193–212. DOI: 10.1137/1.9781611973594.ch7. URL: https://doi.org/10.1137%2F1.9781611973594.ch7.

DORIGO, Marco; Thomas STÜTZLE (2010). "Ant Colony Optimization: Overview and Recent Advances". In: *Handbook of Metaheuristics*. Springer US, pp. 227–263. DOI: 10.1007/978-1-4419-1665-5_8. URL: https://doi.org/10.1007%2F978-1-4419-1665-5_8.

DREXL, Michael (2012). "Rich vehicle routing in theory and practice". In: *Logistics Research* 5.1-2, pp. 47–63. DOI: 10.1007/s12159-012-0080-2. URL: https://doi.org/10.1007%2Fs12159-012-0080-2.

D'SOUZA, Craig; S.N. OMKAR; J. SENTHILNATH (2012). "Pickup and delivery problem using metaheuristics techniques". In: *Expert Systems with Applications* 39.1, pp. 328–334. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2011.07.022.

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. DOI: 10.1109/itsc.2019.8916955. URL: https://doi.org/10.1109%2Fitsc.2019.8916955.

ERDMANN, Marvin; Florian DANDL; Klaus BOGENBERGER (2019). "Dynamic Car-Passenger Matching based on Tabu Search using Global Optimization with Time Windows". In: *2019 8th International Conference on Modeling Simulation and Applied Optimization (ICMSAO)*. IEEE. DOI: 10.1109/icmsao.2019.8880293. URL: https://doi.org/10.1109%2Ficmsao.2019.8880293.

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. DOI: 10.1016/j.trc.2021.103104. URL: https://doi.org/10.1016%2Fj.trc.2021.103104.

ERDMANN, Marvin; Florian DANDL; Bernd KALTENHÄUSER; Klaus BOGENBERGER (2020). "Dynamic Car-Passenger Matching of Online and Reservation Requests". In: *99th Annual Meeting of the Transportation Research Board (TRB 2020)*.

FAGNANT, Daniel J.; Kara M. KOCKELMAN; Prateek BANSAL (2016). "Operations of Shared Autonomous Vehicle Fleet for Austin, Texas, Market". In: *Transportation Research Record: Journal of the Transportation Research Board* 2563.1, pp. 98–106. DOI: 10.3141/2536-12. URL: https://doi.org/10.3141%2F2536-12.

Fielbaum, Andrés; Javier Alonso-Mora (2020). "Unreliability in ridesharing systems: Measuring changes in users' times due to new requests". In: *Transportation Research Part C: Emerging Technologies* 121, p. 102831. issn: 0968-090X. doi: https://doi.org/10.1016/j.trc.2020.102831.

Fitzsimmons, Emma G.; Winnie Hu; McGeehan Patrick (2018). "The New York Area Was Nearly Paralyzed by 6 Inches of Snow. What Went Wrong?" In: *The New York Times*. url: https://www.nytimes.com/2018/11/16/nyregion/snowstorm-total-delays-commute.html.

Fukasawa, Ricardo; Jens Lysgaard; Marcus Poggi de Aragão; Marcelo Reis; Eduardo Uchoa; Renato F. Werneck (2004). "Robust Branch-and-Cut-and-Price for the Capacitated Vehicle Routing Problem". In: *Integer Programming and Combinatorial Optimization*. Springer Berlin Heidelberg, pp. 1–15. doi: 10.1007/978-3-540-25960-2_1. url: https://doi.org/10.1007%2F978-3-540-25960-2_1.

Gendreau, Michel; François Guertin; Jean-Yves Potvin; Éric Taillard (1999). "Parallel Tabu Search for Real-Time Vehicle Routing and Dispatching". In: *Transportation Science* 33.4, pp. 381–390. doi: 10.1287/trsc.33.4.381. url: https://doi.org/10.1287%2Ftrsc.33.4.381.

Gendreau, Michel; Alain Hertz; Gilbert Laporte (1992). "New Insertion and Postoptimization Procedures for the Traveling Salesman Problem". In: *Operations Research* 40.6, pp. 1086–1094. doi: 10.1287/opre.40.6.1086. url: https://doi.org/10.1287%2Fopre.40.6.1086.

Gendreau, Michel; Ola Jabali; Walter Rei (2014). "Chapter 8: Stochastic Vehicle Routing Problems". In: *Vehicle Routing*. Society for Industrial and Applied Mathematics, pp. 213–239. doi: 10.1137/1.9781611973594.ch8. url: https://doi.org/10.1137%2F1.9781611973594.ch8.

Gendreau, Michel; Gilbert Laporte; Jean-Yves Potvin (2002). "6. Metaheuristics for the Capacitated VRP". In: *The Vehicle Routing Problem*. Chap. 6, pp. 129–154. doi: 10.1137/1.9780898718515.ch6.

Gendreau, Michel; Jean-Yves Potvin (2010). "Tabu Search". In: *Handbook of Metaheuristics*. Springer US, pp. 41–59. doi: 10.1007/978-1-4419-1665-5_2. url: https://doi.org/10.1007%2F978-1-4419-1665-5_2.

Glover, Fred W. (2013). "Tabu Search". In: *Encyclopedia of Operations Research and Management Science*. Springer US, pp. 1537–1544. doi: 10.1007/978-1-4419-1153-7_1034. url: https://doi.org/10.1007%2F978-1-4419-1153-7_1034.

Gurobi Optimization, LLC (2021). *Gurobi Optimizer Reference Manual*. url: http://www.gurobi.com.

Harris, Charles R. et al. (2020). "Array programming with NumPy". In: *Nature* 585, pp. 357–362. doi: 10.1038/s41586-020-2649-2.

Henao, Alejandro; Wesley E. Marshall (2018). "The impact of ride-hailing on vehicle miles traveled". In: *Transportation* 46.6, pp. 2173–2194. doi: 10.1007/s11116-018-9923-2. url: https://doi.org/10.1007%2Fs11116-018-9923-2.

Herbawi, W.; M. Weber (2012). "The ridematching problem with time windows in dynamic ridesharing: A model and a genetic algorithm". In: *2012 IEEE Congress on Evolutionary Computation*, pp. 1–8. doi: 10.1109/CEC.2012.6253001.

Ho, Sin C.; Michel Gendreau (2006). "Path relinking for the vehicle routing problem". In: *Journal of Heuristics* 12.1-2, pp. 55–72. doi: 10.1007/s10732-006-4192-1. url: https://doi.org/10.1007%2Fs10732-006-4192-1.

Ho, Sin C.; W.Y. Szeto; Yong-Hong Kuo; Janny M.Y. Leung; Matthew Petering; Terence W.H. Tou (2018). "A survey of dial-a-ride problems: Literature review and recent developments". In: *Transportation Research Part B: Methodological* 111, pp. 395–421. doi: 10.1016/j.trb.2018.02.001. url: https://doi.org/10.1016%2Fj.trb.2018.02.001.

Ho, Songguang; Sarat Chandra Nagavarapu; Ramesh Ramasamy Pandi; Justin Dauwels (2018). *An Improved Tabu Search Heuristic for Static Dial-A-Ride Problem*. arXiv: 1801.09547 [cs.AI].

Holland, John H. (1975). *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. University of Michigan Press. isbn: 9780472084609.

Horn, Mark E.T. (2002). "Fleet scheduling and dispatching for demand-responsive passenger services". In: *Transportation Research Part C: Emerging Technologies* 10.1, pp. 35–63. doi: 10.1016/s0968-090x(01)00003-1. url: https://doi.org/10.1016%2Fs0968-090x%2801%2900003-1.

Hunter, John D (2007). "Matplotlib: A 2D graphics environment". In: *Computing in science & engineering* 9.3, pp. 90–95.

Hyland, Michael; Florian Dandl; Klaus Bogenberger; Hani S. Mahmassani (2020). "Integrating demand forecasts into the operational strategies of shared automated vehicle mobility services: spatial resolution impacts". In: *Transportation Letters* 12.10, pp. 671–676. doi: 10.1080/19427867.2019.1691297.

Hyland, Michael; Hani S. Mahmassani (2017). "Taxonomy of Shared Autonomous Vehicle Fleet Management Problems to Inform Future Transportation Mobility". In: *Transportation Research Record* 2653.1, pp. 26–34. doi: 10.3141/2653-04.

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: https://doi.org/10.1016/j.trc.2018.05.003.

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: https://doi.org/10.1016/j.tra.2020.02.017.

IBM ILOG CPLEX (2017). "V12.8: CPLEX User's Manual". In: *International Business Machines Corporation.*

Ihler, Alexander; Jon Hutchins; Padhraic Smyth (2006). "Adaptive event detection with time-varying poisson processes". In: *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '06*. ACM Press. doi: 10.1145/1150402.1150428. url: https://doi.org/10.1145%2F1150402.1150428.

Impagliazzo, Russell; Ramamohan Paturi (2001). "On the Complexity of k-SAT". In: *Journal of Computer and System Sciences* 62.2, pp. 367–375. doi: 10.1006/jcss.2000.1727. url: https://doi.org/10.1006%2Fjcss.2000.1727.

INTERNATIONAL TELECOMMUNICATION UNION AND UNITED NATIONS (2021). *Measuring digital development: Facts and figures 2020*. URL: https://www.itu.int/en/ITU-D/Statistics/Pages/facts/default.aspx. (accessed: February 25, 2022).

IRNICH, Stefan; Michael SCHNEIDER; Daniele VIGO (2014). "Chapter 9: Four Variants of the Vehicle Routing Problem". In: *Vehicle Routing*. Society for Industrial and Applied Mathematics, pp. 241–271. DOI: 10.1137/1.9781611973594.ch9. URL: https://doi.org/10.1137%2F1.9781611973594.ch9.

IRNICH, Stefan; Paolo TOTH; Daniele VIGO (2014). "Chapter 1: The Family of Vehicle Routing Problems". In: *Vehicle Routing*. Society for Industrial and Applied Mathematics, pp. 1–33. DOI: 10.1137/1.9781611973594.ch1. URL: https://doi.org/10.1137%2F1.9781611973594.ch1.

JAILLET, Patrick; Michael R. WAGNER (2008). "Online Vehicle Routing Problems: A Survey". In: *Operations Research/Computer Science Interfaces*. Springer US, pp. 221–237. DOI: 10.1007/978-0-387-77778-8_10. URL: https://doi.org/10.1007%2F978-0-387-77778-8_10.

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. DOI: 10.1016/0191-2615(86)90020-2. URL: https://doi.org/10.1016%2F0191-2615%2886%2990020-2.

JORGENSEN, R M; J LARSEN; K B BERGVINSDOTTIR (2007). "Solving the Dial-a-Ride problem using genetic algorithms". In: *Journal of the Operational Research Society* 58.10, pp. 1321–1331. DOI: 10.1057/palgrave.jors.2602287.

AL-KANJ, Lina; Juliana NASCIMENTO; Warren B. POWELL (2020). "Approximate dynamic programming for planning a ride-hailing system using autonomous fleets of electric vehicles". In: *European Journal of Operational Research* 284.3, pp. 1088–1106. DOI: 10.1016/j.ejor.2020.01.033. URL: https://doi.org/10.1016%2Fj.ejor.2020.01.033.

KARP, Richard M. (1972). "Reducibility among Combinatorial Problems". In: *Complexity of Computer Computations*. Springer US, pp. 85–103. DOI: 10.1007/978-1-4684-2001-2_9. URL: https://doi.org/10.1007%2F978-1-4684-2001-2_9.

KOLLEWE, Julia; THE GUARDIAN (2019). *UK: Uber drivers strike over poor pay & working conditions*. URL: https://www.business-humanrights.org/en/latest-news/uk-uber-drivers-strike-over-poor-pay-working-conditions. (accessed: February 25, 2022).

LAPORTE, Gilbert (1992). "The traveling salesman problem: An overview of exact and approximate algorithms". In: *European Journal of Operational Research* 59.2, pp. 231–247. DOI: 10.1016/0377-2217(92)90138-y. URL: https://doi.org/10.1016%2F0377-2217%2892%2990138-y.

LAPORTE, Gilbert; Stefan ROPKE; Thibaut VIDAL (2014). "Chapter 4: Heuristics for the Vehicle Routing Problem". In: *Vehicle Routing*. Society for Industrial and Applied Mathematics, pp. 87–116. DOI: 10.1137/1.9781611973594.ch4. URL: https://doi.org/10.1137%2F1.9781611973594.ch4.

