A Novel Vehicle Repositioning Strategy for Free-Floating Vehicle Sharing Systems with Heterogeneous Fleets using Stochastic Optimization

Bachelor Thesis

Author: Moritz Gottschling (Student ID: 7350270)
Supervisor: Muhammed Demircan

Department of Information Systems for Sustainable Society
Faculty of Management, Economics and Social Sciences
University of Cologne

July 16, 2021

Eidesstattliche Versicherung

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne die Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten und nicht veröffentlichten Schriften entnommen wurden, sind als solche kenntlich gemacht. Die Arbeit ist in gleicher oder ähnlicher Form oder auszugsweise im Rahmen einer anderen Prüfung noch nicht vorgelegt worden. Ich versichere, dass die eingereichte elektronische Fassung der eingereichten Druckfassung vollständig entspricht.

Die Strafbarkeit einer falschen eidesstattlichen Versicherung ist mir bekannt, namentlich die Strafandrohung gemäß § 156 StGB bis zu drei Jahren Freiheitsstrafe oder Geldstrafe bei vorsätzlicher Begehung der Tat bzw. gemäß § 161 Abs. 1 StGB bis zu einem Jahr Freiheitsstrafe oder Geldstrafe bei fahrlässiger Begehung.

Moritz Gottschling

Köln, den 02.02.2021

Abstract

In this paper, we develop a novel relocation strategy for vehicle sharing systems, which considers multiple vehicle types and possible substitutional effects between them. To accomplish this, we first model the vehicle sharing system, using stochastic optimization. We then conduct a data analysis on a dataset consisting of trips from vehicle sharing providers in cologne to examine whether our model's assumptions are reasonable. We apply our model to said dataset and benchmark it against a model that does not consider substitutional effects between different vehicle types. We find that considering substitutional effects when planning relocations can increase profitability. However, the additional complexity that comes with multiple vehicle types drastically increased the computational costs. To use our model efficiently, it needs further optimization.

Contents

List of Figures

List of Tables

1 Introduction

Free-Floating-Vehicle-Sharing-Systems (FFVSS), like Tier, Lime, NextBike and ShareNow, have become increasingly popular in cities. These systems allow users to rent nearby vehicles at any time and return it anywhere in the service region. They decrease the need for private cars, bicycles, mopeds and other vehicles, which leads to lower utilization of parking lots. By providing vehicle types that are not powered by fossil fuels, like bicycles or kick-scooters, FFVSS can decrease CO2 emissions (Zhang & Mi, 2018). Unlike Station-Based-Vehicle-Sharing-Systems, users of free-floating systems can park their rented vehicle almost anywhere. This also enables users to use other modes of transport subsequently. Therefore, free-floating systems provide their users with high flexibility, which leads to their increased popularity (Chen et al., 2020).

Users of these systems are only willing to walk a specific distance to reach the desired vehicle (Wang & Yan, 2016). Therefore, to increase usage rates, even more, the vehicles must be in an acceptable walking range. However, there often is an imbalance between demand and availability of vehicles in these systems (Reiss & Bogenberger, 2015; Schmöller et al., 2015), which prevents users from using them. This imbalance arises from the fact that system users can rent a vehicle whenever they want, for as long as they want, and then drop it off wherever they want. To respond to this imbalance, operators of FFVSS relocate vehicles from regions with low demand and high availability to regions with high demand and low availability. There are several researchers who develop relocation strategies that address the imbalance problem. For most of the time, FFVSS had vehicle fleets with only one type of transport, primarily cars or bicycles. Recently FFVSS operators started to deploy different types of vehicles in one fleet (e.g. Tier with mopeds and e-scooters). Relocation strategies for FFVSS with one vehicle type can still be used, but they do not consider possible substitutional effects between the demand for the different vehicle types. We expect that, when taking different vehicle types into account, the profitability can be improved. Therefore, this paper aims to develop and test a new strategy to relocate vehicles in FFVSS, which takes the substitutional effects of different vehicle types into account.

The rest of this paper is structured as follows. Firstly, we review the current state of the literature of vehicle relocation strategies in Section 2. After that, we explain the theoretical foundations that are required to understand our model in Section 3. Then we formulate our model as stochastic program in Section 4. Next, we examine a real-world dataset and show how to use our model on it in Section 5. Afterwards, we examine the results of our model in Section 6. Finally, we analyze in Section 7 how the results of our paper contribute to the current

state of the literature.

2 Literature Review

There has been various research on relocation strategies for FFVSS with a single vehicle type. However, this field of study is still young. Therefore we concentrate on the period from 2015 to today (2021) in our literature review.

He et al. (2019) developed a stochastic dynamic program, which minimizes relocation costs and penalties caused by unmet demand. They also described an optimal strategy for a system with two regions. For systems with more than two regions, they introduced an approximation algorithm based on distributionally robust optimization. To accomplish this, they first presented a model that only considers one period and ignores any future costs. Extending this model into multiple periods makes the underlying problem NP-hard. To address this issue, they introduced the linear decision rule (LDR). They later replaced the LDR with the enhanced linear decision rule to further improve performance.

Benjaafar et al. (2018) modeled the problem in a similar way to He et al. (2019). They also developed a stochastic dynamic program, which minimizes operational costs. In addition to the optimal strategy for a two-region system described by He et al. (2019), they also formulated an optimal strategy for an n-region system. In this strategy, they described a well-defined region in which no relocation should take place. In any other region, relocations should take place to the boundary of the no relocation region. They solved their model with a cutting plane-based approximate dynamic programming algorithm. They also proved that their algorithm approximates the true value function of the problem, and therefore their model leads to more accurate results than the robust approach by He et al. (2019).

Lu et al. (2017) developed a model for both reservation-based and free-floating systems, taking into account parking lot costs and free-floating parking permits, respectively. They formulated the model as a two-stage stochastic integer programming model. In the first stage, the model determines a starting allocation of the vehicles and the amount of parking lot tickets and free-floating parking permits, minimizing the related costs. In the second stage, a spatial-temporal network for a finite set of generated demand samples is constructed. This network models one-way trips, round trips, relocations, and idle vehicles as flows. They intend their model to be applied daily, providing an initial allocation for each day. However, the model does not consider the relocation costs that occur when providing this initial allocation. In contrast to the model of (He et al., 2019), this model allows for denying demand, which is not realistic in the case of free-floating systems. To counterfeit the high computational cost due to the underlying stochastic program, the authors implemented Benders Decompo-

sition, as first seen in (Benders, 1962), and further improve the algorithm with mixed-integer rounding. Their improvement proves to be effective as it decreases computing time by a factor of 20 to 100.

Most studies, including He et al. (2019), Lu et al. (2018), Benjaafar et al. (2018), consider relocation costs to be proportional to the number of relocated vehicles and the distance of the relocation. However, this can be inaccurate when dealing with smaller vehicles, like bicycles or scooters. Because these types of vehicles are small, it is possible to relocate multiple vehicles at once by one operator, which makes the cost per vehicle marginal, while the setup cost is high (Zhao et al., 2020). Zhao et al. (2020) show that an optimal strategy as defined by He et al. (2019) and Benjaafar et al. (2018) changes, when setup costs are taken into account.

Due to the high dimensionality of problems in bicycle relocation, their solutions are often computationally intractable. Li et al. (2018) introduced an Inter-Independent Inner-Balance Clustering algorithm, which groups the regions of interest into clusters that can be examined separately. This clustering reduces the problem size and can therefore increase performance. They also developed a Spatio-Temporal Reinforcement Learning model to approximate an optimal relocation strategy.

Tang et al. (2020) developed a model for vehicle relocation with a robustness optimization framework. They focused on accounting for as much uncertainty as possible. Their model considers the trips' demand, destination and, travel time uncertainty as well as uncertainty in the duration of relocations. While most of the named papers maximize profitability (Lu et al., 2017; He et al., 2019), flow (Shu et al., 2013) or minimize unmet demand (Jian et al., 2016; Li et al., 2018), Tang et al. (2020) maximize the probability of meeting certain operational constraints, such as the vehicle capacity of regions, the maximum cost budget for relocations or keeping unmet demand under a certain threshold. The fact that the model maximizes the probability of meeting constraints has the advantage that the solution is more likely to be feasible (and therefore realistic) when the demand deviates from the considered scenarios.

Shu et al. (2013) modeled a station-based bicycle-sharing system with a network flow model. Their model estimates the maximum flow capacity given an initial allocation of bicycles and station capacities. They also extended their model to measure the effectiveness of bicycle redistribution. However, in contrast to the previously named papers, their model only considers relocation at the end of each day and not multiple times per day.

Relocating vehicles is not the only option to balance the demand and availability. There is also research on creating incentives for users to prevent or revert

imbalances (Fricker & Gast, 2012).

3 Theoretical Foundations

In order to understand our model, some knowledge in the domain of Linear Programming and Stochastic Programming is required. Therefore we introduce the basic concepts of Linear Programming and Stochastic Programming in Section 3.1 and Section 3.2, respectively.

3.1 Linear Programming

Linear programs are used to find the optimal value for a mathematical problem. In our paper, we will use a linear program to model a FFVSS. Linear programs consist of constraints and an objective function. The goal in Linear Programming is to optimize (minimize/maximize) the objective function by changing the values of decision variables while fulfilling the set of constraints. We can write the decision variables in a linear program as a vector $x \in \mathbb{R}^n$. In Linear Programming, the objective function and the constraints are both linear. Therefore it is possible to write the objective function as $c^\intercal x$ with a fixed $c \in \mathbb{R}^n$. Similarly the constraints can be written as $Ax \leq b$ with $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$, also fixed. We call all $x \in \mathbb{R}^n$, for which $Ax \leq b$ holds, solutions. The solution for which the objective function is optimal (maximized/minimized) is called the optimal solution. The canonical form of a linear program in matrix notation is written as

$$\begin{aligned} \max_x \quad & c^\intercal x \\ s.t. \quad & Ax \leq b \\ & x \geq \mathbf{0} \end{aligned} \tag{1}$$

with $\mathbf{0} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix} \in \mathbb{R}^n$. For our model we will not use the matrix notation but the following notation, which is equivalent:

$$\begin{aligned} \max_x \quad & c_1x_1 + c_2x_2 + ... + c_nx_n \\ s.t. \quad & a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n \leq b_1 \\ & a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_n \leq b_2 \\ & \vdots \\ & a_{m1}x_1 + a_{m2}x_2 + ... + a_{mn}x_n \leq b_m \\ & x_1 \geq 0 \\ & \vdots \\ & x_n \geq 0 \end{aligned} \tag{2}$$

or written with quantifiers and summations:

$$\begin{aligned} \max_{x} \quad & \sum_{i=1}^{n} c_i x_i \\ s.t. \quad & \sum_{i=1}^{n} a_{ji} x_i \leq b_j \quad \forall\, 1 \leq j \leq m \\ & x_i \geq 0 \qquad\qquad \forall\, 1 \leq i \leq n \end{aligned} \tag{3}$$

Geometrically, each constraint defines a half-space, with the intersect of these half-spaces being a polytope. All points inside the polytope fulfil all the constraints and are therefore valid solutions. Finding the optimal solution is equivalent to finding the point in the polytope, which maximizes (minimizes) the objective function.