LEE, Der-Horng; Hao WANG; Ruey Long CHEU; Siew Hoon TEO (2004). "Taxi Dispatch System Based on Current Demands and Real-Time Traffic Conditions". In: *Transportation*

Research Record: Journal of the Transportation Research Board 1882.1, pp. 193–200. DOI: 10.3141/1882-23. URL: https://doi.org/10.3141%2F1882-23.

LI, Xiangyong; Stephen C.H. LEUNG; Peng TIAN (2012). "A multistart adaptive memory-based tabu search algorithm for the heterogeneous fixed fleet open vehicle routing problem". In: *Expert Systems with Applications* 39.1, pp. 365–374. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2011.07.025.

LITTLE, John D. C.; Katta G. MURTY; Dura W. SWEENEY; Caroline KAREL (1963). "An Algorithm for the Traveling Salesman Problem". In: *Operations Research* 11.6, pp. 972–989. DOI: 10.1287/opre.11.6.972. URL: https://doi.org/10.1287%2Fopre.11.6.972.

LUND, K.; O.B.G. MADSEN; J.M. RYGAARD (1996). *Vehicle Routing Problems with Varying Degrees of Dynamism*. IMM-REP. IMM, Institute of Mathematical Modelling, Technical University of Denmark.

MACIEJEWSKI, Michal; Joschka BISCHOFF (2015). "Large-scale Microscopic Simulation of Taxi Services". In: *Procedia Computer Science* 52, pp. 358–364. DOI: 10.1016/j.procs.2015.05.107. URL: https://doi.org/10.1016%2Fj.procs.2015.05.107.

MADSEN, O.B.G.; H.F. RAVN; J.M RYGAARD (1995). "A heuristic algorithm for a dial-a-ride problem with time windows, multiple capacities, and multiple objectives". In: *Annals of Operations Research* 60, pp. 193–208.

MATAI, Rajesh; Surya SINGH; Murari LAL (2010). "Traveling Salesman Problem: an Overview of Applications, Formulations, and Solution Approaches". In: *Traveling Salesman Problem, Theory and Applications*. InTech. DOI: 10.5772/12909. URL: https://doi.org/10.5772%2F12909.

MCKINNEY, Wes et al. (2010). "Data structures for statistical computing in python". In: *Proceedings of the 9th Python in Science Conference*. Vol. 445. Austin, TX, pp. 51–56.

MCKINSEY CENTER FOR FUTURE MOBILITY (2020a). *From no mobility to future mobility: Where COVID-19 has accelerated change*. URL: https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/from-no-mobility-to-future-mobility-where-covid-19-has-accelerated-change. (accessed: February 25, 2022).

MCKINSEY CENTER FOR FUTURE MOBILITY (2020b). *Why shared mobility is poised to make a comeback after the crisis*. URL: https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/why-shared-mobility-is-poised-to-make-a-comeback-after-the-crisis. (accessed: February 25, 2022).

MIN, Hokey (1989). "The multiple vehicle routing problem with simultaneous delivery and pick-up points". In: *Transportation Research Part A: General* 23.5, pp. 377–386. DOI: 10.1016/0191-2607(89)90085-x. URL: https://doi.org/10.1016%2F0191-2607%2889%2990085-x.

MITROVIĆ-MINIĆ, Snežana; Ramesh KRISHNAMURTI; Gilbert LAPORTE (2004). "Double-horizon based heuristics for the dynamic pickup and delivery problem with time windows". In: *Transportation Research Part B: Methodological* 38.8, pp. 669–685. DOI: 10.1016/j.trb.2003.09.001. URL: https://doi.org/10.1016%2Fj.trb.2003.09.001.

MORABIT, M.; G. DESAULNIERS; A. LODI (2020). *Machine-Learning-Based Column Selection for Column Generation*. Les Cahiers du GERAD. GERAD, HEC Montréal.

Naddef, Denis; Giovanni Rinaldi (2002). "3. Branch-And-Cut Algorithms for the Capacitated VRP". In: *The Vehicle Routing Problem*. Chap. 3, pp. 53–84. doi: 10.1137/1.9780898718515.ch3.

Nair, Gopindra S.; Chandra R. Bhat; Irfan Batur; Ram M. Pendyala; William H.K. Lam (2020). "A model of deadheading trips and pick-up locations for ride-hailing service vehicles". English (US). In: *Transportation Research, Part A: Policy and Practice* 135, pp. 289–308. issn: 0965-8564. doi: 10.1016/j.tra.2020.03.015.

Naji-Azimi, Zahra; Majid Salari (2013). "A complementary tool to enhance the effectiveness of existing methods for heterogeneous fixed fleet vehicle routing problem". In: *Applied Mathematical Modelling* 37.6, pp. 4316–4324. doi: 10.1016/j.apm.2012.09.027. url: https://doi.org/10.1016%2Fj.apm.2012.09.027.

National Institute for Transportation and Communities (2014). *Lessons from the Green Lanes: Evaluating Protected Bike Lanes in the U.S.* url: https://nitc.trec.pdx.edu/research/project/583. (accessed: February 25, 2022).

Nazari, MohammadReza; Afshin Oroojlooy; Lawrence Snyder; Martin Takac (2018). "Reinforcement Learning for Solving the Vehicle Routing Problem". In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio; H. Wallach; H. Larochelle; K. Grauman; N. Cesa-Bianchi; R. Garnett. Vol. 31. Curran Associates, Inc.

Nikolaev, Alexander G.; Sheldon H. Jacobson (2010). "Simulated Annealing". In: *Handbook of Metaheuristics*. Springer US, pp. 1–39. doi: 10.1007/978-1-4419-1665-5_1. url: https://doi.org/10.1007%2F978-1-4419-1665-5_1.

NYC Taxi & Limousine Commision (2021). *TLC trip record data*. url: https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page. (accessed: February 25, 2022).

Osman, Ibrahim H. (1993). "Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem". In: *Annals of Operations Research* 41, pp. 421–451.

Osman, Ibrahim H.; Niaz A. Wassan (2002). "A reactive tabu search meta-heuristic for the vehicle routing problem with back-hauls". In: *Journal of Scheduling* 5.4, pp. 263–285. doi: 10.1002/jos.122. url: https://doi.org/10.1002%2Fjos.122.

Pandi, R. R.; S. G. Ho; S. C. Nagavarapu; T. Tripathy; J. Dauwels (2018). "GPU-Accelerated Tabu Search Algorithm for Dial-A-Ride Problem". In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 2519–2524. doi: 10.1109/ITSC.2018.8569472.

Pecin, Diego; Artur Pessoa; Marcus Poggi; Eduardo Uchoa (2016). "Improved branch-cut-and-price for capacitated vehicle routing". In: *Mathematical Programming Computation* 9.1, pp. 61–100. doi: 10.1007/s12532-016-0108-8. url: https://doi.org/10.1007%2Fs12532-016-0108-8.

Pillac, Victor; Michel Gendreau; Christelle Guéret; Andrés L. Medaglia (2013). "A review of dynamic vehicle routing problems". In: *European Journal of Operational Research* 225.1, pp. 1–11. issn: 0377-2217. doi: https://doi.org/10.1016/j.ejor.2012.08.015.

Pinto, Helen K.R.F.; Michael 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. DOI: 10.1016/j.trc.2019.06.010. URL: https://doi.org/10.1016%2Fj.trc.2019.06.010.

PORRU, Simone; Francesco Edoardo MISSO; Filippo Eros PANI; Cino REPETTO (2020). "Smart mobility and public transport: Opportunities and challenges in rural and urban areas". In: *Journal of Traffic and Transportation Engineering (English Edition)* 7.1, pp. 88–97. DOI: 10.1016/j.jtte.2019.10.002. URL: https://doi.org/10.1016%2Fj.jtte.2019.10.002.

PRINS, Christian (2004). "A simple and effective evolutionary algorithm for the vehicle routing problem". In: *Computers & Operations Research* 31.12, pp. 1985–2002. DOI: 10.1016/s0305-0548(03)00158-8. URL: https://doi.org/10.1016%2Fs0305-0548%2803%2900158-8.

PRODHON, Caroline; Christian PRINS (2016). "Metaheuristics for Vehicle Routing Problems". In: *Metaheuristics*. Ed. by Patrick SIARRY. Cham: Springer International Publishing, pp. 407–437. ISBN: 978-3-319-45403-0. DOI: 10.1007/978-3-319-45403-0_15.

PROSSER, Patrick; Paul SHAW (1997). "Study of Greedy Search with Multiple Improvement Heuristics for Vehicle Routing Problems". In:

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. DOI: 10.1287/trsc.14.2.130. URL: https://doi.org/10.1287%2Ftrsc.14.2.130.

PSARAFTIS, Harilaos N.; Min WEN; Christos A. KONTOVAS (2015). "Dynamic vehicle routing problems: Three decades and counting". In: *Networks* 67.1, pp. 3–31. DOI: 10.1002/net.21628. URL: https://doi.org/10.1002%2Fnet.21628.

PYTHON SOFTWARE FOUNDATION (2021). *Python 3.7.12 documentation*. URL: https://docs.python.org/3.7.

RAIDL, Günther R.; Jakob PUCHINGER; Christian BLUM (2010). "Metaheuristic Hybrids". In: *Handbook of Metaheuristics*. Springer US, pp. 469–496. DOI: 10.1007/978-1-4419-1665-5_16. URL: https://doi.org/10.1007%2F978-1-4419-1665-5_16.

RATCLIFF, Roger; Philip L. SMITH; Scott D. BROWN; Gail MCKOON (2016). "Diffusion Decision Model: Current Issues and History". In: *Trends in Cognitive Sciences* 20.4, pp. 260–281. DOI: 10.1016/j.tics.2016.01.007. URL: https://doi.org/10.1016%2Fj.tics.2016.01.007.

RENAUD, Jacques; Gilbert LAPORTE; Fayez F. BOCTOR (1996). "A tabu search heuristic for the multi-depot vehicle routing problem". In: *Computers & Operations Research* 23.3, pp. 229–235. DOI: 10.1016/0305-0548(95)o0026-p. URL: https://doi.org/10.1016%2F0305-0548%2895%29o0026-p.

RESENDE, Mauricio G.C.; Celso C. RIBEIRO (2010). "Greedy Randomized Adaptive Search Procedures: Advances, Hybridizations, and Applications". In: *Handbook of Metaheuristics*. Springer US, pp. 283–319. DOI: 10.1007/978-1-4419-1665-5_10. URL: https://doi.org/10.1007%2F978-1-4419-1665-5_10.

ROBINSON, Julia B. (1949). *On the Hamiltonian game (a traveling-salesman problem)*. RAND Corporation.

ROPKE, Stefan; David PISINGER (2006a). "A unified heuristic for a large class of Vehicle Routing Problems with Backhauls". In: *European Journal of Operational Research* 171.3, pp. 750–775. DOI: 10.1016/j.ejor.2004.09.004. URL: https://doi.org/10.1016%2Fj.ejor.2004.09.004.

Ropke, Stefan; David Pisinger (2006b). “An Adaptive Large Neighborhood Search Heuristic for the Pickup and Delivery Problem with Time Windows”. In: *Transportation Science* 40.4, pp. 455–472. DOI: 10.1287/trsc.1050.0135. URL: https://doi.org/10.1287%2Ftrsc.1050.0135.

Savelsbergh, Martin W. P.; Marc Sol (1995). “The General Pickup and Delivery Problem”. In: *Transportation Science* 29.1, pp. 17–29. DOI: 10.1287/trsc.29.1.17. URL: https://doi.org/10.1287%2Ftrsc.29.1.17.

Sayarshad, Hamid R.; Joseph Y. J. Chow (2016). “Survey and empirical evaluation of nonhomogeneous arrival process models with taxi data”. In: *Journal of Advanced Transportation* 50.7, pp. 1275–1294. DOI: 10.1002/atr.1401. URL: https://doi.org/10.1002%2Fatr.1401.

Schaller Consulting (2018a). *Making Congestion Pricing Work for Traffic and Transit in New York City*. URL: http://www.schallerconsult.com/rideservices/makingpricingwork.htm. (accessed: February 25, 2022).

Schaller Consulting (2018b). *The New Automobility: Lyft, Uber and the Future of American Cities*. URL: http://www.schallerconsult.com/rideservices/automobility.htm. (accessed: February 25, 2022).

Semet, Frédéric; Paolo Toth; Daniele Vigo (2014). “Chapter 2: Classical Exact Algorithms for the Capacitated Vehicle Routing Problem”. In: *Vehicle Routing*. Society for Industrial and Applied Mathematics, pp. 37–57. DOI: 10.1137/1.9781611973594.ch2. URL: https://doi.org/10.1137%2F1.9781611973594.ch2.

Shaheen, Susan; Adam Cohen; Nelson Chan; Apaar Bansal (2020). “Sharing strategies: carsharing, shared micromobility (bikesharing and scooter sharing), transportation network companies, microtransit, and other innovative mobility modes”. In: *Transportation, Land Use, and Environmental Planning*. Elsevier, pp. 237–262. DOI: 10.1016/b978-0-12-815167-9.00013-x. URL: https://doi.org/10.1016%2Fb978-0-12-815167-9.00013-x.

Shaw, Paul (1998). “Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems”. In: *Principles and Practice of Constraint Programming — CP98*. Springer Berlin Heidelberg, pp. 417–431. DOI: 10.1007/3-540-49481-2_30. URL: https://doi.org/10.1007%2F3-540-49481-2_30.

Sheridan, Patricia Kristine; Erich Gluck; Qi Guan; Thomas Pickles; Barış Balcioğlu; Beno Benhabib (2013). “The dynamic nearest neighbor policy for the multi-vehicle pick-up and delivery problem”. In: *Transportation Research Part A: Policy and Practice* 49, pp. 178–194. DOI: 10.1016/j.tra.2013.01.032. URL: https://doi.org/10.1016%2Fj.tra.2013.01.032.

Syed, Arslan Ali; Irina Gaponova; Klaus Bogenberger (2019). “Neural Network-Based Metaheuristic Parameterization with Application to the Vehicle Matching Problem in Ride-Hailing Services”. In: *Transportation Research Record: Journal of the Transportation Research Board* 2673.10, pp. 311–320. DOI: 10.1177/0361198119846099. URL: https://doi.org/10.1177%2F0361198119846099.

Syed, Arslan Ali; Bernd Kaltenhäuser; Irina Gaponova; Klaus Bogenberger (2019). "Asynchronous Adaptive Large Neighborhood Search Algorithm for Dynamic Matching Problem in Ride Hailing Services". In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE. doi: 10.1109/itsc.2019.8916943. url: https://doi.org/10.1109%2Fitsc.2019.8916943.

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. doi: 10.1038/srep42868. url: https://doi.org/10.1038%2Fsrep42868.

Taillard, Éric (1999). "A heuristic column generation method for the heterogeneous fleet VRP". In: *RAIRO - Operations Research* 33.1, pp. 1–14. doi: 10.1051/ro:1999101. url: https://doi.org/10.1051%2Fro%3A1999101.

Talbi, El-Ghazali (2016). "Combining metaheuristics with mathematical programming, constraint programming and machine learning". In: *Transportation Research Part C: Emerging Technologies* 240, pp. 171–215. doi: https://doi.org/10.1007/s10479-015-2034-y.

Tan, K.C; L.H Lee; Q.L Zhu; K Ou (2001). "Heuristic methods for vehicle routing problem with time windows". In: *Artificial Intelligence in Engineering* 15.3, pp. 281–295. issn: 0954-1810. doi: https://doi.org/10.1016/S0954-1810(01)00005-X.

Texas A&M Transportation Institute (2021). *Urban Mobility Report 2021*. url: https://static.tti.tamu.edu/tti.tamu.edu/documents/mobility-report-2021.pdf. (accessed: February 25, 2022).

Theisen, Maximilian; Veronika Lerche; Mischa von Krause; Andreas Voss (2020). "Age differences in diffusion model parameters: a meta-analysis". In: *Psychological Research*. doi: 10.1007/s00426-020-01371-8. url: https://doi.org/10.1007%2Fs00426-020-01371-8.

Thurstone, L. L. (1927). "A law of comparative judgment." In: *Psychological Review* 34.4, pp. 273–286. doi: 10.1037/h0070288. url: https://doi.org/10.1037%2Fh0070288.

Toth, Paolo; Daniele Vigo (1997). "An Exact Algorithm for the Vehicle Routing Problem with Backhauls". In: *Transportation Science* 31.4, pp. 372–385. doi: 10.1287/trsc.31.4.372. url: https://doi.org/10.1287%2Ftrsc.31.4.372.

Toth, Paolo; Daniele Vigo (2002a). "1. An Overview of Vehicle Routing Problems". In: *The Vehicle Routing Problem*. Chap. 1, pp. 1–26. doi: 10.1137/1.9780898718515.ch1.

Toth, Paolo; Daniele Vigo (2002b). "2. Branch-And-Bound Algorithms for the Capacitated VRP". In: *The Vehicle Routing Problem*. Chap. 2, pp. 29–51. doi: 10.1137/1.9780898718515.ch2.

Toth, Paolo; Daniele Vigo (2003). "The Granular Tabu Search and Its Application to the Vehicle-Routing Problem". In: *INFORMS Journal on Computing* 15.4, pp. 333–346. doi: 10.1287/ijoc.15.4.333.24890.

Train, Kenneth E. (2001). *Discrete Choice Methods with Simulation*. Cambridge University Press. doi: 10.1017/cbo9780511805271. url: https://doi.org/10.1017%2Fcbo9780511805271.

Tripathy, Twinkle; Sarat Chandra Nagavarapu; Kaveh Azizian; Ramesh Ramasamy Pandi; Justin Dauwels (2017). "Solving Dial-A-Ride Problems Using Multiple Ant Colony System with Fleet Size Minimisation". In: *Advances in Intelligent Systems and*

Computing. Springer International Publishing, pp. 325–336. DOI: 10.1007/978-3-319-66939-7_28. URL: https://doi.org/10.1007%2F978-3-319-66939-7_28.

UBER (2021). *The history of Uber*. URL: https://www.uber.com/en-DE/newsroom/history/. (accessed: February 25, 2022).

UNITED NATIONS (2019). *World Urbanization Prospects: The 2018 Revision*. United Nations. DOI: 10.18356/b9e995fe-en. URL: https://doi.org/10.18356%2Fb9e995fe-en.

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. DOI: 10.1016/j.tra.2019.01.018. URL: https://doi.org/10.1016%2Fj.tra.2019.01.018.

WØHLK, Sanne (2008). "A Decade of Capacitated Arc Routing". In: *Operations Research/Computer Science Interfaces*. Springer US, pp. 29–48. DOI: 10.1007/978-0-387-77778-8_2. URL: https://doi.org/10.1007%2F978-0-387-77778-8_2.

XU, Zhe; Zhixin LI; Qingwen GUAN; Dingshui ZHANG; Qiang LI; Junxiao NAN; Chunyang LIU; Wei BIAN; Jieping YE (2018). "Large-Scale Order Dispatch in On-Demand Ride-Hailing Platforms: A Learning and Planning Approach". In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. KDD '18. London, United Kingdom: Association for Computing Machinery, pp. 905–913. ISBN: 9781450355520. DOI: 10.1145/3219819.3219824.

YU, Jiangbo; Michael HYLAND (2020). "A generalized diffusion model for preference and response time: Application to ordering mobility-on-demand services". In: *Transportation Research Part C: Emerging Technologies* 121, p. 102854. ISSN: 0968-090X. DOI: https://doi.org/10.1016/j.trc.2020.102854.

ZHANG, Lingyu; Tao HU; Yue MIN; Guobin WU; Junying ZHANG; Pengcheng FENG; Pinghua GONG; Jieping YE (2017). "A Taxi Order Dispatch Model based On Combinatorial Optimization". In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM. DOI: 10.1145/3097983.3098138. URL: https://doi.org/10.1145%2F3097983.3098138.

ZHANG, Rick; Marco PAVONE (2014). "Control of Robotic Mobility-On-Demand Systems: a Queueing-Theoretical Perspective". In: *Robotics: Science and Systems X*. Robotics: Science and Systems Foundation. DOI: 10.15607/rss.2014.x.026. URL: https://doi.org/10.15607%2Frss.2014.x.026.

ZHANG, Rick; Federico ROSSI; Marco PAVONE (2016). "Model predictive control of autonomous mobility-on-demand systems". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/icra.2016.7487272.

ZHANG, Tao; W.A. CHAOVALITWONGSE; Yuejie ZHANG (2012). "Scatter search for the stochastic travel-time vehicle routing problem with simultaneous pick-ups and deliveries". In: *Computers & Operations Research* 39.10, pp. 2277–2290. DOI: 10.1016/j.cor.2011.11.021. URL: https://doi.org/10.1016%2Fj.cor.2011.11.021.

Appendix

Abbreviations

The following table lists abbreviations used in this work in alphabetical order.

ARP	arc routing problem
CARP	capacitated arc routing problem
CCM	conventional customer model
CPP	Chinese postman problem
CVRP	capacitated vehicle routing problem
DaRP	dial-a-ride problem
DoD	degree of dynamism
DCVRP	distance-constrained capacitated vehicle routing problem
DVRP	distance-constrained vehicle routing problem
FCFS	first-come-first-served
HFVRP	heterogeneous fleet vehicle routing problem
ID	identification number
KPI	key performance indicator
LNS	large neighborhood search
MDVRP	multi-depot vehicle routing problem
MIP	mixed integer programming
NNP	nearest neighbor policy
ODM	on-demand mobility
PDP	pickup-and-delivery problems
PSA	parameter sensitivity analysis
PVRP	periodic vehicle routing problem
RQ	research question
TSP	traveling salesperson problem
TTRP	truck-and-trailer routing problem
USA	United States of America
VRP	vehicle routing problems
VRPB	vehicle routing problem with backhauls
VRPSPD	vehicle routing problem with simultaneous pickup and delivery
VRPWT	vehicle routing problem with time windows
VSP	vehicle scheduling problem

Symbols and Variables

The following table lists symbols and variables used in this work in alphabetical order. Numbers precede Latin letters, which are succeeded by Greek letters.

C	vehicle capacity
c_{dist}	distance costs
c_{fix}	vehicle fix costs
c_z	demand in z
$d(z_1, z_2)$	distance between z_1 and z_2
D_i	driven mileage of vehicle i
D_t	decision state
F_1	term in F_{con} representing the total driven mileage
F_2	term in F_{con} representing the total user waiting time
F_3	term in F_{con} representing the number of served requests
F_{con}	control function
F_{obj}	objective function
i	index of a vehicle out of all vehicles I
I	set of all vehicles
I_k	set of vehicles connected to k
j	index of a customer out of all customers J
J	set of all customers
J_{a}	subset of J with customers who already accepted an offer
J_{s}	subset of J with served customers
k	subset of J
K_i	set of bundles that can be assigned to i
K_j	set of bundles that contain j
M	fleet size
N_i	length of the task queue of vehicle i
$n_{\mathsf{p}}(\xi_{ik})$	number of customers served in ξ_{ik} picked up outside of their pickup time window
o	index of offer
p_{base}	service base fare
p_{dist}	distance fare
p_{tw}	penalty for pickup time window validations
q	offer quality
q_o	quality of o-th offer
R	demand
r^o_{a}	acceptance rate of o-th offer
$r^o_{\mathsf{a,u}}$	rate of unexpected acceptances of o-th offer
$r^o_{\mathsf{r,u}}$	rate of unexpected rejections of o-th offer
R_{s}	served demand
S	feedback rate

t_{boa}	boarding and deboarding time
t_{d}	decision duration
t_{d}^{o}	decision duration for o-th offer
$t_{\mathrm{d,max}}$	maximum decision-making time
t_{f}	final time step of simulation
t_{hor}	repositioning time horizon
t_{lock}	lock time
t_{max}	maximum waiting time
t_{o}	optimization period
t_{pu}	pickup time
t_{r}	interval of repositioning
t_{re}	reaction time
t_{req}	request time
t_{sim}	simulation time
t_{step}	simulation step size
t_{twl}	pickup time window length
t_{w}	waiting time
$t_{\mathrm{w,1}}$	first decisive waiting time
$t_{\mathrm{w,2}}$	second decisive waiting time
$t_{\mathrm{w,k}}(\xi_{ik})$	sum of individual waiting times of customers served in ξ_{ik}
V	valence during each step of random walk in DCM
v_{exc}^{z}	number of excess vehicles in z
v_z	vehicles in z
x_{ik}	binary decision variable
z	repositioning zone
$z_{\mathrm{do},j}$	drop-off location of customer j
$z_{\mathrm{pu},j}$	pickup location of customer j
α	weighing factor of F_1 in F_{con}
β	weighing factor of F_2 in F_{con}
γ	weighing factor of F_3 in F_{con}
Δt	time step size of decision-making process
ϵ	random term in V
θ	decision threshold
ξ_i	task queue of vehicle i
ξ_{ik}	task queue of i that serves k optimally
ϕ	diffusion rate

Additional Results

In addition to the main results presented and described throughout this work, the following figures are meant to provide further insights into the system performance of the evaluated ride hailing and ride pooling service models from Chapters 4 and 5, as well as the customer models from Chapter 6.