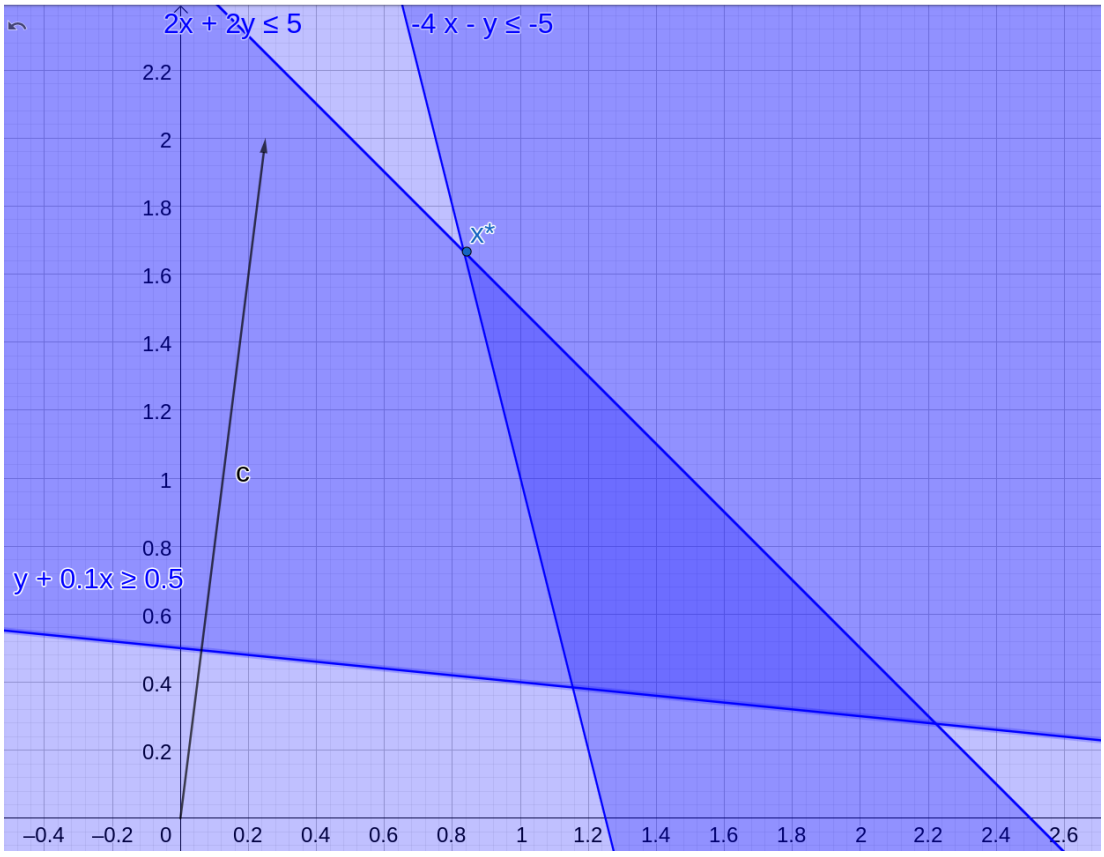


Figure 1: Two-Dimensional Linear Program

Figure 1 shows the visualization of the following linear program:

$$\begin{aligned} \max_{x,y} \quad & 0.25x + 2y \\ s.t. \quad & 2x + 2y \leq 5 \\ & -4 - y \leq -5 \\ & y + 0.1x \geq 0.5 \end{aligned} \tag{4}$$

The objective function $c^\intercal \binom{x}{y}$ with $c = \binom{0.25}{2}$ is displayed as a vector. The value of the objective function increases into the direction of the vector c. We can easily see that the optimal solution of the linear program above is x^*.

How fast we can solve a linear program depends on its size and whether its decision variables are real numbers or integers. Linear programs with only real numbers as decision variables are proven P-complete, which means that there exists an algorithm that solves linear programs in polynomial time (the ellipsoid method). If, however, a linear program has decision variables that are integers, then the linear program becomes NP-hard, which in turn means that if $P \neq NP$ (which is believed to be true by a majority of researchers), it cannot be solved in polynomial time. This does not mean that these so-called integer linear programs are impossible to solve, but solving them can take very long when the problems get large.

3.2 Stochastic Programming

Problems in stochastic programming involve uncertainty, which means that some model parameters are unknown and follow a probability distribution. They are, therefore, random variables. We differentiate between two kinds of probability distributions: continuous and discrete. Continuous probability distributions are often more realistic but lead to very hard or even impossible to solve stochastic programs. Therefore if we want to solve a stochastic program where some random variables follow a continuous probability distribution, we have to approximate that distribution with a discrete one. The advantage of stochastic programs with random variables that follow a discrete distribution is that we can transform them into their deterministic equivalent. This deterministic equivalent can be expressed as and solved like a linear program.

In a stochastic program with a discrete probability distribution, we call the realizations of the random variables scenarios. Say we want to model a vehicle sharing system in only one region. If we would assume that the demand is deterministic (not stochastic), then we could describe the demand with one parameter $d \in \mathbb{N}_0$. A sample value would be $d = 5$. It is, however, more realistic that the demand is uncertain. In this case, we can model the system as a stochastic program and describe the demand with a random variable. Then we can write the demand as $d_i \in \mathbb{N}_0$, where i is the scenario index. If we assume that d could take on three different values, with the same probability, we can now write d like this:

d_1	2
d_2	1
d_3	3

Our model will span multiple periods, which means that each random variable will take on multiple values. We call a stochastic variable that takes on multiple values over time a stochastic process. One scenario is now no longer represented by a single value but by a series of values. The following table represents three different scenarios over three periods, with t indicating the period and s indicating the scenario index.

d_{ts}	$s=1$	$s=2$	$s=3$
$t=1$	1	1	1
$t=2$	2	2	3
$t=3$	1	2	2

Table 1: Example Scenarios

Notice that in the first period, all scenarios have the same value. If any decision variables represent decisions that have to be made in the first period and cannot be changed after, then these decisions have to be consistent. That means that if the parameters of two scenarios are equal, then the decision variables have to be chosen equally as well. In stochastic programming, this is ensured by non-anticipativity constraints. These constraints force decisions made in the same period and with the same knowledge to be the same. Say our in our example, the model would include relocations in each period that happen after the demand is realized, denoted by r_{ts}. In the first period the non-anticipativity constraint would be: $r_{11} = r_{12} = r_{13}$. It is usual to visualize the scenarios in a scenario tree where each node represents a possible state. Each node, which is not in the last period, has successors, and the branches to those nodes can be thought of as the realizations of the random variables. One scenario is the path from the root node (first period) to any of the leave nodes (last period). Therefore, the decisions made in each node have to consistent for all scenarios that include that node. Figure 2 shows the a scenario tree with the values from Table 1.

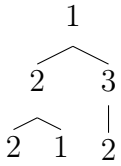


Figure 2: Uncertain Demand Represented In A Scenario Tree

If we try to approximate a continuous probability distribution properly, the number of scenarios must be high. Unfortunately, increasing the number of sce-

narios also increases the size of the linear program that is the deterministic equivalent of the stochastic program. Therefore stochastic programs with a high number of scenarios often become computationally intractable. To address this issue, various decomposition techniques can be used to decrease solving time by exploiting the structure of stochastic programs.

For a comprehensive overview of stochastic programming we recommend Conejo et al. (2010) and Birge and Louveaux (2011).

4 Model Formulation

In this paper we model the vehicle relocation problem as a stochastic integer linear program. We start by explaining the model on a high level and introduce all relevant indices in Section 4.1. Then we will formulate the models constraints and objective function in Section 4.2 and 4.3 respectively, neglecting the stochastic nature of the model. We will then convert the model to a stochastic program in Section 4.4. In Section 4.5 we show the complete Stochastic Integer Linear program, which we further extent in Section 4.6.

4.1 Model Description

In this section, we will explain how our model depicts a real vehicle sharing system. To do that, we first have to introduce all relevant indices.

We split the service region in which the FFVSS operates into N regions. We denote the set of these N regions as $I = \{i_0, i_1, \ldots, i_{N-1}\}$.

Let $T \in \mathbb{N}_0$ be the number of periods and $[T] \coloneqq \{0, 1, ..., T-1\}$ the set of all periods. All periods $t \in [T]$ have equal duration. We assume all trips and relocations to start and end in a single period, which means that trip and relocation durations have to be shorter than the period duration.

To depict vehicle types in the fleet, let M be the number of different vehicle types and $[M] \coloneqq \{0, 1, ..., M-1\}$ the set of all vehicle types. We assume that customers will use certain vehicle types as a substitute for others. Substitutions seem plausible as it is likely that customers will search for an alternative mode of transport when no vehicle of the desired type is nearby. We further assume that whether this substitution can take place depends on the type of vehicle substituted. To model which vehicle type can substitute another, we use the ordering in the set $[M]$. Let $m \in [M]$ be a vehicle type. Customers use this vehicle type as a substitution for all vehicle types $m' < m$. We intend vehicle types of higher order to be vehicle types that are more flexible, meaning more comfortable, suitable for longer trips and capable of transporting multiple passengers. Therefore it seems reasonable that these vehicle types can substitute any other. Vehicle types of lower order should be vehicle types that are suitable for short trips. Therefore, it also seems plausible that they cannot substitute vehicle types of higher order. We will investigate how reasonable these assumptions are in our data analysis in Section 5.1.

At the beginning of a period t, the vehicles are distributed across the service region. x_{itm} depicts the number of vehicles of type $m \in [M]$ in region $i \in I$. Note that for $t = 0$ all $x_{itm} \forall i \in I, m \in [M]$ are given constants and represent the initial allocation of the vehicle fleet. The first event that occurs in each period is

the inbound demand. We depict demand as a number of users wanting to travel from a region $i \in N$ to another region $j \in N$ with a vehicle of type $m \in [M]$. The demand is represented as $d_{ijtm} \in \mathbb{N}_0$. Depending on the availability of vehicles x_{itm}, the demand is realized, resulting in trips. Let $y_{ijtm} \in \mathbb{N}_0$ be the number of trips from region i to region j in period t with vehicle type m. Accordingly y_{ijtm} with $i \neq j$ is the number of one-way trips from region i to region j and y_{iitm} is the number of round trips in region i. We depict unfulfilled demand by u_{ijtm}. More precisely u_{ijtm} stands for demand for vehicles of type $m' \leq m$, that could not be fulfilled by vehicles of type $m' \leq m$. This unfulfilled demand can still be fulfilled by any vehicle type $m' > m$. We depict the sum of all unfulfilled demand originating from region i as U_{itm}. It is not allowed for unfulfilled demand U_{itm} to occur if there are remaining vehicles in the region. If not all vehicles in a region are used for trips (the demand is smaller than the number of vehicles), then we depict the idle vehicles in a region by v_{itm}. After all trips in a period ended, we depict the number of vehicles in a region after demand realization with $\overline{x}_{itm}$. Then the operator can relocate vehicles from one region into another. We depict these relocations with r_{ijtm}. After the relocations the next period $t + 1$ begins. Even though individual trip distances may differ, we assume that all trips starting in region i ending in region j have the same trip distance.

4.2 Constraints

In this Section, we will formulate the constraints of an integer linear program modeling a vehicle sharing system as described in 4.1.