To Chapter 4: Ride Hailing Use Case

The evaluation of ride hailing service models in Chapter 4 finds that Service Model 3 ("only global optimization") slightly outperforms the other service models in many KPIs relevant to the service provider and the city the service is run in at the cost of significantly longer response and waiting times for customers. Specifically, the number of requests served with Service Model 3 is the highest between the evaluated service models (see Figure 4.2) while the distance driven emptily by the fleet is the shortest (see Figure 4.4. In combination, this also leads to the highest mean occupancy of vehicles, which is shown in Figure A.1. The fact that even in the ride hailing use case – which does not allow shared rides – the mean occupancy in scenarios with smaller fleets reaches values of above 1 passenger per vehicle comes from the distribution of one to four passengers per request explained in Section 3.4.1.

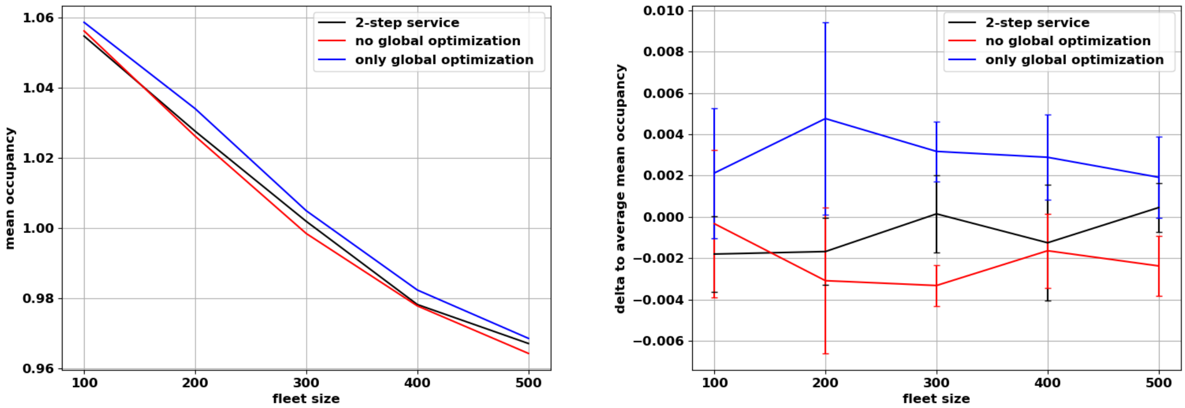
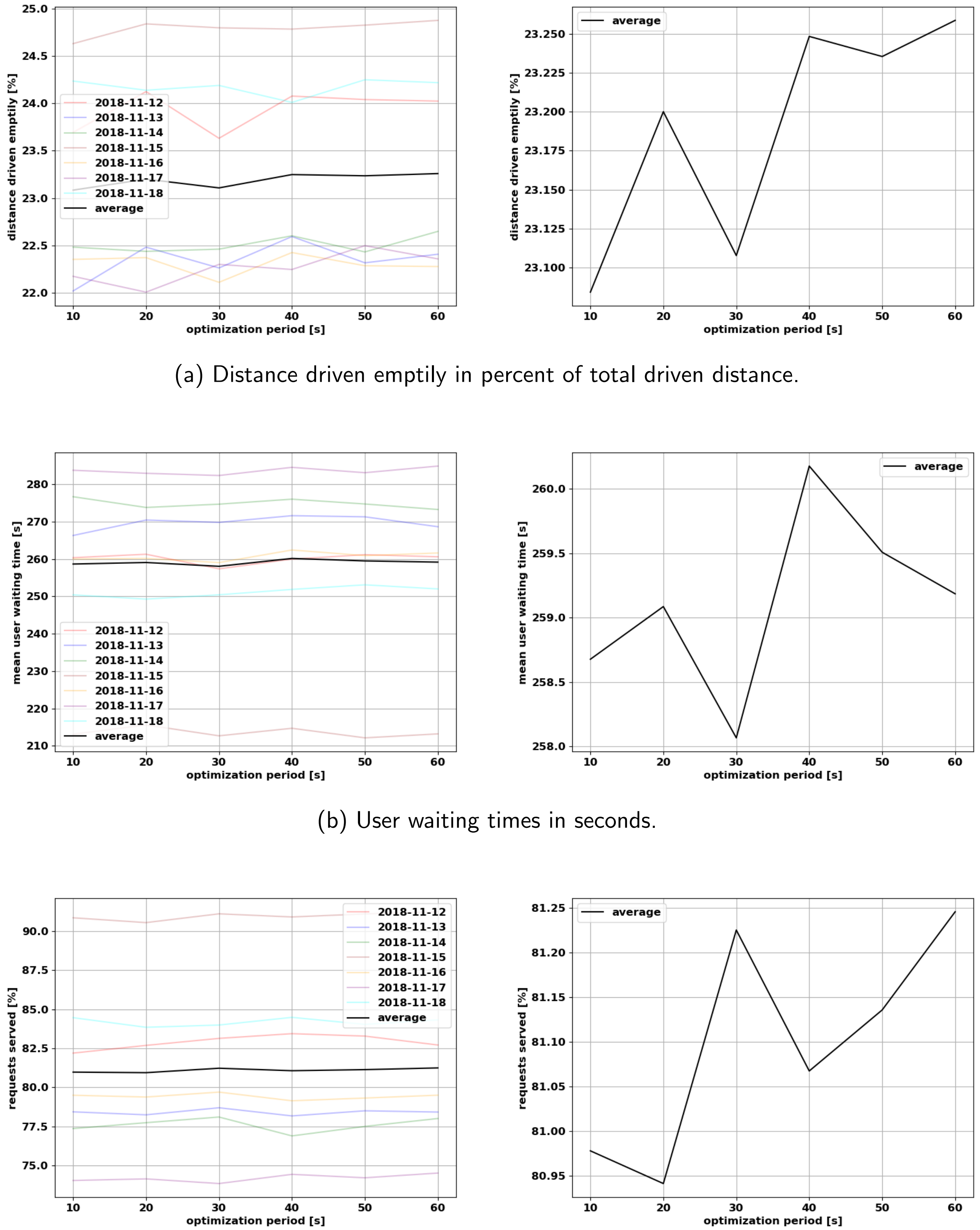


Figure A.1: Occupancy in average passengers per vehicle for various service models in the ride hailing use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

The parameter sensitivity analysis conducted for Service Model 2 showed that the evaluated KPIs are mostly insensitive with respect to the model parameters. Further proof of this observation is given in Figures A.2 to A.5, in which additional performance indicators are presented in dependency to the optimization period, the objective weight of distance, the assignment lock time, and the pickup time window length, respectively.

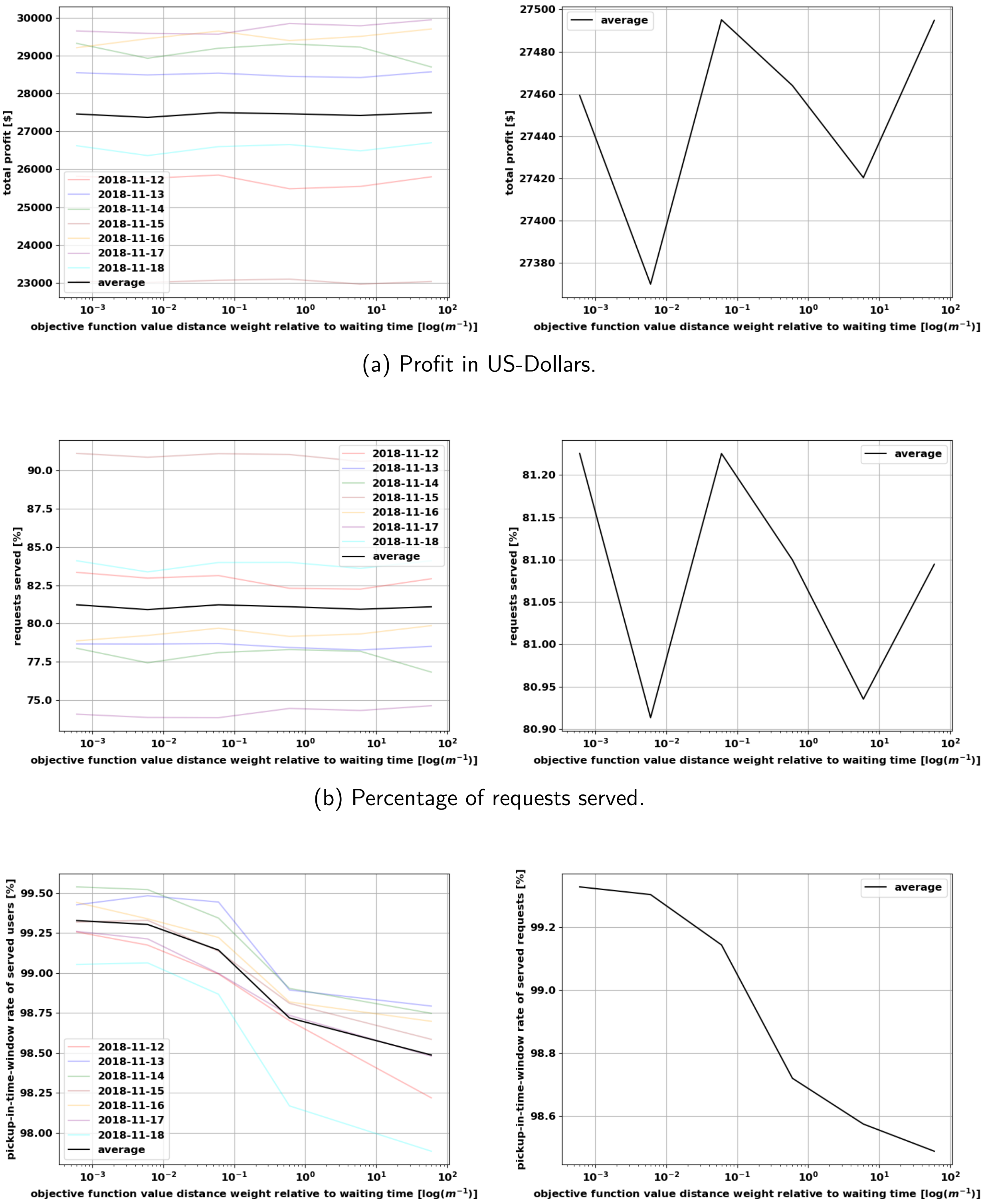


(a) Distance driven emptily in percent of total driven distance.

(b) User waiting times in seconds.

(c) Percentage of requests served.

Figure A.2: Parameter sensitivity analyses in simulations of the 2-step service model with 300 vehicles in the ride hailing use case with values of the optimization period t_o between 10 s and 60 s. Left: values for all simulation dates and the average value. Right: average value in detail.

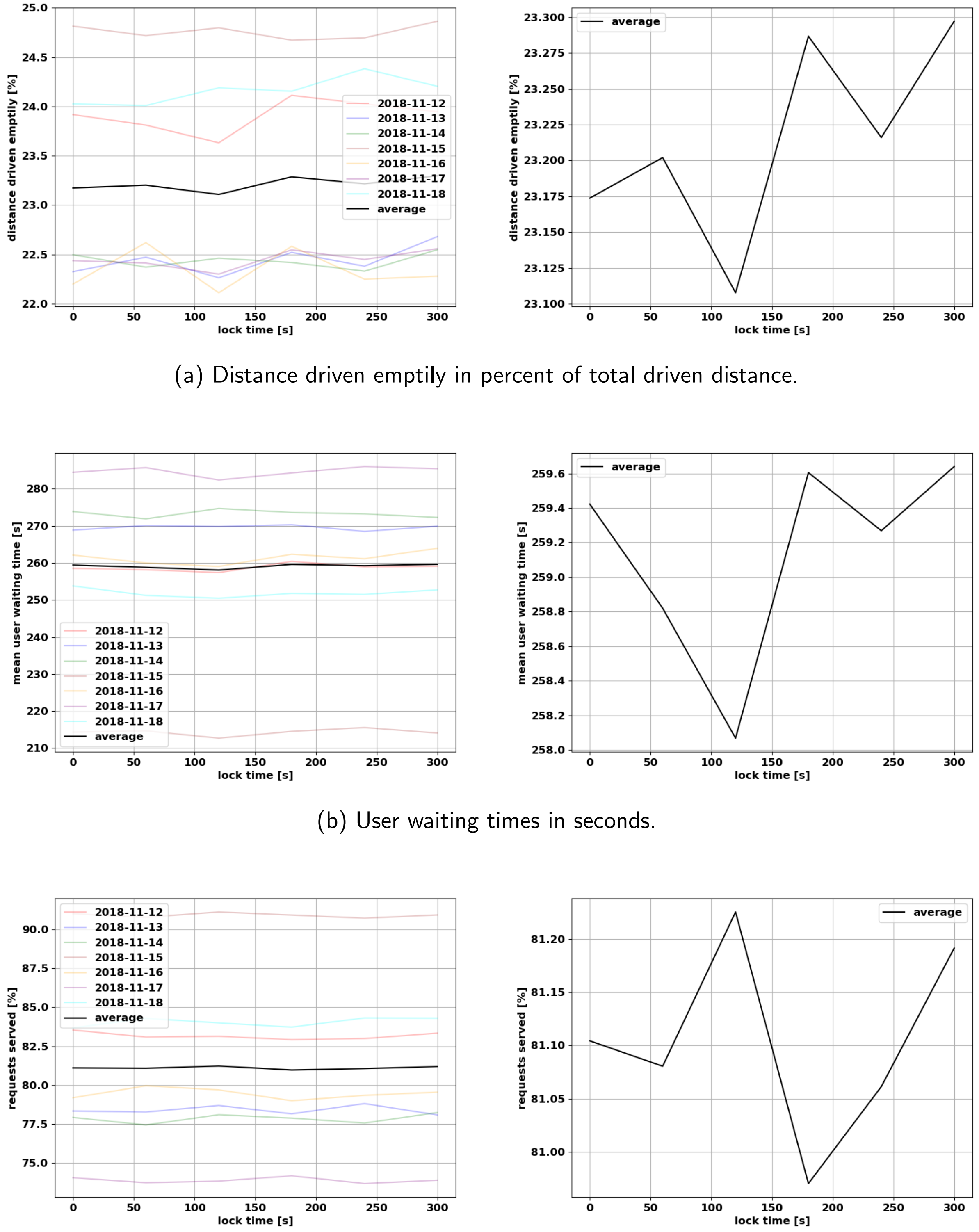


(a) Profit in US-Dollars.

(b) Percentage of requests served.

(c) Pickup-in-time-window rate in percent.

Figure A.3: Parameter sensitivity analyses in simulations of the 2-step service model with 300 vehicles in the ride hailing use case with values of the objective weight of distance α between $6 \times 10^{-4}\,\mathrm{m}^{-1}$ and $6 \times 10^{1}\,\mathrm{m}^{-1}$. Left: values for all simulation dates and the average value. Right: average value in detail.

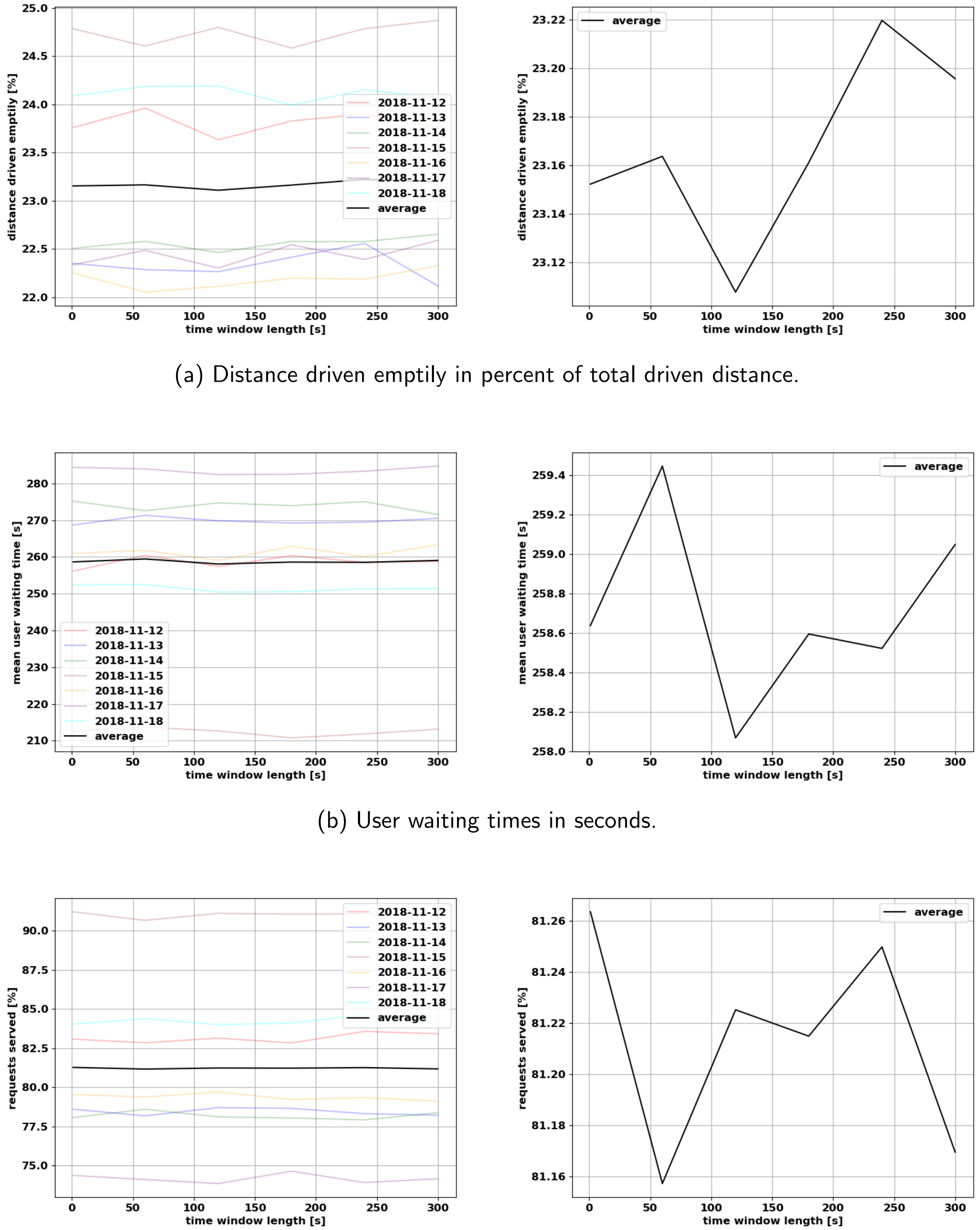


(a) Distance driven emptily in percent of total driven distance.

(b) User waiting times in seconds.

(c) Percentage of requests served.

Figure A.4: Parameter sensitivity analyses in simulations of the 2-step service model with 300 vehicles in the ride hailing use case with values of assignment lock time t_{lock} between 0 min and 5 min. Left: values for all simulation dates and the average value. Right: average value in detail.



(a) Distance driven emptily in percent of total driven distance.

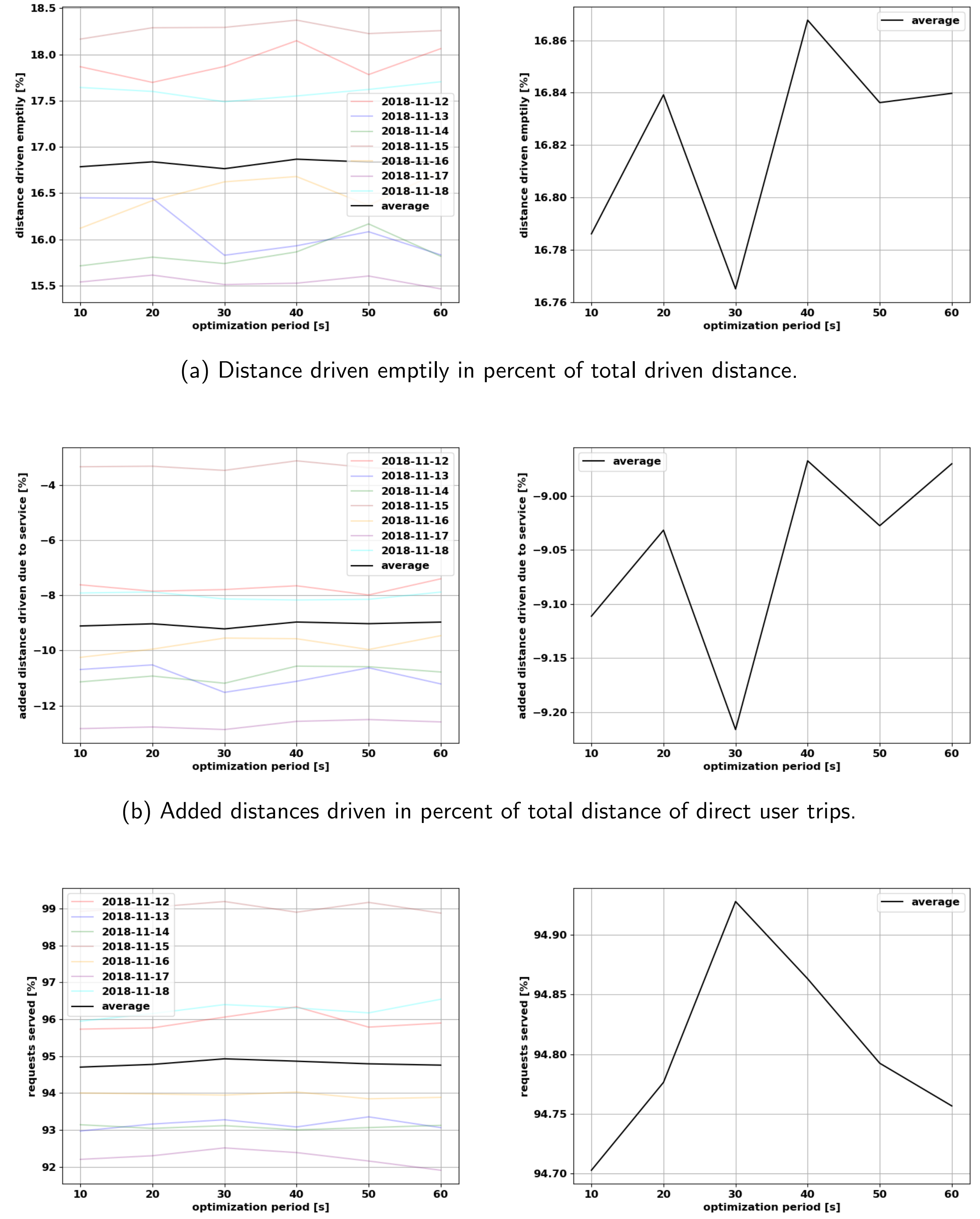
(b) User waiting times in seconds.

(c) Percentage of requests served.

Figure A.5: Parameter sensitivity analyses in simulations of the 2-step service model with 300 vehicles in the ride hailing use case with values of pickup time window lengths t_{twl} between 0 min and 5 min. Left: values for all simulation dates and the average value. Right: average value in detail.

To Chapter 5: Ride Pooling Use Case

Between the ride pooling service models covered in Chapter 5, the 2-step service model is found to be a good trade-off between profitability for the service provider, traffic improvement in the business area in terms of reduced vehicle mileage, and customer-related KPIs such as response and waiting times. The parameter sensitivity analysis of this service model found that the most impactful parameter when it comes to the effect on service KPIs is the objective weight of distance α, which is why this parameter is evaluated in more detail in Section 5.3.2. In order to provide more evidence of the relative insensitivity of most KPIs to parameters other than α, Figures A.6 to A.10 show additional KPIs with their respective dependencies with respect to the optimization period, the assignment lock time and the pickup time window length.