To make sure that the number of trips originating from region i is equal to the demand and the unfilled demand, we formulate constraint 5. Note that constraint 5 is only defined for the vehicle type $m = 0$, as the constraint does not contain any substitutions and only the first vehicle type cannot be used as a substitute for any other vehicle type.

$$y_{ijtm} = d_{ijtm} - u_{ijtm} \quad \forall i, j \in I, t \in [T-1], m = 0 \tag{5}$$

Constraint 6 is derived from constraint 5, by adding the unfulfilled demand of the next lower vehicle type. Therefore a vehicle of type m can substitute the unfulfilled demand a of vehicle of type $m - 1$. With recursion this ensures that m can substitute all unfulfilled demand of vehicles of class $m' < m$.

$$y_{ijtm} = d_{ijtm} - u_{ijtm} + u_{ijt(m-1)} \quad \forall i, j \in I, t \in [T-1], m \in [M] \setminus \{0\} \tag{6}$$

To ensure that the model does not deny any demand that could be fulfilled we

introduce constraints 7, 8, 9, 10. U^b_{itm} and v^b_{itm} are binary variables that represent whether the region i has any unfulfilled demand of type m or idle vehicles of type m in period t, respectively.

$$U_{itm} = \sum_{j \in I} u_{ijtm} \quad \forall i \in I, t \in [T-1], m \in [M] \tag{7}$$

$$v_{itm} \leq C_m v^b_{itm} \quad \forall i \in I, t \in [T-1], m \in [M] \tag{8}$$

$$U_{itm} \leq M U^b_{itm} \quad \forall i \in I, t \in [T-1], m \in [M] \tag{9}$$

$$U^b_{itm} + v^b_{itm} \leq 1 \quad \forall i \in I, t \in [T-1], m \in [M] \tag{10}$$

M is a large constant (larger than any possible value of unfulfilled demand) and C_m is the total number of vehicles of type m in the fleet.

Constraint 11 ensures that the number of vehicles in a region before demand realization is equal to the number of idle vehicles in that region and the number of trips starting in that region.

$$x_{itm} = v_{itm} + \sum_{j \in I} y_{ijtm} \quad \forall i \in I, t \in [T-1], m \in [M] \tag{11}$$

Similarly, constraint 12 ensures that the number of vehicles in a region after demand realization is equal to the number of idle vehicle in that region and the number of trips ending in that region.

$$\overline{x}_{itm} = v_{itm} + \sum_{j \in I} y_{jitm} \quad \forall i \in I, t \in [T-1], m \in [M] \tag{12}$$

The relocations mark the transition from one period to the next. Constraint 13 describes that transition by ensuring that the vehicles in a region after demand realization in period t and the inbound and outgoing relocations are equal to the number of vehicles before demand realization in the next period $t+1$.

$$x_{i(t+1)m} = \overline{x}_{itms} + \sum_{j \in I \setminus \{i\}} (r_{jitm} - r_{ijtm}) \quad \forall i \in I, t \in [T-1], m \in [M] \tag{13}$$

4.3 Objective Function

To consider costs and profits we define the relocation cost per vehicle of type m from region i to region j as c_{ijm} and the profit generated by each trip with a vehicle of type m from region i to region j as p_{ijm}. Costs and profits can be specified by the operator. One should note that c_{iim} is the cost of a vehicle of

type m remaining in region i. It can be seen as the maintenance or parking cost, however, it can be set to zero as well.

Our model will maximize the overall profit, therefore we formulate the objective function as follows:

$$\sum_{m\in[M]}\sum_{t\in[T-1]}\left(\sum_{i,j\in I}(p_{ijm}y_{ijtm}) - \sum_{i,j\in I, i\neq j}(c_{ijm}r_{ijtm}) - \sum_{i\in I}(c_{iim}v_{itm})\right) \tag{14}$$

4.4 Conversion To Stochastic Program

So far, our model is deterministic, which means that we assume that we know the demand. It is, however, challenging to predict vehicle sharing demand. One of the reasons being that the demand and the availability of vehicles are interdependent (Jorge & Correia, 2013). Therefore demand in vehicle sharing systems is highly uncertain. To incorporate this uncertainty into our model, we will adjust the constraints and the objective function to depict a stochastic linear program.

As explained in Section 3.2, to solve stochastic programs, it is (almost) always necessary that the probability distribution is discrete. Therefore our model will use demand in the form of scenarios as an input. This is reasonable as it is possible to approximate a continuous distribution with a discrete one. To solve stochastic programs with discrete probability distributions efficiently they can be reformulated into a linear program (the deterministic equivalent) for which many efficient solvers exist.

The deterministic equivalent of a stochastic program looks similar to simply incorporating the same linear program with (demand) different parameters into one big linear program. Let $S = \{s_0, s_1, ..., s_{max}\}$ be the set of different scenarios. We add the new index s to all of our decision variables and the demand parameter. Now d_{ijtms} represents the number of users wanting to travel from region i to region j at period t with a vehicle of type m in scenario s. To transform the constraints we simply use the new variables and demand parameter with the index s and formulate all constraints for all $s \in S$. Constraint 6 for example is now defined like so:

$$y_{ijtms} = d_{ijtms} - u_{ijtms} \quad \forall i, j \in I, t \in [T-1], m = 0, s \in S \tag{6}$$

The objective function now depicts the expected profit over all scenarios. Let $\pi : S \to [0, 1]$ be the probability function which maps each scenario to its corresponding probability of occurrence, so that $\sum_{s\in S}\pi(s) = 1$. In the new objective function, we will multiply each scenario's objective function with the probability

of the scenario and then sum it all up.

$$\sum_{s\in S}\left(\pi(s)\sum_{m\in[M]}\sum_{t\in[T-1]}\left(\sum_{i,j\in I}(p_{ijm}y_{ijtms})-\sum_{i,j\in I,i\neq j}(c_{ijm}r_{ijtms})-\sum_{i\in I}(c_{iim}v_{itms})\right)\right) \tag{14}$$

We also have to add non-anticipativity constraints to our model. Remember that, non-anticipativity constraints ensure that the decision variables are equal in time periods and scenarios in which the demand is and was the same. For example, let $T = 6$ and $S = \{0, 1\}$. If $d_{ijtms} = d_{ijtms'} \forall i, j \in I, t < 3, m \in [M], s = 0, s' = 1$ or in other words the demand of scenario 1 and scenario 2 is the same for all time periods $t < 3$, then the relocation decisions and trips also have to be the same. To express this behavior mathematically, let $\sigma_t : S \to \mathcal{P}(S)$ with $\sigma_t(s) = \{s' : d_{ijt'ms} = d_{ij'tms'}\ \forall i, j \in I, t' \leq t, \forall m \in [M]\}$ be a function that returns for a given s and t the set of scenarios which have the same demand as s until at least time period t. Constraints 15 and 16 are the non-anticipativity constraints for trips and relocations, respectively.

$$y_{ijtms} = y_{ijtms'} \quad \forall i, j \in I, t \in [T-1], m \in [M], s \in S, s' \in \sigma_t(s) \tag{15}$$

$$r_{ijtms} = r_{ijtms'} \quad \forall i, j \in I, t \in [T-1], m \in [M], s \in S, s' \in \sigma_t(s) \tag{16}$$

We do not need to add non-anticipativity constraint for the other decision variables, as equations 15 and 16 will ensure non-anticipativity for the other decision variables as well. Proofs for this can be read in Appendix D.

4.5 Stochastic Integer Linear Program

An overview of all relevant symbols and their explanation can be seen in Table 2.

Table 2: Symbol Explanations

Symbol	**Explanation**
Sets	
I	Set of regions
$[T]$	Ordered set of periods
$[M]$	Ordered set of vehicle types
S	Set of scenarios
Parameters	
d_{ijtms}	Number of users wanting to travel from region i to region j in period t in scenario s, considering the vehicle type m
x_{i0ms}	Starting allocation of vehicles
c_{ijm}	Relocation cost per vehicle of type m from region i to region j
c_{iim}	Maintenance cost per vehicle of type m in region i
p_{ijm}	Profit per trip with a vehicle of type m from region i to region j
C_m	Fleet capacity - number of vehicles of type m in the fleet
M	Large constant
Variables	
y_{ijtms}	Number of trips with vehicles of type m from region i to region j in period t in scenario s; one-way trips if $i \neq j$, round trips otherwise
u_{ijtms}	Number of unfullfilled demand from users wanting to travel from region i to region j in period t in scenario s, considering the vehicle type m or any vehicle type 'smaller' than m
U_{itms}	Number of unfullfilled demand from users wanting to travel from region i in period t in scenario s, considering the vehicle type m or any vehicle type 'smaller' than m
r_{ijtms}	Number of vehicles of type m being relocated from region i to region j in period t in scenario s
x_{itms}	Number of vehicles of type m at the beginning of period t in region i in scenario s before demand is realized
$\bar{x}_{itms}$	Number of vehicles of type m at the beginning of period t in region i in scenario s after demand is realized

The complete integer linear program is defined as follows:

$$\max_{y,r,x,u,U,U^b,v,v^b} \sum_{s\in S}\left(\pi(s)\sum_{m\in[M]}\sum_{t\in[T-1]}\left(\sum_{i,j\in I}(p_{ijm}y_{ijtms}) - \sum_{i,j\in I, i\neq j}(c_{ijm}r_{ijtms}) - \sum_{i\in I}(c_{iim}v_{itms})\right)\right) \quad (14)$$

s.t.

$$y_{ijtms} = d_{ijtms} - u_{ijtms} \qquad \forall i,j\in I, t\in[T-1], m=0, s\in S \quad (5)$$

$$y_{ijtms} = d_{ijtms} - u_{ijtms} + u_{ijt(m-1)s} \qquad \forall i,j\in I, t\in[T-1], m\in[M]\setminus\{0\}, s\in S \quad (6)$$

$$U_{itms} = \sum_{j\in I} u_{ijtms} \qquad \forall i\in I, t\in[T-1], m\in[M], s\in S \quad (7)$$

$$v_{itms} \leq C_m v^b_{itms} \qquad \forall i\in I, t\in[T-1], m\in[M], s\in S \quad (8)$$

$$U_{itms} \leq M U^b_{itms} \qquad \forall i\in I, t\in[T-1], m\in[M], s\in S \quad (9)$$

$$U^b_{itms} + v^b_{itms} \leq 1 \qquad \forall i\in I, t\in[T-1], m\in[M], s\in S \quad (10)$$

$$x_{itms} = v_{itms} + \sum_{j\in I} y_{ijtms} \qquad \forall i\in I, t\in[T-1], m\in[M], s\in S \quad (11)$$

$$\overline{x}_{itms} = v_{itms} + \sum_{j\in I} y_{jitms} \qquad \forall i\in I, t\in[T-1], m\in[M], s\in S \quad (12)$$

$$x_{i(t+1)ms} = \overline{x}_{itms} + \sum_{j\in I\setminus\{i\}}(r_{jitms} - r_{ijtms}) \quad \forall i\in I, t\in[T-1], m\in[M], s\in S \quad (13)$$

$$y_{ijtms} = y_{ijtms'} \qquad \forall i,j\in I, t\in[T-1], m\in[M], s\in S, s'\in\sigma_t(s) \quad (15)$$

$$r_{ijtms} = r_{ijtms'} \qquad \forall i,j\in I, t\in[T-1], m\in[M], s\in S, s'\in\sigma_t(s) \quad (16)$$

$$y_{ijtms}\in\mathbb{N}_0 \qquad \forall i,j\in I, t\in[T-1], m\in[M], s\in S$$

$$r_{ijtms}\in\mathbb{N}_0 \qquad \forall i,j\in I, t\in[T-1], m\in[M], s\in S$$

$$v_{itms}\in\mathbb{N}_0 \qquad \forall i\in I, t\in[T-1], m\in[M], s\in S$$

$$v^b_{itms}\in\{0,1\} \qquad \forall i\in I, t\in[T-1], m\in[M], s\in S$$

$$u_{ijtms}\in\mathbb{N}_0 \qquad \forall i,j\in I, t\in[T-1], m\in[M], s\in S$$

$$U_{itms}\in\mathbb{N}_0 \qquad \forall i\in I, t\in[T-1], m\in[M], s\in S$$

$$U^b_{ijtms}\in\{0,1\} \qquad \forall i\in I, t\in[T-1], m\in[M], s\in S$$

$$x_{itms}\in\mathbb{N}_0 \qquad \forall i\in I, t\in T, m\in[M], s\in S$$

$$\overline{x}_{itms}\in\mathbb{N}_0 \qquad \forall i\in I, t\in T, m\in[M], s\in S$$

An overview of the assumptions that we made for this model can be seen in Appendix A.

4.6 Extensions

The linear program that is underlying our model can be extended in various ways. We show how to restrict our model to only relocate in certain relocation periods in Section 4.6.1 and how to make our model risk-aware in Section 4.6.2.

4.6.1 Relocation Periods

Some operators may not relocate vehicles over the course of a whole day but only in certain relocation periods. Our model incorporates this behaviour with a set of relocation periods $R \subseteq [T]$. Now we modify Equation 13 to only allow relocations in relocation periods.

$$x_{i(t+1)m} = \overline{x}_{itms} + \sum_{j \in I \setminus \{i\}} (r_{jitm} - r_{ijtm}) \quad \forall i \in I, t \in R, m \in [M] \tag{13.A}$$

$$x_{i(t+1)m} = \overline{x}_{itms} \quad \forall i \in I, t \in [T] \setminus R, m \in [M] \tag{13.B}$$

If $|R| < T$, or in other words if there are less relocation periods than periods, then the models complexity is reduced, which decreases the runtime.

4.6.2 Value-at-risk

Currently, our model is risk-neutral, as the objective function maximizes the expected value of all scenarios. In an economic context, risk-aversion is often desired. To make our model risk-averse, we will extend the model with the value-at-risk. The value-at-risk is used as a risk measure in many domains. We will, however, only look at the value-at-risk in the context of Stochastic Programming. To better understand the value-at-risk, we will first decompose our model's objective function. Let f be the objective function for a single scenario s.

$$f_s(y, r, v) = \sum_{m \in [M]} \sum_{t \in [T-1]} \left(\sum_{i,j \in I} (p_{ijm} y_{ijtms}) - \sum_{i,j \in I, i \neq j} (c_{ijm} r_{ijtms}) - \sum_{i \in I} (c_{iim} v_{itms}) \right) \tag{17}$$

Now the objective function of our model can be written with f.

$$\sum_{s \in S} \left(\pi(s) f_s(y, r, v) \right) \tag{18}$$

The value-at-risk can be calculated for a given solution to the stochastic program and requires an additional parameter $\alpha \in (0, 1)$. For a given $\alpha \in (0, 1)$, the value-at-risk is equal to the largest value η ensuring that the probability of the occurrence of a scenario where the value of the objective function is less than η is lower than $1 - \alpha$. In other words, the value-at-risk is the $1 - \alpha$-quantile of the objective function distribution (Conejo et al., 2010).

$$\mathrm{VaR}(\alpha, y, r, v) = \max\{\eta : \sum_{s: f_s(y,r,v) < \eta} \pi(s) \leq 1 - \alpha\} \tag{19}$$

We can now integrate the value-at-risk into our objective function. To accomplish this we define the weight of the value-at-risk β.

$$(1 - \beta)\Bigg(\sum_{s \in S}\Big(\pi(s) f_s(y, r, v)\Big)\Bigg) + \beta\, \mathrm{VaR}(\alpha, y, r, v) \tag{20}$$

5 Method

In this section, we will explain how to use our model from Section 4 on real-world data to test if it can increase the profitability of FFVSS. To do so, we will first introduce the dataset and analyze it in Section 5.1. In Section 5.2 we will show and explain how we choose the model parameters. In Section 5.2.1 we will also explain how to extract demand from our dataset into a scenario form and how to generate more scenarios. We will then also show how to reduce the scenarios to a reasonable amount, to improve solving time. Then, in Section 5.3 we will explain how we implemented our model and which solver we used.

5.1 Data Preparation and Analysis

The data that we are using is data from the kick-scooter providers Circ, Bird, Lime and Tier, the bicycle providers Ford, KVB, as well as the car-sharing providers Car2Go and DriveNow. This data covers a total of 1 055 091 trips. See Figure 3 for an sample of our dataset and Figure 4 for the share of each provider on the total dataset. The considered service region is Cologne.

	provider	vehicleType	datetime_start	datetime_end	longitude_start	latitude_start	longitude_end	latitude_end
0	circ	kick scooter	2019-11-05 14:44:26	2019-11-05 14:54:08	6.939225	50.93653	6.935835	50.942377
1	circ	kick scooter	2019-11-05 22:49:15	2019-11-18 13:04:11	6.935895	50.94224	6.973340	50.942193

Figure 3: Two Samples From Trip Dataset

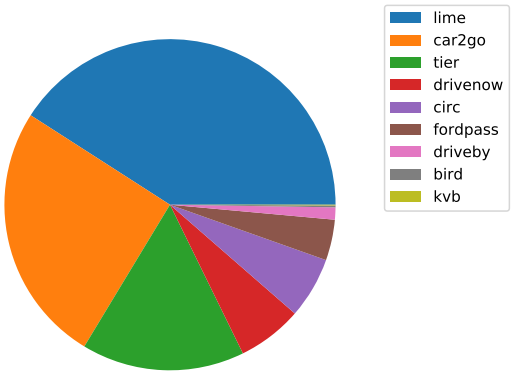


Figure 4: Trips Per Provider

To prepare and analyze the dataset, we use the python data analysis library Pandas and the scientific computing library NumPy and the python libraries

folium and H3-py for geographical visualization and grouping. First, we check our dataset for missing data and outliers. However, the relevant columns in the dataset do not contain any missing values or outliers. Therefore no data preparation is needed.

To better understand our data, in Figure 5, we take a look at the distribution of vehicle types. Most of our data is from kick scooters, while only a small portion is from bicycles.

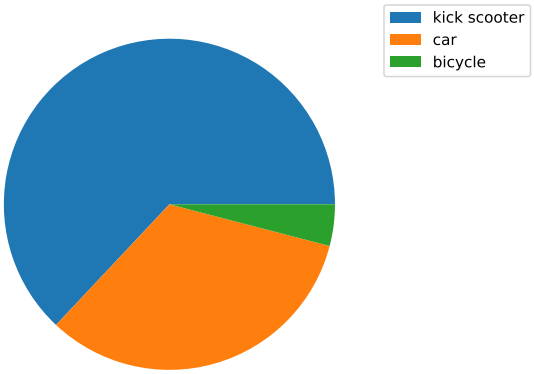


Figure 5: Trips Per Vehicle Type

Our data lies between November and February with 38% of trips in December, as seen in Figure 6. In these months, it is mostly cold, which could impact the number of trips and the preferred vehicle type. For more general demand, one should consider using data from at least a whole year.

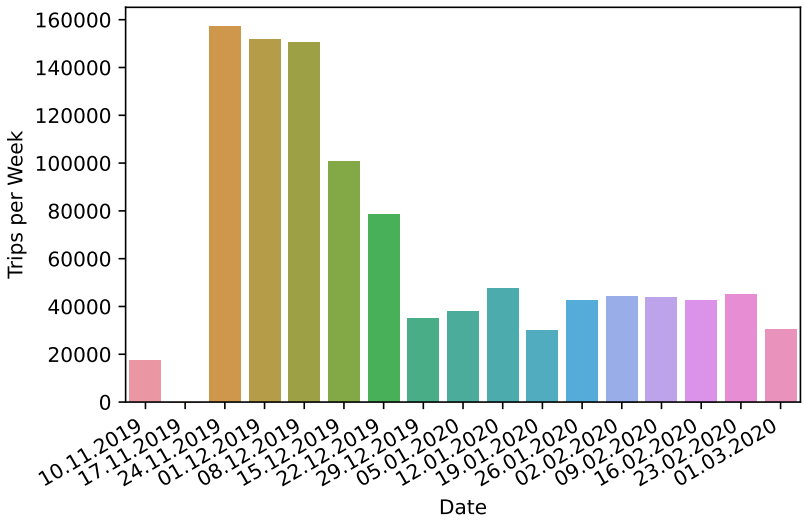


Figure 6: Trips Per Week

Figure 7 shows the trip distance distribution for each vehicle type.

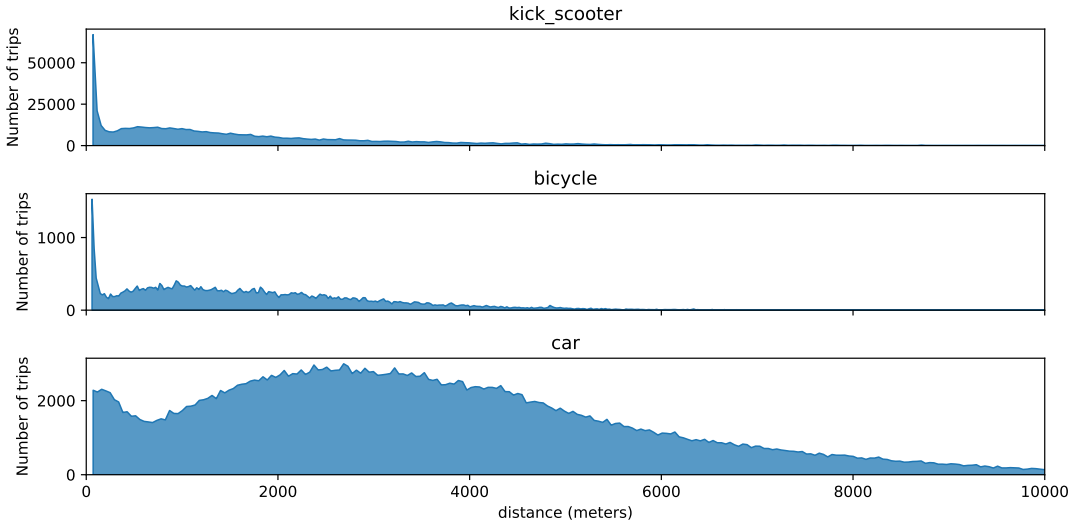


Figure 7: Trips Distance Distribution

We can see that for kick scooters and bicycles, there is a spike right at around 50m trip distance. This most likely is due to round trips. However, for cars, this spike seems to be significantly smaller. This could be because, for cars, the distance per trip is measured by the car itself and not calculated afterwards by the distance between the start- and endpoint. Calculating the distance by the start- and endpoint results in distances near zero for round trips.