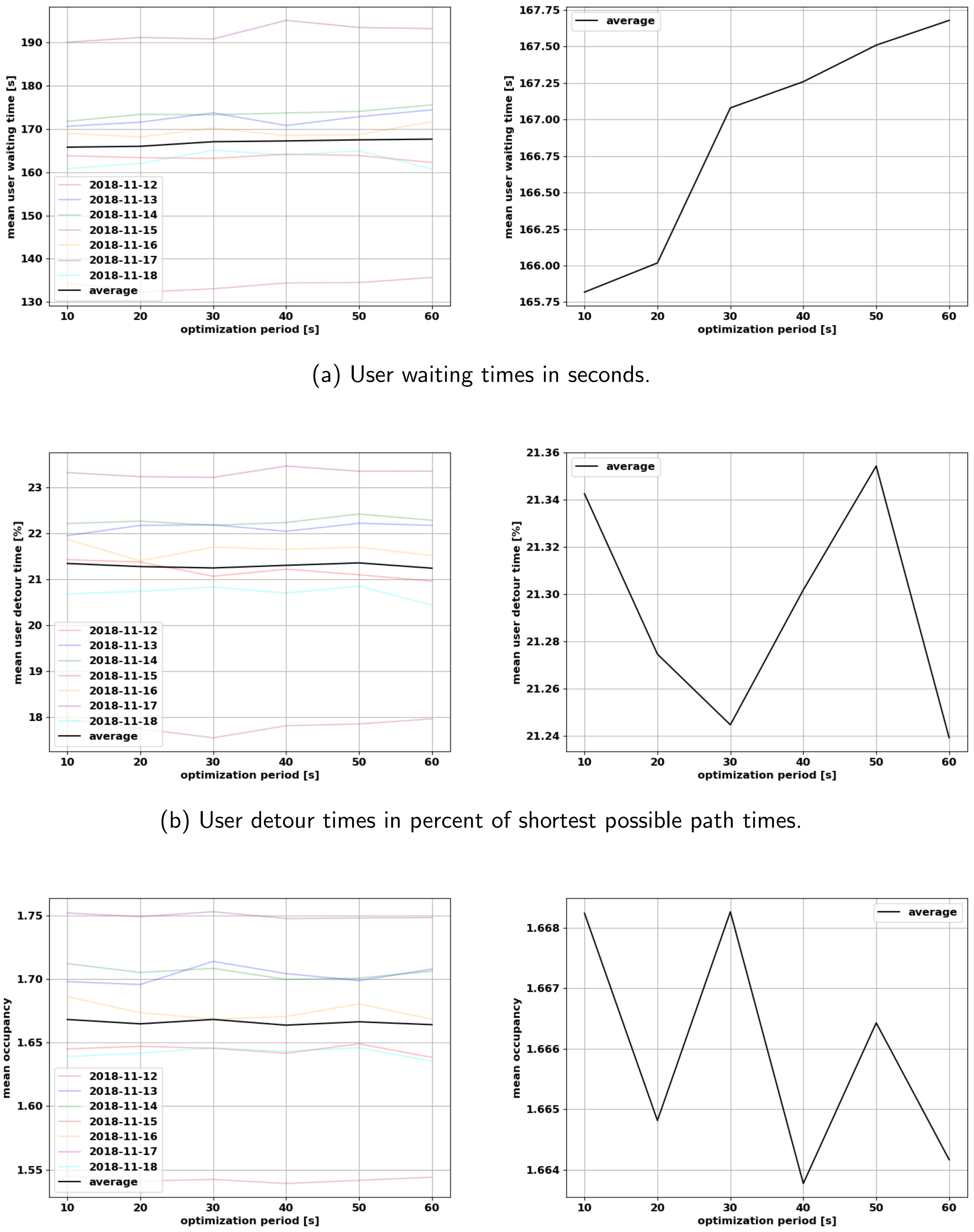


(a) Distance driven emptily in percent of total driven distance.

(b) Added distances driven in percent of total distance of direct user trips.

(c) Percentage of requests served.

Figure A.6: Parameter sensitivity analyses in simulations of the 2-step service model with 300 vehicles in the ride pooling use case with values of the optimization period t_o between 10 s and 60 s. Left: values for all simulation dates and the average value. Right: average value in detail.

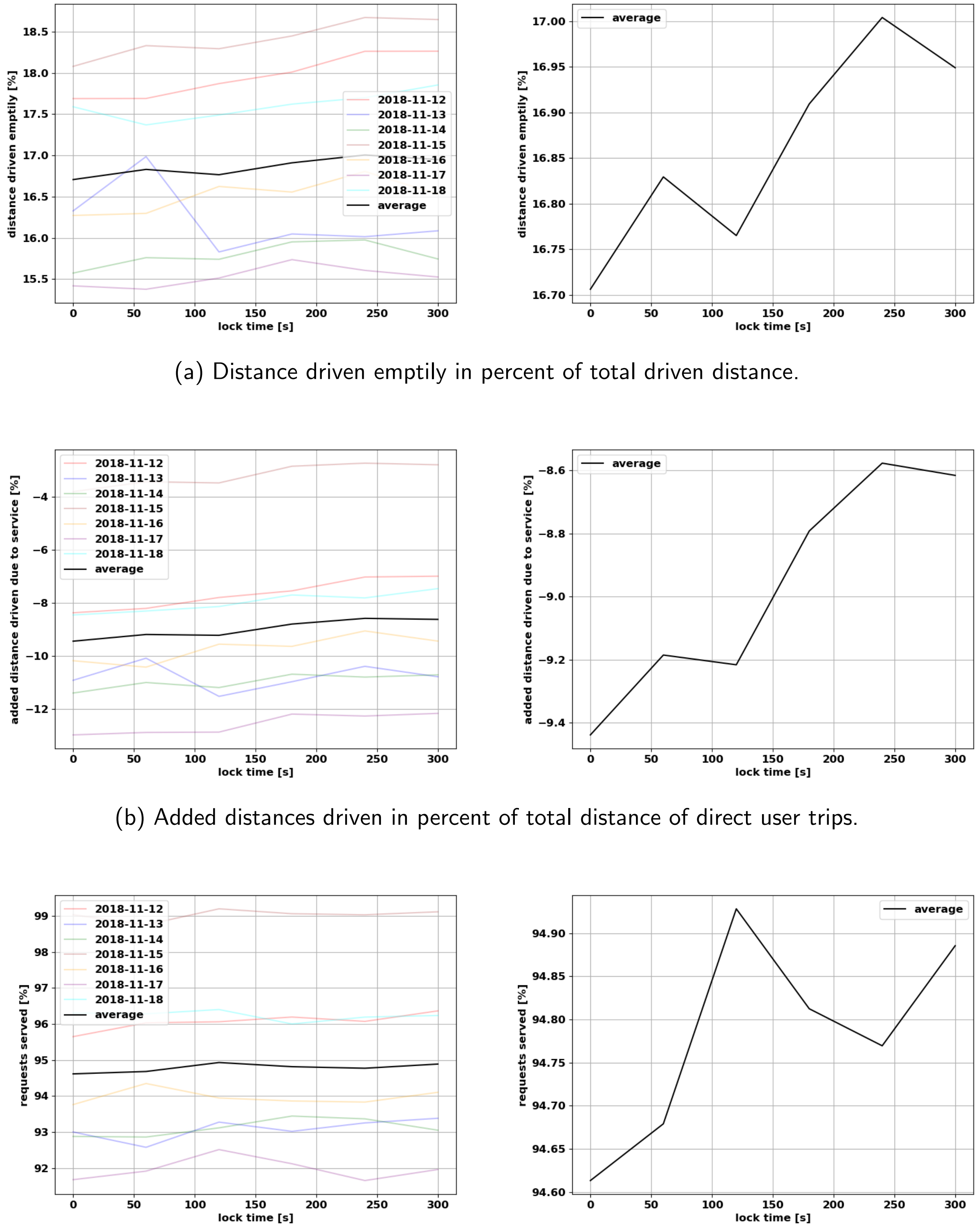


(a) User waiting times in seconds.

(b) User detour times in percent of shortest possible path times.

(c) Occupancy in average passengers per vehicle.

Figure A.7: Parameter sensitivity analyses in simulations of the 2-step service model with 300 vehicles in the ride pooling use case with values of the optimization period t_o between 10 s and 60 s. Left: values for all simulation dates and the average value. Right: average value in detail.

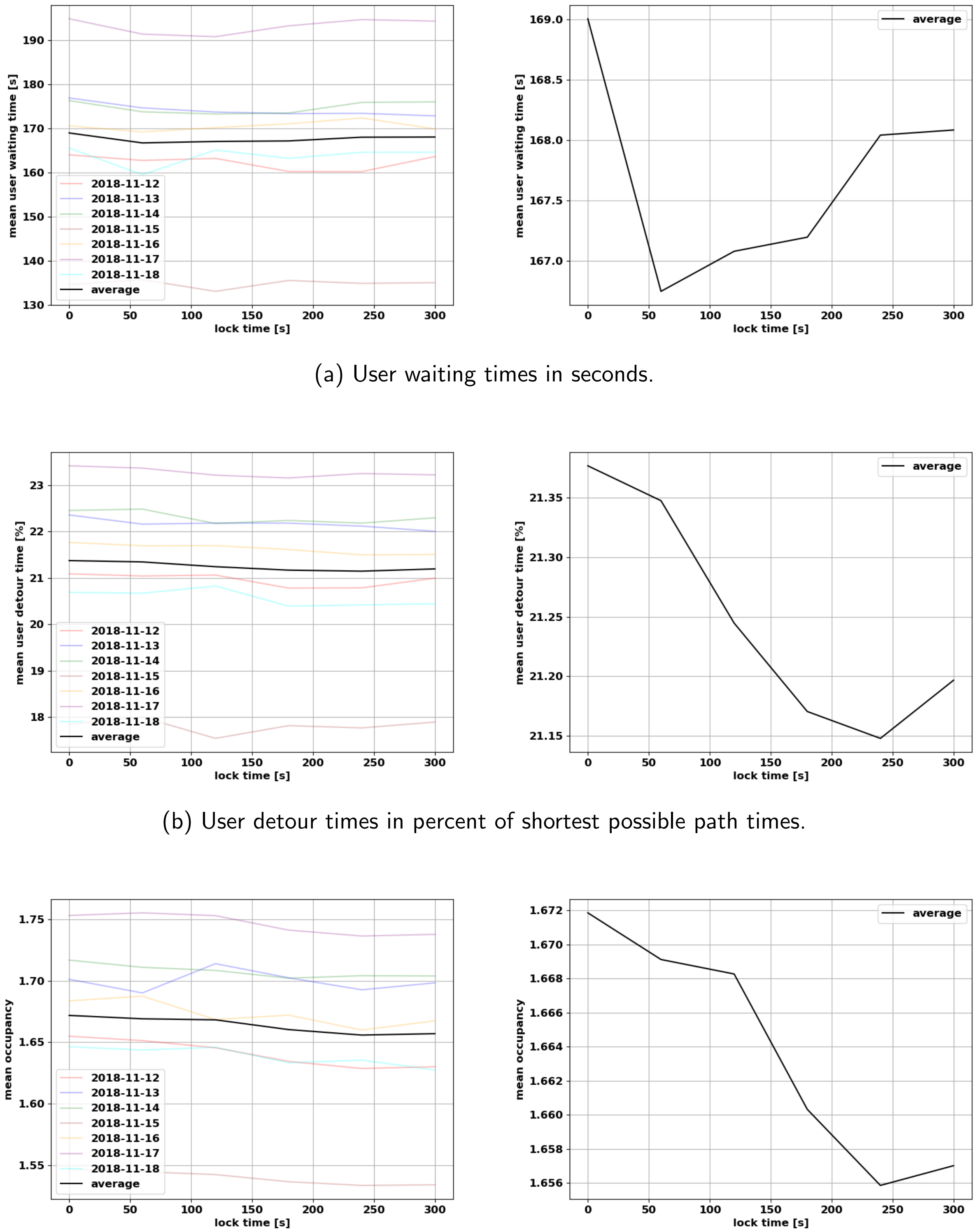


(a) Distance driven emptily in percent of total driven distance.

(b) Added distances driven in percent of total distance of direct user trips.

(c) Percentage of requests served.

Figure A.8: Parameter sensitivity analyses in simulations of the 2-step service model with 300 vehicles in the ride pooling use case with values of assignment lock time t_{lock} between 0 min and 5 min. Left: values for all simulation dates and the average value. Right: average value in detail.