We will now investigate how the trip distance distribution of our dataset is consistent or conflicts with our substitution assumptions. We can see that the trip's distance distributions overlap, which is a general indication that substitutions exist. We can also see that users tend to travel long trips (>4km) only with cars. This is consistent with our assumptions that vehicle types exist that cannot be substituted. However, cars are not used very often for short trips (around 1km), which conflicts with our assumption that cars can substitute all other vehicle types.

To verify that there is a statistically significant difference in the trips distances of the different vehicle types, we perform three two-sided t-tests.

Table 3: T-Test Results

Samples	**P-Value**	**Test Statistic**
Cars vs. Bicycles	156.3512	0
Cars vs. Scooters	516.0129	0
Bicycles vs. Scooters	19.9472	$1.6737e-88$

The t-tests show a statistically significant difference between the distances of

the trips of different vehicle types. This further indicates that some trips are too long to be travelled by bicycle or by kick-scooter.

Figure 8 and 9 visualize the difference between the number of trips ending in a region and the number of trips starting in a region. A region with a negative difference, which means that more trips start in this region than trips end there, is colored red. A negative difference leads to a shortage of vehicles and, therefore, to unmet demand. A region with a positive difference, which means that more trips end in this region than trips start there, is colored green. A positive difference leads to unused vehicles and, therefore, a waste of resources. We can interpret these differences as a change in the system's balance. Figure 8 only accounts for trips between 00:00 and 12:00 and figure 9 only accounts for trips between 12:00 and 24:00.

Between 00:00 and 12:00, there are a few regions with more inbound trips than outgoing trips. These regions are mainly business areas and the city's center. Residential areas mostly have more outgoing trips than inbound. This behavior is most likely due to people going to work, and it shows that the system's balance changes between 00:00 and 12:00. Therefore either before or after this period, the system has to be imbalanced. For the period between 12:00 and 00:00, the balance changes are opposite. There are more outgoing trips from the city's center and business areas than there are ingoing. Also, there are more inbound trips in residential areas than there are outgoing. This behavior is most likely due to people going home after work. Therefore balance changes also occur in this period, which means that there has to be an imbalance before or after the period.

One might argue that the balance changes in these two periods equalize one another. However, this does not seem to be the case. As seen in figure 10, which accounts for all trips, the balance changes even occur long-term. So there are short-term imbalances during each day and long-term imbalances. These imbalances show the importance of vehicle relocations.

5.2 Model Configuration

The model's parameters as found in the stochastic program are the demand d_{ijtms}, the probability of each scenario $\pi(s)$, the cost c_{ijm} and the profit p_{ijm} of each possible trip route, as well as the parking costs c_{iim}, and the initial allocation of vehicles x_{i0ms}. We index those parameters and the model's decision variables by the set of regions I, the set of periods $[T]$, the set of vehicle types $[M]$, and the set of scenarios S. We will start by choosing reasonable values for the indices.

To define the set of regions, we will use H3, a Hexagonal Hierarchical Spatial

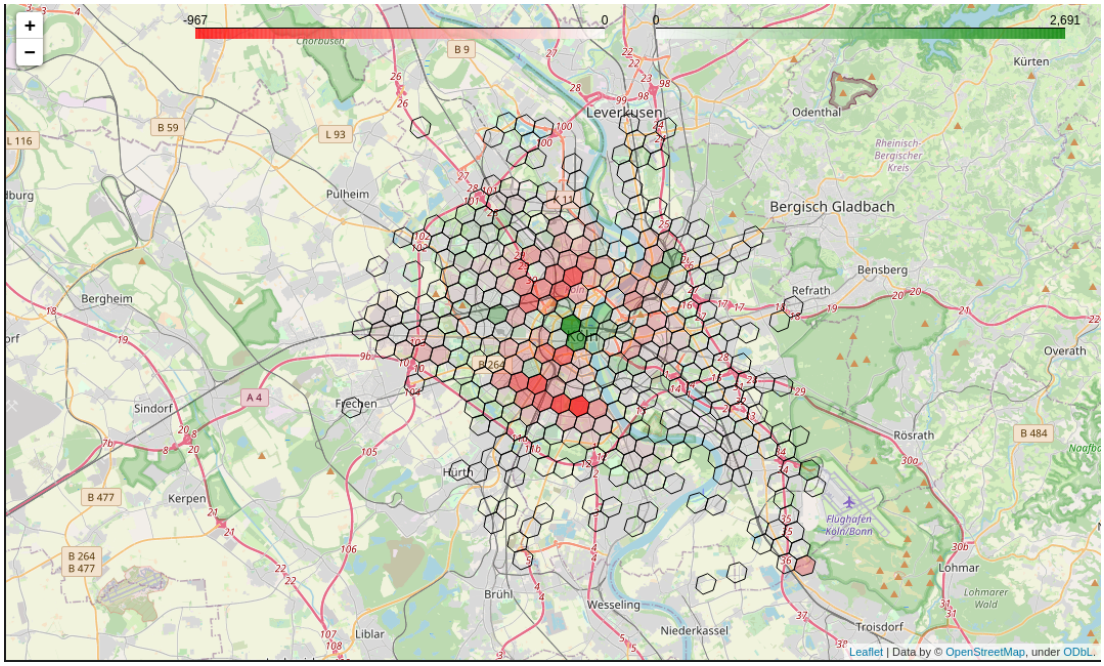


Figure 8: Balance Changes - Morning

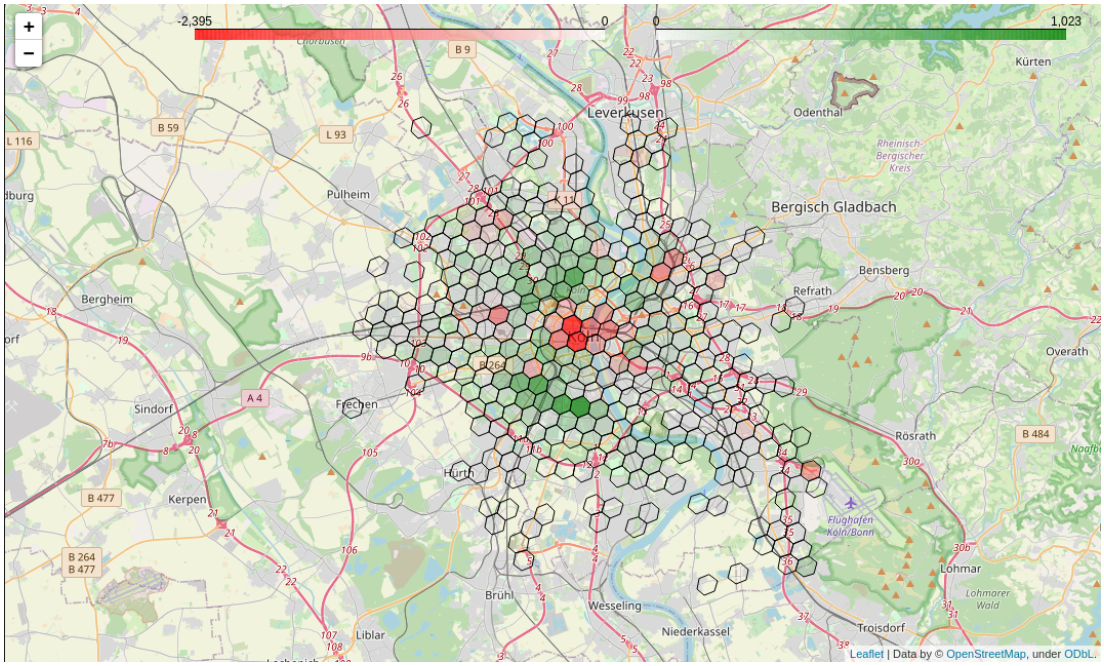


Figure 9: Balance Changes - Evening

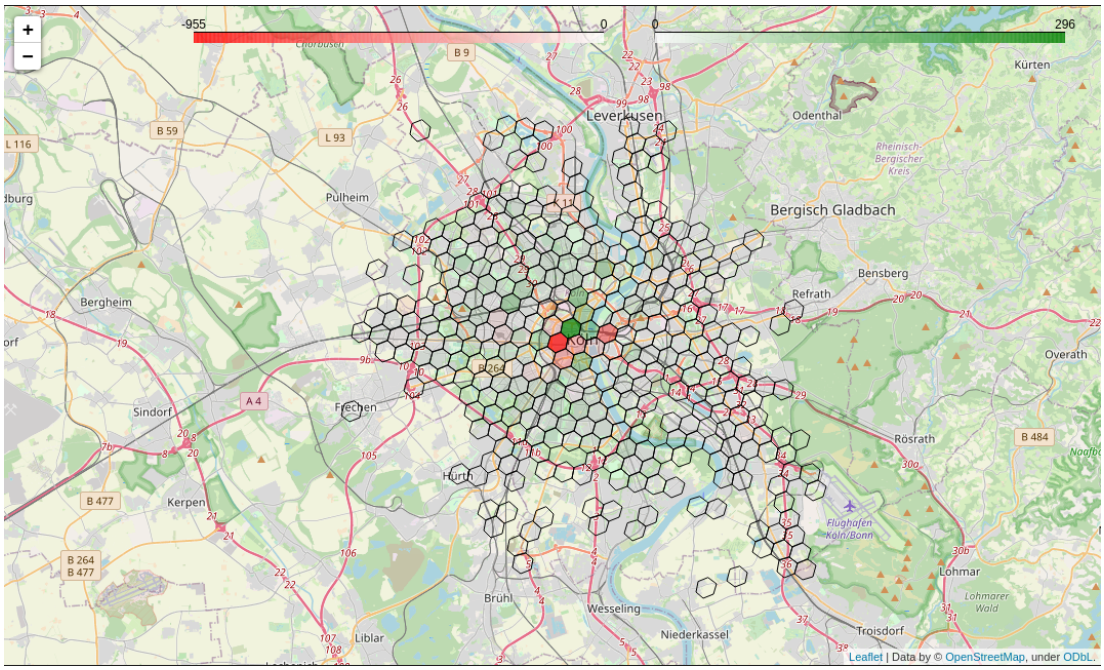
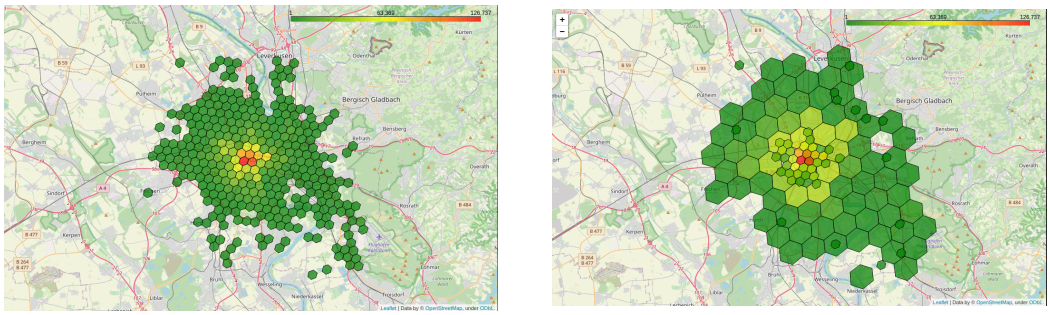


Figure 10: Balance Changes - Total

Index. H3 divides the Earth's surface into hexagons. We can use these hexagons to aggregate spatial data into regions. H3 allows using different resolutions. The resolution defines the size of each hexagon and, therefore, how many hexagons there are. A higher resolution leads to more, smaller hexagons, while a lower resolution leads to fewer, larger hexagons. Therefore, we have to decide which resolution to use. As our model assumes that demand in a region can be satisfied by any suitable vehicle in that region, we have to consider the maximum distance users of the system are willing to walk to the vehicles. Little research has been done on what the accepted walking distance for FFVSS is. Wang and Yan (2016) suggest that users' walking time should not exceed ten minutes. With the average walking speed of pedestrians, this leads to a maximum walking distance of 642m (Fitzpatrick et al., 2006). Therefore the most suitable resolutions are resolution seven and eight with an average edge length of 1220m and 461m, respectively. However, we also have to consider that a lower resolution will lead to a larger number of regions and, therefore, increase our model's dimensionality. Therefore, we will use different resolutions across the service region according to the demand in that region. We will first define the set of regions as all hexagons of resolution eight where any trip started or ended. Then we will downscale 80% of regions to resolution seven. This downscaling is visualized in Figure 11.

We will define our set of periods for a whole day, as we saw in Section 5.1 that the short-term imbalances occur over a day. For the sake of simplicity, we will choose the same duration for all periods. However, it would be possible to select periods with different durations, similarly as we choose regions with different



(a) Pre-Downscaling (b) Post-Downscaling

Figure 11: Downscaling Of Resolution

sizes. As all trips in the service region have to be completed in one period, the period duration should not be too short. We also know from our data analysis that vehicles are used on average four times a day. However, there can be only one trip per vehicle per period. Therefore the period duration should not be too long. To balance these two assumptions, we set the period duration to two hours.

We will use kick scooters, bicycles, and cars as vehicle types in that order as those are the vehicle types in our dataset. Note that the order of the vehicles means that cars can substitute all bicycle demand, and cars and bicycles can substitute all kick scooter demand. These substitutions are plausible as we found in our data analysis that cars are used for longer trips than bicycles and kick scooters, and kick scooters are used for longer trips than bicycles and cars. Our findings are consistent with Scheiner (2010), who found that less than 10% of people use bicycles to perform trips when traveling a distance greater than 3km.

The higher the number of scenarios, the more accurate and realistic our model will be. However, increasing the number of scenarios increases the models' dimensionality and can make the problem computationally intractable.

5.2.1 Scenarios

As demand in our model is uncertain, it will be different for each scenario. We will now show how to use our dataset to obtain an arbitrary amount of scenarios.

Extraction And Generation To generate scenarios we assume that the demand from one region to another in a specific time period for a specific vehicle type follows a poisson distribution. The Poisson distribution describes the probabilities of a certain number of events occurring over a fixed period. In our case, the events are a person wanting to travel to another region (demand). The Poisson distribution requires the events to be independent of one another and that the average rate of events occurring is constant, which is plausible in the case of demand. The Poisson distribution is described by only one parameter λ, which is

equal to the mean and the variance of the distribution. We will use our dataset to estimate the Poisson distribution for demand for each start region, end region, period, vehicle type quadruple. To estimate the distribution, we will use the maximum likelihood estimate of the Poisson distribution, as seen in Equation 21. Let $k_1, ..., k_n$ be samples from a poisson distribution, then the maximum likelihood estimate of parameter λ is given by the following formula:

$$\hat{\lambda}_{MLE} = \frac{1}{n} \sum_{i=1}^{n} k_i \tag{21}$$

We can now calculate the average number of trips for each start region, end region, period, vehicle type quadruple, which will then be the maximum likelihood estimate for the Poisson distribution. Using the estimate, we can generate an arbitrary number of scenarios.

To properly use the advantages of stochastic programming, we need the set of scenarios to follow a scenario tree structure as described in 3.2. To ensure this structure, we start by generating one demand value for each quadruple in the first period and then use that demand for all scenarios. We call the number of different demand values per quadruple in a specific period the number of realizations. Therefore, the first period has one realization, which can be seen as the root node in the scenario tree. We increase the number of realizations for each subsequent period by multiplying the previous number of realizations by a fixed factor, which we call the branching factor. Note that the branching factor is the number of realizations of the underlying discrete random variables. However, we refer to this number as the branching factor to not confuse it with the total number of realizations in one period. For each period, we assign each scenario to one of the realizations of that period. Let $|S|$ be the number of scenarios we want to generate, T the number of periods, $|\omega^t|$ the number of realizations in period t, and k the branch-factor, then the two following equations hold.

$$k^{T-1} = |S| \tag{22}$$

$$|\omega^t| = k^t \; \forall t \in [T] \tag{23}$$

Reduction A common approach in stochastic programming is to generate a large number of scenarios and reduce them afterward. Scenario reduction techniques make use of clustering methods so that the reduced number of scenarios is a good representation of the whole. We group our data into multiple clusters where the scenarios that belong to the same cluster are similar. We can then

choose one sample from each of these clusters, representing all scenarios in its corresponding cluster. We call these representative samples the cluster centers as we will choose those closest to the actual average of all samples in a cluster. The set of all representatives is the reduced set of scenarios. Note that the probability of occurrence for each representative will be the sum of each scenario's probability in the cluster of the representative. We will use k-medoids as the clustering technique. In contrast to the better known k-means technique, k-medoids has the advantage that the centers of each cluster are actual samples, which leads to more realistic centers. K-mean simple uses the average of all samples in a cluster as the cluster's center, leading to non-integer values in the center. In our case, this would be particularly problematic, as demand values have to be integers.

5.2.2 Initial Allocation, Profit And Cost

In reality, the initial allocation of vehicles is given by the actual allocation from the previous day. For the sake of simplicity, we will distribute the vehicles equally among all regions.

The profit of a trip depends on the trip's duration as well as the fuel consumption. Our model does not consider the trip's duration. Therefore we will approximate the real profit with the help of the trip's distance. We calculate the profit per kilometer using the average speed of each vehicle type and use the price per minute from the providers Tier, Nextbike, and ShareNow for the different vehicle types, respectively. Note that the average speed from our dataset may be lower than in reality because we cannot consider when and how long vehicles standstill during rent.

Tier charges 19 cents per minute with a unlock fee of 1 €. However, we do not consider this fee. NextBike charges 1 €\30 minutes for the basic rate. ShareNow has different charging fees for different cars. We set the price per minute to 0.29 €. See Table 4 for the profit per kilometer.

As we do not have any data on relocation costs, we can only make assumptions. It seems reasonable that car relocation costs are much higher than the relocations of bicycles and kick-scooters because multiple bicycles and kick-scooters can be relocated at once. We assume that relocation takes place with the same speed as trips, and relocation drivers are paid a salary of 14 €\hour. This assumption leads us to a cost of approximately 2.09€\km. We also assume that 20 bicycles or 40 kick-scooters can be relocated at the same time.

To calculate profit p_{ijm} and cost c_{ijm} for each possible trip route with the help of cost and profit per kilometer, we need to know the distance of a trip from region i to region j. If $i \neq j$, we will use the (haversine) distance between the centers of region i and region j given by H3. If $i = j$ we set fixed parking cost

Table 4: Profits and Costs

Vehicle Type	Average Speed (km\min)	Profit (€\min)	Profit (€\km)	Cost (€\min)	Cost (€\km)
Car	0.1117	0.29	2.5973	0.2333	2.0899
Bicycle	0.0497	$0.0\overline{3}$	0.6710	0.0116	0.2349
Kick Scooter	0.2244	0.19	0.8467	0.0058	0.0260

to 1 €\h for cars, 0.2 €\h for mopeds and 0.1 €\h for bicycles. Parking costs for FFVSS vary (Shaheen et al., 2019) and we do not have any data on what the prices for FFVSS are in Cologne. For the distance of a trip from region i to region i we will use a formula, that calculates the average distance between two points in a hexagon with a certain side length. We derived this formula by conducting multiple simulations with hexagons with different side length. The formula and the simulation can be found in our implementation in Appendix B.

5.3 Implementation And Solver

We implement our linear program with the python library PuLP, which allows us to create linear programs and solve them with various available solvers. We use CPLEX as the solver. All of our benchmarks are done on a laptop with a Ryzen 5800H as CPU, running with 16 threads at 3.2GHz, and 32GB of RAM.

6 Results

To evaluate our model's performance, we implement two different benchmarks. In the first benchmark, we test whether relocations, in general, improve profitability. In the second benchmark, we test whether considering multiple vehicle types has an advantage over looking at each vehicle type separately. All our benchmarks are done without the value-at-risk in the objective function to interpret the objective function as actual profit. We will examine how the value-at-risk influences the performance of our model. As our model is a stochastic program, we will also calculate the value of perfect information.

We were not able to use the model configurationa as described in Section 5.2, because our model has high computational costs and we run all benchmarks on a personal computer. The configuration we used can be found in Appendix C.

6.1 No Relocations Benchmark

To evaluate our model's relocations we will compare the optimal value to the optimal value of the same linear program, but without relocations. To accomplish this we introduce a new constraint that disables relocations.

$$r_{ijtms} = 0 \quad \forall i, j \in I, i \neq j, t \in [T], m \in [M], s \in S \tag{24}$$

Figure 12 provides the optimal objective for different vehicle fleet sizes. Note that as our model includes parking costs, the objective may be negative.

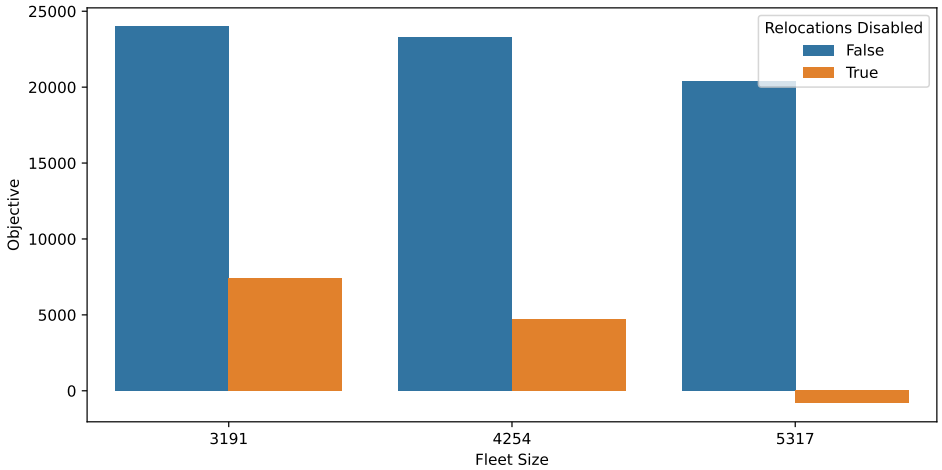


Figure 12: No Relocation Benchmark

We can see that for all fleet sizes, the improvement in profit through relocations is larger than 60%. This improvement shows that relocations are generally

profitable. As we run this benchmark with different fleet sizes, we can also see that the profitability decreases with a larger fleet size due to parking costs.