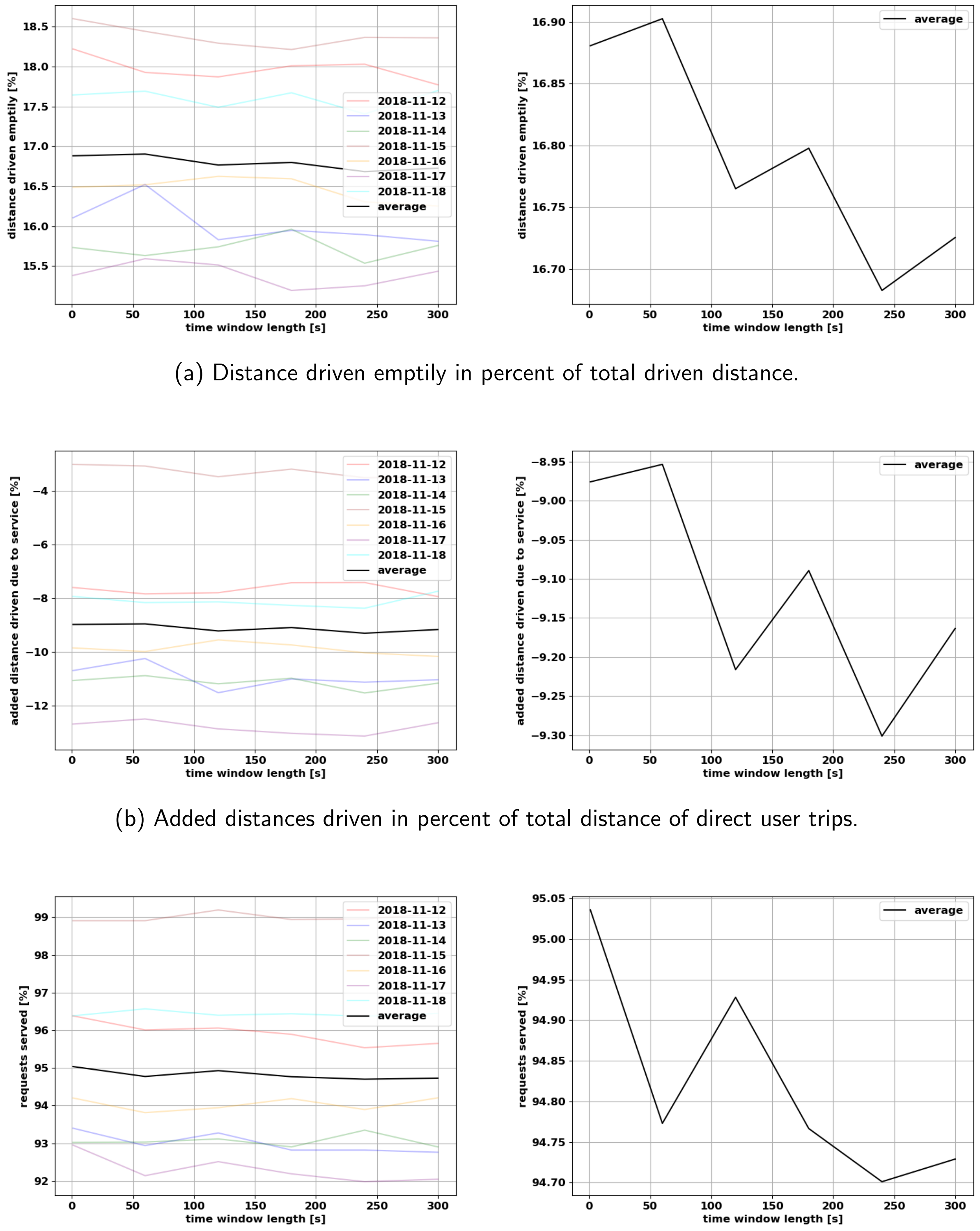


(a) User waiting times in seconds.

(b) User detour times in percent of shortest possible path times.

(c) Occupancy in average passengers per vehicle.

Figure A.9: Parameter sensitivity analyses in simulations of the 2-step service model with 300 vehicles in the ride pooling use case with values of assignment lock time t_{lock} between 0 min and 5 min. Left: values for all simulation dates and the average value. Right: average value in detail.

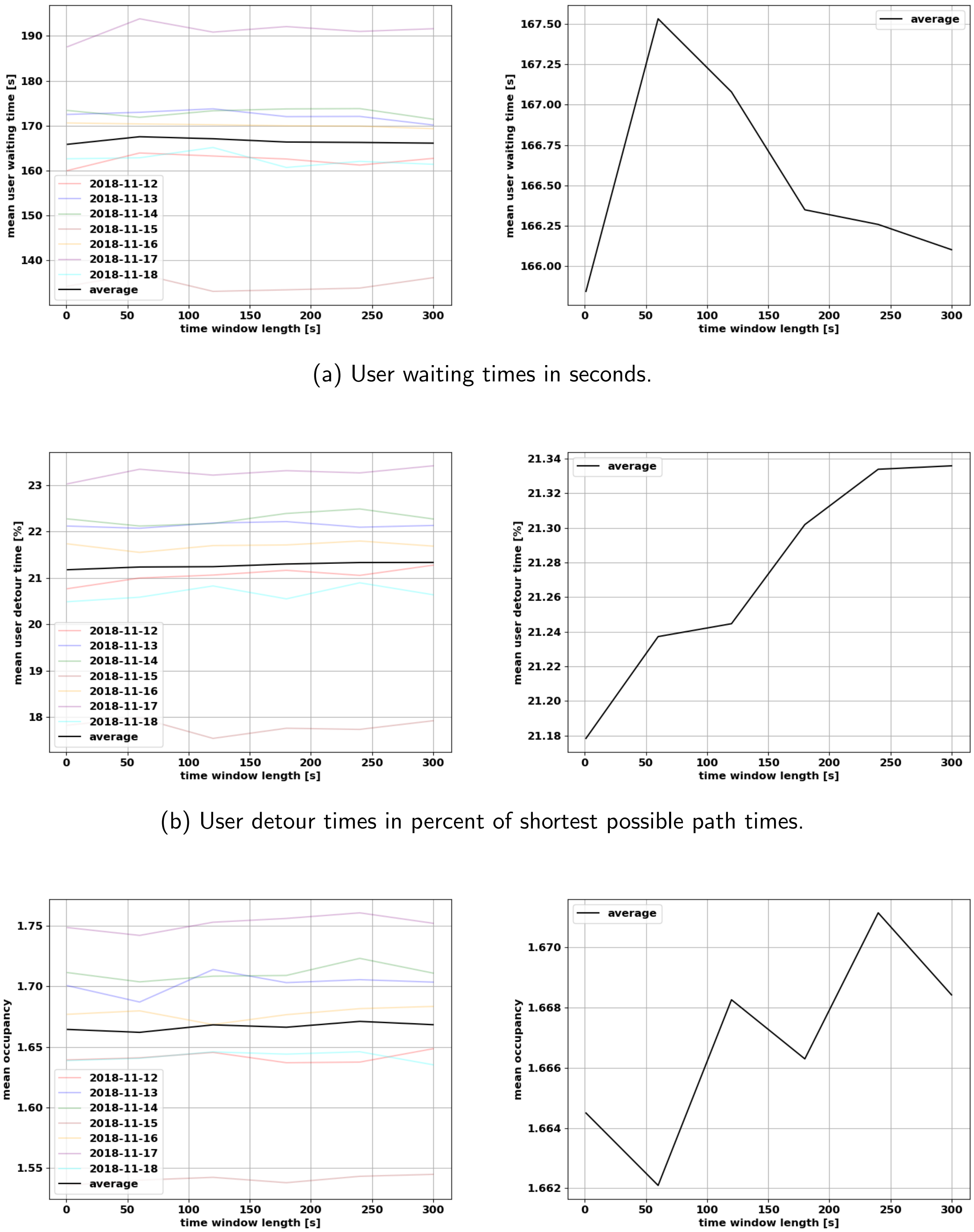


(a) Distance driven emptily in percent of total driven distance.

(b) Added distances driven in percent of total distance of direct user trips.

(c) Percentage of requests served.

Figure A.10: Parameter sensitivity analyses in simulations of the 2-step service model with 300 vehicles in the ride pooling use case with values of pickup time window lengths t_{twl} between 0 min and 5 min. Left: values for all simulation dates and the average value. Right: average value in detail.



(a) User waiting times in seconds.

(b) User detour times in percent of shortest possible path times.

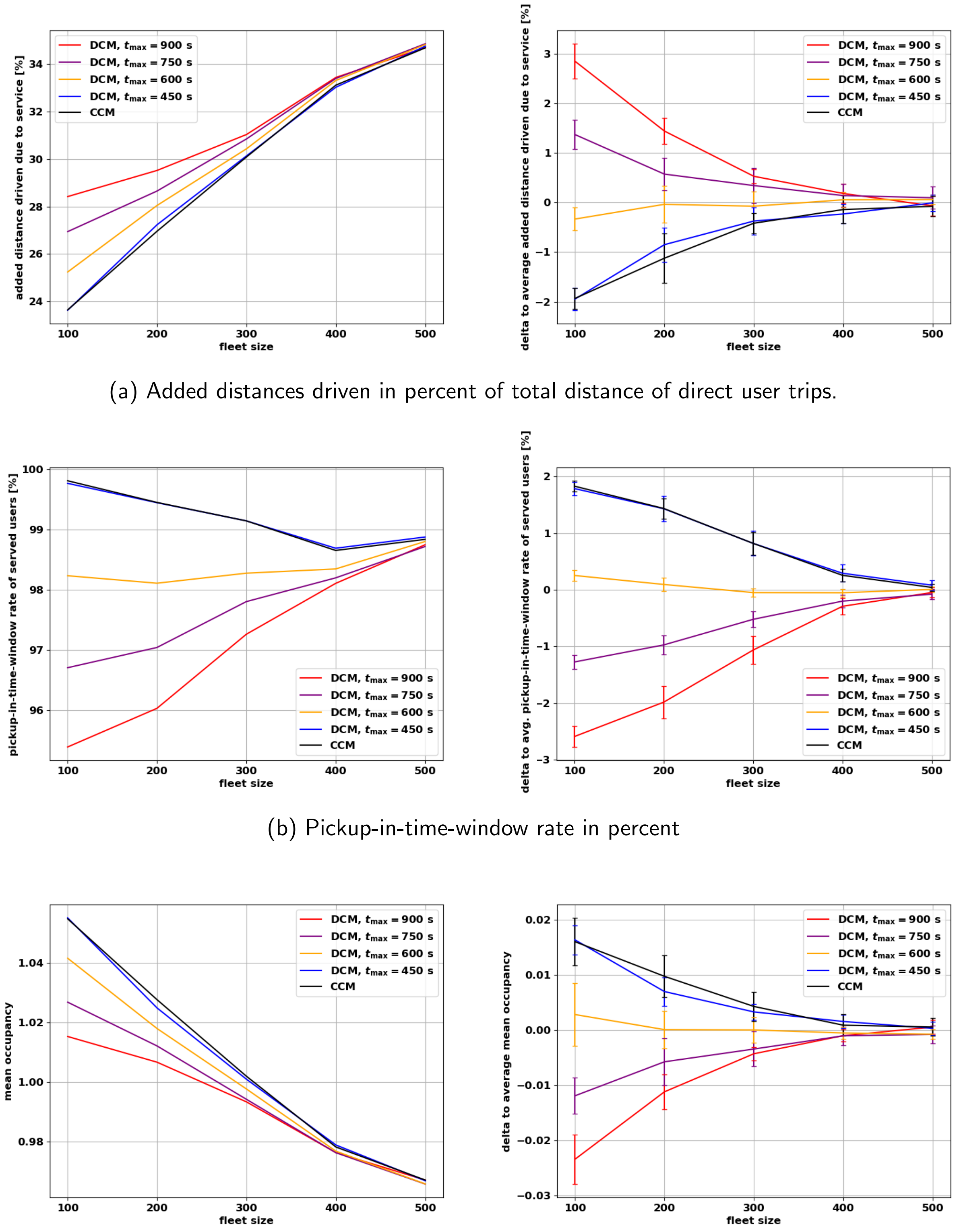
(c) Occupancy in average passengers per vehicle.

Figure A.11: Parameter sensitivity analyses in simulations of the 2-step service model with 300 vehicles in the ride pooling use case with values of pickup time window lengths t_{twl} between 0 min and 5 min. Left: values for all simulation dates and the average value. Right: average value in detail.