Figure 13 shows the average number of trips of all scenarios and the average number of unfulfilled demand for all scenarios. Note that the unfulfilled demand and the number of trips are perfectly negatively correlated, but we show both for the sake of exposition.

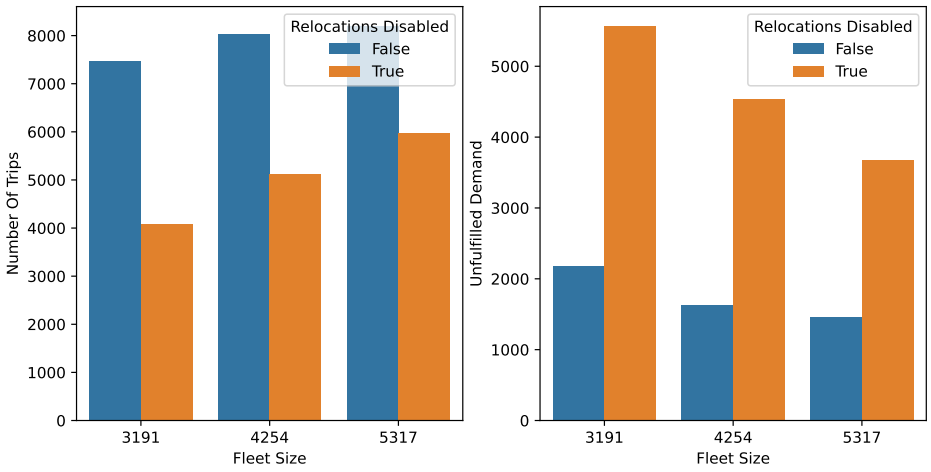


Figure 13: No Relocation Benchmark Side Effects

As we can see, relocations improve not only the profitability but also the number of trips and the unfulfilled demand.

6.2 Single Modal Benchmark

To test whether our model has an advantage over models that only consider a single vehicle type, we compare the results of our model to three instances of the same model with only one vehicle type each. We start by running the instance that only considers kick scooters and then use the unfulfilled demand of that model as additional demand for the instance that considers bicycles. We repeat this process with the instance for cars. Thereby the three instances model almost the same behaviour as one instance that considers all three vehicle types at once. The difference is that the three instances with only one vehicle type optimize without considering any substitions. Table 5 shows the profit, the number of trips, the unfulfilled demand and the number of relocations for the multi-modal instance and the three single modal instances. Note that the number of trips, unfulfilled demand, and relocations are the average overall scenarios and can therefore be non-integer.

As we can see, the profitability is improved by 548% in the multi-modal

Table 5: Single Modal vs. Multi Modal

Type	Profit (€)	Trips	Unful. Demand	Relocations	Runtime (s)
Multi	23305.031111	8026.75	1627.50	2367.0	869.7
Single	3594.434444	8735.50	918.75	2919.0	1.74

instance. However, we see that the number of trips is decreased, and the number of unfulfilled demand is increased in the multi-modal instance. This shows that it is not always profitable to facilitate the maximum number of trips. The runtime of the multi-modal instance is almost 500 times longer than the three single modal instances combined. This drastic increase in runtime is caused by an increased dimensionality in the underlying linear program.

6.3 Value-at-risk Runtime Benchmark

The value-at-risk is a widely adopted risk measure that allows us to implement risk-awareness in our model. We will, therefore, only test how the modification of the objective function that implements the value-at-risk into our model impacts performance. Figure 6 shows the solvers runtime for a given weighting factor $\beta = 0.00, 0.25, 0.50, 0.75$.

Table 6: Value-at-risk Runtime Benchmark

β	Runtime (s)
0.00	907.85
0.25	2583.36
0.50	2889.62
0.75	545.03

As we can see, we get contradictory results. The runtime for $\beta = 0.25, 0.50$ is more than twice as long as for $\beta = 0.00$. However, further increasing β decreases the runtime drastically. We suspect that the cause of this is the unpredictable and erratic behavior of runtime performance in tree search methods, like the branch-and-cut algorithm that is used in CPLEX (Fischetti & Monaci, 2014).

6.4 Value Of Perfect Information

The value of perfect information describes the improvement in profit when demand would be perfectly known without any uncertainty. It can be interpreted as the maximum price for which it would be worth buying information on how the demand realization will be. In our case, the value of perfect information indicates how valuable good predictions are for a vehicle sharing system. To calculate the

value of perfect information, we solve a relaxed version of our stochastic program without the non-anticipativity constraints. The value of perfect information is then defined as the difference between the optimal value of the relaxed stochastic program and the original one. Figure 14 shows the optimal value of the stochastic program for different fleet sizes with and without non-anticipativity constraints. The gap between each pair of bars is the value of perfect information.

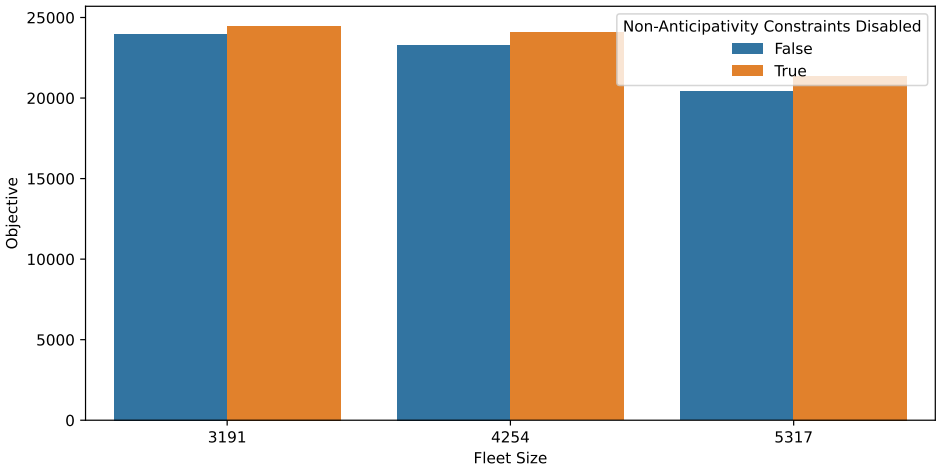


Figure 14: Value Of Perfect Information

We see that the value of perfect information in all three cases is relatively small (2.0% - 4.5%) compared to the optimal value itself. Note that the value of perfect information largely depends on the divergence of the scenarios, or in other words how far apart their values are.

7 Discussion

To the best of our knowledge, no other papers deal with relocation strategies in FFVSS with substitutional effects between multiple vehicle types. We show that if substitutional effects exist as we have described them in our model, considering them when planning relocations can significantly increase profit.

Even though our model increased profitability, it decreased the total number of trips made and increased unfulfilled demand. These adverse side effects are most likely undesirable. We expect, however, that if the objective function is modified to include those metrics, they can be improved as well. We found modelling substitutions of vehicle types in a linear program to be hard and not very flexible. Therefore, the assumptions we made on these substitutions are chosen so that they can be modelled efficiently in a linear program without drastically increasing its complexity. Even though some of our data analysis supports our assumptions, and we think they are a reasonable approximation of real-world behaviour, our assumptions are not backed by literature and need validation. Further research could investigate what factors influence users to choose a specific vehicle type over another and when one vehicle type can substitute another.

Solving our model proves to be very computationally intensive. Including multiple vehicle types effectively increases the size of the underlying linear program by a factor that is the number of considered vehicle types. However, previous research on relocations in vehicle fleets with a single-vehicle type shows that various techniques, e.g. decomposition, can decrease computational costs of stochastic programs. We think that implementing these techniques could significantly improve the solving time of our model and multi-modal models in general.

Literature suggests that including setup costs for relocations can have a significant impact on optimal relocation strategies (Zhao et al., 2020). Therefore, our model can be made more realistic by implementing these setup costs. Setup costs would most likely increase the model's complexity and computational cost even further and would therefore require optimizations first.

Like most stochastic programs, our model is vulnerable to deviations from the considered scenarios. A high number of scenarios leads our model to become computationally intractable very fast. Possible further research could therefore implement robust optimization like Tang et al. (2020) did for single-model systems to better manage these deviations.

When extending our model with the value-at-risk to make it risk-aware, we observed erratic behaviour in the solver's runtime. As erratic behaviour is generally undesirable, further research can implement other risk measures that do not suffer from the drawbacks of the value-at-risk. However, we suspect that the

erratic runtime behaviour is inevitable in large-scale stochastic programs like our model.

The validity of our model largely depends on how realistic and representative the demand in the used scenarios is. We extract the demand from a large trip dataset. This dataset, however, only accounts for fulfilled demand and not for unfulfilled demand. This is a problem that not only our model suffers from, which underlines its importance. Therefore, further research could examine ways to also collect data on unfulfilled demand.

Our model accounts for round-trips. However, because it does not consider trip duration, but only the distance between start and end point, it underestimates the profit of round trips. This problem could be addressed by incorporating the trips' duration into our model.

8 Conclusion

In this paper, we developed a model to determine optimal relocations in a FFVSS, which takes different vehicle types and substitutional effects between them into account. Our model proved to increase profitability in comparison to models that only consider a single vehicle type. However, our model also has high computational costs, as it is underlying a large-scale integer linear program. Nevertheless, vehicle sharing system providers that operate vehicle fleets with multiple vehicle types need to take substitutional effects into account to improve the value of relocations in these systems.

A Assumptions

- If demand can be satisfied by multiple vehicle types, the model can decide which vehicle type to use
- Only one trip per vehicle per period
- The trip duration is always shorter than the period's duration
- Trips do not span more than one period
- Unlimited relocations are allowed
- Costs for relocators to travel between relocations are not depicted
- Customer demand is lost if there are no vehicles in the region
- Relocation cost is proportional to the number of relocated vehicles, i.e. setup costs are not considered

B Implementation

The implementation of our model can be found on github.
https://github.com/mogottsch/vehicle-reposition-2

C Benchmark Model Configuration

H3 Resolution	7
Downscale Quantile	0.9
Period Duration	8 hours
Number Of Periods	3
Number Of Relocation Periods	1
Number Of Scenarios After Reduction	4

D Proofs

Theorem 1. *Let S be a set of scenarios, I a set of regions, $[M]$ a set of vehicles, and $[T]$ a set of periods. Then for all $t \in [T-1]$ and for all $s \in S, s' \in \sigma_t(s)$ the following equation holds.*

$$u_{ijtms} = u_{ijtms'} \; \forall i, j \in I, m \in [M] \tag{25}$$

Proof. Induct on m.
Base case (m = 0):

$$
\begin{aligned}
& y_{ijtms} = d_{ijtms} - u_{ijtms} && \text{[equation 5]} \\
\iff & y_{ijtms} - d_{ijtms} = -u_{ijtms} \\
\iff & y_{ijtms'} - d_{ijtms'} = -u_{ijtms} && \text{[equation 15]} \\
\iff & -u_{ijtms'} = -u_{ijtms}
\end{aligned}
$$

Inductive Hypothesis: Assume $u_{ijtms} = u_{ijtms'}\ \forall i, j \in I$ for some $m \in [M]$

Inductive Step: $[m-1 \to m]$

$$
\begin{aligned}
& y_{ijtms} = d_{ijtms} - u_{ijtms} + u_{ijt(m-1)s} && \text{[equation 6]} \\
\iff & y_{ijtms'} = d_{ijtms} - u_{ijtms} + u_{ijt(m-1)s} && \text{[equation 16]} \\
\iff & d_{ijtms'} - u_{ijtms'} + u_{ijt(m-1)s'} = d_{ijtms} - u_{ijtms} + u_{ijt(m-1)s} \\
\iff & d_{ijtms} - u_{ijtms'} + u_{ijt(m-1)s} = d_{ijtms} - u_{ijtms} + u_{ijt(m-1)s} && \text{[I.H.]} \\
\iff & u_{ijtms'} = u_{ijtms}
\end{aligned}
$$

□

Theorem 2. *Let S be a set of scenarios, I a set of regions, $[M]$ a set of vehicles, and $[T]$ a set of periods. Then for all $t \in [T-1]$ and for all $s \in S, s' \in \sigma_t(s)$ the following equation holds.*

$$U_{itms} = U_{itms'}\ \forall i \in I, m \in [M] \tag{26}$$

Proof.

$$
\begin{aligned}
& U_{itms} = \sum_{j \in I} u_{ijtms} && \text{[equation 7]} \\
\iff & U_{itms} = \sum_{j \in I} u_{ijtms'} && \text{[equation 25]} \\
\iff & U_{itms} = U_{itms'}
\end{aligned}
$$

□

Theorem 3. *Let S be a set of scenarios, I a set of regions, $[M]$ a set of vehicles, and $[T]$ a set of periods. Then for all $t \in [T-1]$ and for all $s \in S, s' \in \sigma_t(s)$ the following equations hold.*

$$x_{itms} = x_{itms'} \; \forall i \in I, m \in [M] \tag{27}$$

$$v_{itms} = v_{itms'} \; \forall i \in I, m \in [M] \tag{28}$$

$$\overline{x}_{itms} = \overline{x}_{itms'} \; \forall i \in I, m \in [M] \tag{29}$$

Proof. Induct on i

Base case (i = 0):
(27): trivial, as x_{i0ms} is fixed parameter and equal for all scenarios
(28):

$$\begin{aligned}
& x_{i0ms} = v_{i0ms} + \sum_{j \in I} y_{ij0ms} && \text{[equation 12]} \\
\iff & x_{i0ms} - \sum_{j \in I} y_{ij0ms} = v_{i0ms} \\
\iff & x_{i0ms'} - \sum_{j \in I} y_{ij0ms'} = v_{i0ms} && \text{[equation 16]} \\
\iff & v_{i0ms'} = v_{i0ms}
\end{aligned}$$

(29):

$$\begin{aligned}
& \overline{x}_{i0ms} = v_{i0ms} + \sum_{j \in I} y_{ji0ms} && \text{[equation 11]} \\
\iff & \overline{x}_{i0ms} = v_{i0ms'} + \sum_{j \in I} y_{ji0ms'} && \text{[equation 15]} \\
\iff & \overline{x}_{i0ms} = \overline{x}_{i0ms'}
\end{aligned}$$

Inductive Hypothesis:
Assume

$$x_{itms} = x_{itms'} \; \forall i \in I, m \in [M]$$

$$v_{itms} = v_{itms'} \; \forall i \in I, m \in [M]$$

$$\overline{x}_{itms} = \overline{x}_{itms'} \; \forall i \in I, m \in [M]$$

for some $t \in [T]$

Inductive Step: $[t-1 \to t]$

(27):

$$x_{itms} = \overline{x}_{i(t-1)ms} + \sum_{j \in I \setminus \{i\}} (r_{ji(t-1)ms} - r_{ij(t-1)ms}) \qquad \text{[equation 13]}$$

$$\iff x_{itms} = \overline{x}_{i(t-1)ms'} + \sum_{j \in I \setminus \{i\}} (r_{ji(t-1)ms'} - r_{ij(t-1)ms'}) \qquad \text{[I.H. \& equation 16]}$$

$$\iff x_{itms} = x_{itms'}$$

(28): Same proof as in base case.

(29): Same proof as in base case.

□

References

Benders, J. F. (1962, December). Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, *4*(1), 238–252. Retrieved 2021-06-03, from `https://doi.org/10.1007/BF01386316` doi: 10.1007/BF01386316

Benjaafar, S., Jiang, D., Li, X., & Li, X. (2018, December). *Dynamic Inventory Repositioning in On-Demand Rental Networks* (SSRN Scholarly Paper No. ID 2942921). Rochester, NY: Social Science Research Network. Retrieved 2020-11-17, from `https://papers.ssrn.com/abstract=2942921` doi: 10.2139/ssrn.2942921

Birge, J. R., & Louveaux, F. (2011). *Introduction to stochastic programming* (2nd ed ed.). New York: Springer.

Chen, M., Wang, D., Sun, Y., Waygood, E. O. D., & Yang, W. (2020, April). A comparison of users' characteristics between station-based bikesharing system and free-floating bikesharing system: case study in Hangzhou, China. *Transportation*, *47*(2), 689–704. Retrieved 2021-01-15, from `https://doi.org/10.1007/s11116-018-9910-7` doi: 10.1007/s11116-018-9910-7

Conejo, A. J., Carrión, M., & Morales, J. M. (2010). *Decision making under uncertainty in electricity markets* (No. 153). New York, NY: Springer. (OCLC: 845807990)

Fischetti, M., & Monaci, M. (2014, February). Exploiting Erraticism in Search. *Operations Research*, *62*. doi: 10.1287/opre.2013.1231

Fitzpatrick, K., Brewer, M. A., & Turner, S. (2006, January). Another Look at Pedestrian Walking Speed. *Transportation Research Record*, *1982*(1), 21–29. Retrieved 2020-12-18, from `https://doi.org/10.1177/0361198106198200104` (Publisher: SAGE Publications Inc) doi: 10.1177/0361198106198200104

Fricker, C., & Gast, N. (2012, January). Incentives and Redistribution in Homogeneous Bike-Sharing Systems with Stations of Finite Capacity. *EURO Journal on Transportation and Logistics*, *5*. doi: 10.1007/s13676-014-0053-5

He, L., Hu, Z., & Zhang, M. (2019, April). Robust Repositioning for Vehicle Sharing. *Manufacturing & Service Operations Management*, *22*(2), 241–256. Retrieved 2020-11-17, from `https://pubsonline.informs.org/doi/abs/`

10.1287/msom.2018.0734 (Publisher: INFORMS) doi: 10.1287/msom.2018.0734

Jian, N., Freund, D., Wiberg, H. M., & Henderson, S. G. (2016, December). Simulation optimization for a large-scale bike-sharing system. In *Proceedings of the 2016 Winter Simulation Conference* (pp. 602–613). Arlington, Virginia: IEEE Press.

Jorge, D., & Correia, G. (2013, June). Carsharing systems demand estimation and defined operations: a literature review. *European Journal of Transport and Infrastructure Research*, *13*(3). Retrieved 2021-06-08, from https://journals.open.tudelft.nl/ejtir/article/view/2999 (Number: 3) doi: 10.18757/ejtir.2013.13.3.2999

Li, Y., Zheng, Y., & Yang, Q. (2018, July). Dynamic Bike Reposition: A Spatio-Temporal Reinforcement Learning Approach. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 1724–1733). New York, NY, USA: Association for Computing Machinery. Retrieved 2020-12-22, from https://doi.org/10.1145/3219819.3220110 doi: 10.1145/3219819.3220110

Lu, M., Chen, Z., & Shen, S. (2017, October). Optimizing the Profitability and Quality of Service in Carshare Systems Under Demand Uncertainty. *Manufacturing & Service Operations Management*, *20*(2), 162–180. Retrieved 2020-11-17, from https://pubsonline.informs.org/doi/10.1287/msom.2017.0644 (Publisher: INFORMS) doi: 10.1287/msom.2017.0644

Lu, M., Chen, Z., & Shen, S. (2018, May). Optimizing the Profitability and Quality of Service in Carshare Systems Under Demand Uncertainty. *Manufacturing & Service Operations Management*, *20*(2), 162–180. Retrieved 2020-11-17, from http://pubsonline.informs.org/doi/10.1287/msom.2017.0644 doi: 10.1287/msom.2017.0644

Reiss, S., & Bogenberger, K. (2015, September). GPS-Data Analysis of Munich's Free-Floating Bike Sharing System and Application of an Operator-based Relocation Strategy. In *2015 IEEE 18th International Conference on Intelligent Transportation Systems* (pp. 584–589). (ISSN: 2153-0017) doi: 10.1109/ITSC.2015.102

Scheiner, J. (2010, January). Interrelations between travel mode choice and trip distance: trends in Germany 1976–2002. *Journal of Transport Geography*, *18*(1), 75–84. Retrieved 2020-12-18, from http://www.sciencedirect.com/science/article/pii/S0966692309000052 doi: 10.1016/j.jtrangeo.2009.01.001

Schmöller, S., Weikl, S., Müller, J., & Bogenberger, K. (2015, July). Empirical analysis of free-floating carsharing usage: The Munich and Berlin case. *Transportation Research Part C: Emerging Technologies*, *56*, 34–51. Retrieved 2021-01-15, from http://www.sciencedirect.com/science/article/pii/S0968090X1500087X doi: 10.1016/j.trc.2015.03.008

Shaheen, S., Cohen, A., Randolph, M., Farrar, E., Davis, R., & Nichols, A. (2019). *Shared Mobility Policy Playbook*. Retrieved 2021-01-22, from https://trid.trb.org/view/1687681

Shu, J., Chou, M. C., Liu, Q., Teo, C.-P., & Wang, I.-L. (2013, November). Models for Effective Deployment and Redistribution of Bicycles Within Public Bicycle-Sharing Systems. *Operations Research*, *61*(6), 1346–1359. Retrieved 2020-11-17, from https://pubsonline.informs.org/doi/abs/10.1287/opre.2013.1215 (Publisher: INFORMS) doi: 10.1287/opre.2013.1215

Tang, Q., Zhang, Y., & Zhou, M. (2020, May). *Vehicle Repositioning under Uncertainty* (SSRN Scholarly Paper No. ID 3612626). Rochester, NY: Social Science Research Network. Retrieved 2020-11-17, from https://papers.ssrn.com/abstract=3612626 doi: 10.2139/ssrn.3612626

Wang, N., & Yan, R. (2016, January). Research on Consumers' Use Willingness and Opinions of Electric Vehicle Sharing: An Empirical Study in Shanghai. *Sustainability*, *8*(1), 7. Retrieved 2020-12-18, from https://www.mdpi.com/2071-1050/8/1/7 (Number: 1 Publisher: Multidisciplinary Digital Publishing Institute) doi: 10.3390/su8010007

Zhang, Y., & Mi, Z. (2018, June). Environmental benefits of bike sharing: A big data-based analysis. *Applied Energy*, *220*, 296–301. Retrieved 2021-01-15, from http://www.sciencedirect.com/science/article/pii/S0306261918304392 doi: 10.1016/j.apenergy.2018.03.101

Zhao, L., Liu, Z., & Hu, P. (2020, November). Dynamic repositioning for vehicle sharing with setup costs. *Operations Research Letters*, *48*(6), 792–797. Retrieved 2020-11-17, from http://www.sciencedirect.com/science/article/pii/S0167637720301486 doi: 10.1016/j.orl.2020.09.010