To Chapter 6: Diffusion Customer Model

The evaluation of the diffusion customer models in comparison to a conventional customer model shows an overestimation of performance capabilities of ride hailing and ride pooling service models if service operators assume less realistic (conventional) customer models. In addition to the findings presented in Chapter 6, the following figures present further evidence of that. Independent of the use case, KPIs measured in scenarios with the conventional customer model tend to be better than in the corresponding scenarios with diffusion customer models. In Figure A.12, this can be observed for the ride hailing use case. Figure A.15 shows additional KPIs in the ride pooling use case.

The decision-making process in the diffusion customer model introduced in Chapter 6 is based upon the qualities of service offers made by the service operator. These offer qualities are independent of the customer model parameters analyzed in the parameter sensitivity analyses of that chapter, which can be seen in Figures A.13 and A.14 for ride hailing services and in Figures A.16 and A.17 for the ride pooling use case.



(a) Added distances driven in percent of total distance of direct user trips.

(b) Pickup-in-time-window rate in percent

(c) Occupancy in average passengers per vehicle.

Figure A.12: Key performance indicators in simulations with various diffusion customer model parameters in the ride hailing use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

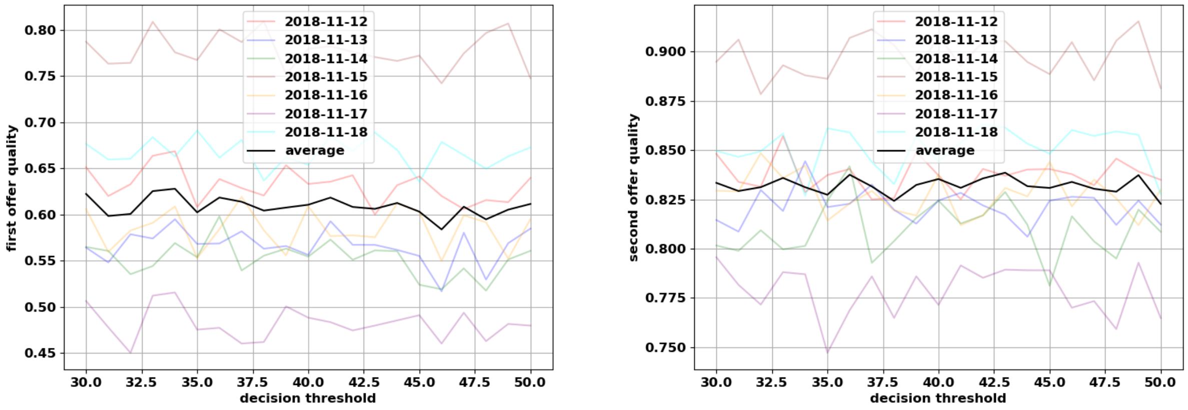


Figure A.13: Offer qualities in simulations with decision thresholds between 30 and 50 in the ride hailing use case. Values for all simulation dates and the average values. Left: related to first offers. Right: related to second offers.

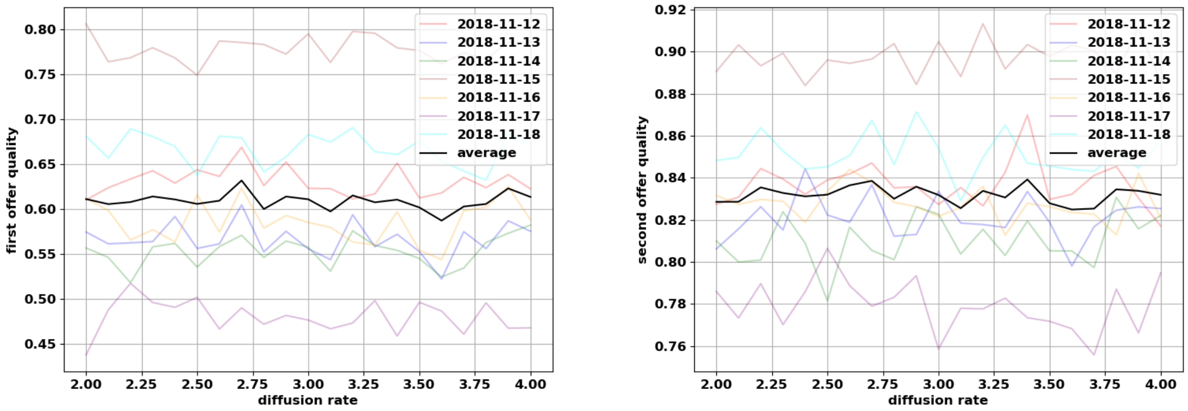
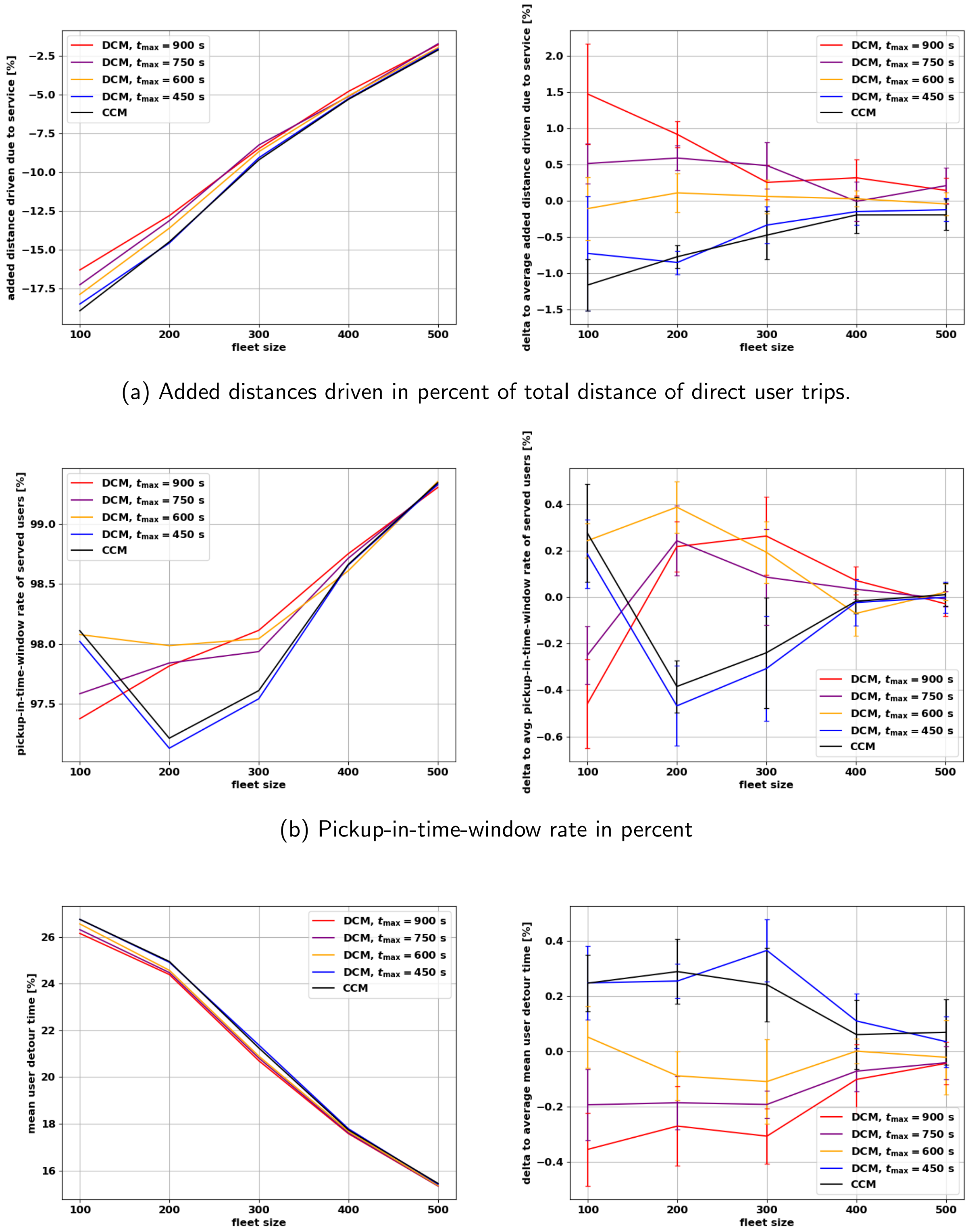


Figure A.14: Offer qualities in simulations with diffusion rates between 2.0 and 4.0 in the ride hailing use case. Values for all simulation dates and the average values. Left: related to first offers. Right: related to second offers.



(a) Added distances driven in percent of total distance of direct user trips.

(b) Pickup-in-time-window rate in percent

(c) User detour times in percent of shortest possible path times.

Figure A.15: Key performance indicators in simulations with various diffusion customer model parameters in the ride pooling use case with fleet sizes from 100 to 500 vehicles. Left: total mean values. Right: delta to average values.

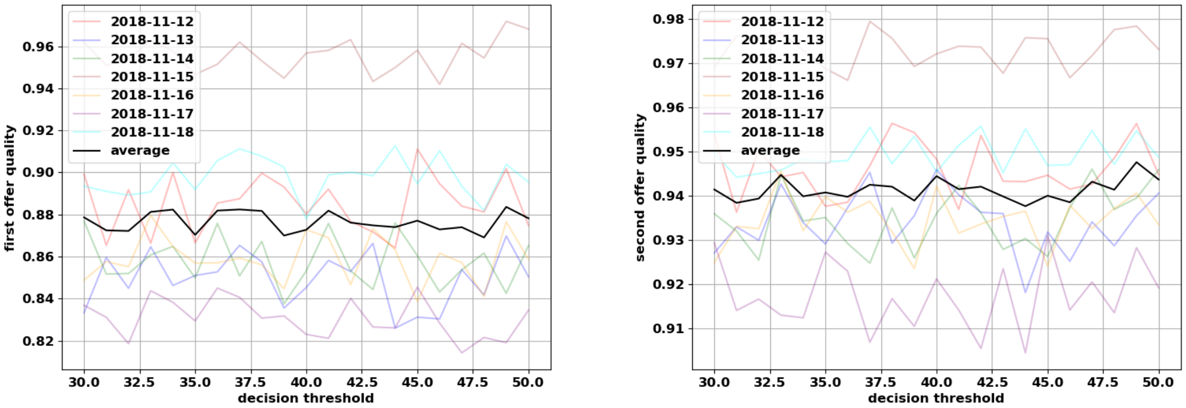


Figure A.16: Offer qualities in simulations with decision thresholds between 30 and 50 in the ride pooling use case. Values for all simulation dates and the average values. Left: related to first offers. Right: related to second offers.

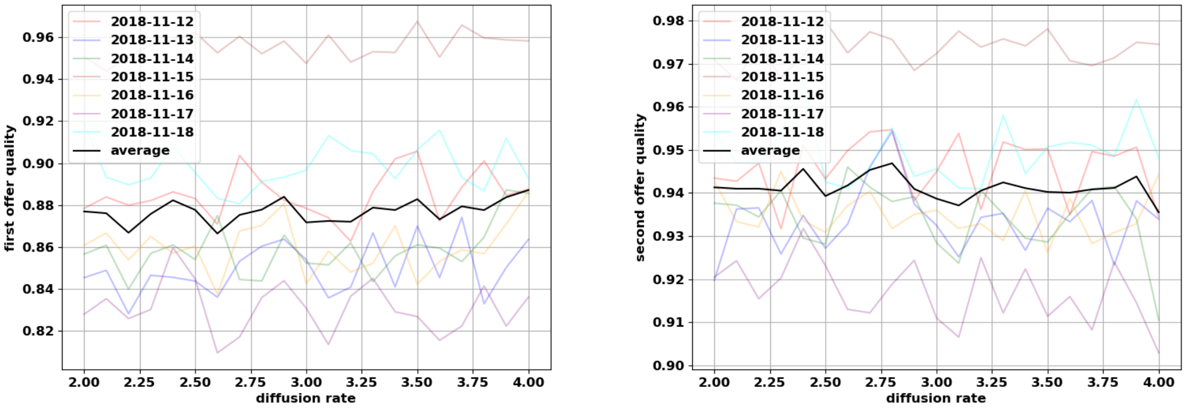


Figure A.17: Offer qualities in simulations with diffusion rates between 2.0 and 4.0 in the ride pooling use case. Values for all simulation dates and the average values. Left: related to first offers. Right: related to second offers.