POLITECNICO DI TORINO

Master's Degree in Computer Engineering

Master's Degree Thesis

Relocation strategies for e-scooter system optimization

Supervisors

Prof. Marco MELLIA

Prof. Danilo GIORDANO

Prof. Luca VASSIO

MSc. Alessandro CIOCIOLA

Candidate

Leonardo TOLOMEI

April 2021

Summary

In the last few years, the concept of micromobility has gained a lot of ground, thanks to the introduction of new transportation modes, like e-scooters. These new, lightweight, electric vehicles are cheap, easy and fun to ride, and they are increasingly becoming a reasonable solution for first- and last-mile trips. New companies started spreading e-scooters throughout different cities around the world, with the aim of offering a new, dockless, e-scooter sharing service. These companies are growing fast and they have already been able to attract several hundred million dollars of investments.

E-scooters - and micromobility in general - have the capability to reduce pollution and traffic congestion, but researchers are questioning if they are really positively contributing to solve these issues. It seems that, moving e-scooters through the city with other motorized vehicles for charging, deployment and relocation, is a major component of their entire life-cycle emissions. Moreover, an even bigger contribution to these emissions is given by manufacturing. Intensive use and vandalism cause a shorter vehicle lifetime and, consequently, a higher e-scooter production demand, thus increasing emissions associated with manufacturing. It is therefore useful to keep the number of deployable e-scooters low, and to try to maximize their utilization, repositioning them in a reasonable way, possibly also reducing the total amount of transportation distance.

In this thesis, we analyze different relocation algorithms, in order to understand, first of all, if it is useful and profitable to relocate. We show how different relocation strategies affect the system, both in terms of performance and costs.

For this purpose, we adopt and extend an existing data-driven, discrete-event simulator for Free-Floating Vehicle Sharing Systems (FFVSS). We introduce new datasets of e-scooter trips, cleaning and preparing them to be utilized with the simulator, and reaching a total amount of 7 North-American cities. We extract a demand model from data and we use it as input for the simulator. We introduce new reactive and proactive algorithms, with the aim of solving the relocation problem with a greedy approach. We test each algorithm under different scenarios and we make cost and revenues assumptions to study the profitability of the system.

Results show that, even with a greedy solution, it is possible to relocate in

a useful way. In particular, we can maintain low or even reduce the number of deployed e-scooter, thanks to *proactive* operations, that every hour assign to a given number of workers a list of relocations to be done. Such strategies have the capability of making the entire system more profitable and more sustainable, reaching, in our best case, 20% increase in terms of satisfied demand and more than 500k$ of profit gain compared to the same simulated scenario without relocation.

Acknowledgements

“ ”

Table of Contents

List of Tables VIII

List of Figures IX

Acronyms XIII

1 Introduction 1
- 1.1 The rising of micromobility . . . 1
- 1.2 What is an e-scooter? . . . 2
- 1.3 E-scooter sharing companies . . . 3
- 1.4 Side effects of e-scooter sharing boom . . . 3
- 1.5 In this thesis . . . 5

2 Background 6

3 Methodologies and Tools 8
- 3.1 Datasets . . . 8
- 3.2 Demand model . . . 11
- 3.3 Simulator and assumptions . . . 13
- 3.4 Relocation algorithms . . . 14
  - 3.4.1 Terminology . . . 14
  - 3.4.2 Strategies . . . 15
  - 3.4.3 Zone selection techniques . . . 15
  - 3.4.4 Performance metrics and cost assumptions . . . 16

4 Results 18
- 4.1 Magic relocation - Louisville . . . 18
- 4.2 Reactive relocation - Post charge - Louisville . . . 20
  - 4.2.1 With Aggregation and KDE sampling techniques . . . 20
  - 4.2.2 With Delta technique . . . 23
- 4.3 Reactive relocation - Post trip - Louisville . . . 27

4.3.1 With Aggregation and KDE sampling techniques 27
4.3.2 With Delta technique 28
4.4 Proactive relocation - Louisville 31
4.4.1 With Aggregation and KDE sampling techniques 31
4.4.2 With Delta technique 34
4.5 Proactive relocation - Minneapolis 38
4.5.1 With Aggregation and KDE sampling techniques 38
4.5.2 With Delta technique 39
4.6 Proactive relocation - Kansas City 43
4.6.1 With Aggregation and KDE sampling techniques 43
4.6.2 With Delta technique 44

5 Conclusions 48

Bibliography 51

List of Tables

3.1 E-scooter Trips Data Sources - Datasets and fleets specifications. . . 9

List of Figures

1.1 A Lime e-scooter [6]. 2
1.2 E-scooter handlebar with dashboard, QR code and power lever [7]. . 2
1.3 Cumulative funding of shared mobility unicorns over time [4]. . . . 3
1.4 Example of e-scooter sharing app. 4

3.1 Number of collected trips per month for each city. 10
3.2 Size of collected trips per month for each city. 10
3.3 Number of collected trips per month for each city in logarithmic scale. 11
3.4 Size of collected trips per month for each city in logarithmic scale. . 12

4.1 Satisfied Demand in Louisville with *magic relocation*. 19
4.2 Total number of relocations in Louisville with *magic relocation*. . . 19
4.3 Satisfied Demand in Louisville with *reactive relocation*, *post charge* enabled trigger and *KDE sampling* as ending zone selection technique. 21
4.4 Total number of relocations in Louisville with *reactive relocation*, *post charge* enabled trigger and *KDE sampling* as ending zone selection technique. 21
4.5 Satisfied Demand Difference in Louisville with *reactive relocation*, *post charge* enabled trigger and *KDE sampling* as ending zone selection technique. 22
4.6 Satisfied Demand in Louisville with *reactive relocation*, *post charge* enabled trigger and *Delta* as zone selection technique. 23
4.7 Total number of relocations in Louisville with *reactive relocation*, *post charge* enabled trigger and *Delta* as zone selection technique. . 24
4.8 Satisfied Demand Difference in Louisville with *reactive relocation*, *post charge* enabled trigger and *Delta* as zone selection technique. . 25
4.9 Profit Difference in Louisville with *reactive relocation*, *post charge* enabled trigger and *Delta* as zone selection technique. 25
4.10 Average number of vehicles moved per relocation in Louisville with *reactive relocation*, *post charge* enabled trigger and *Delta* as zone selection technique. 26

4.11 Satisfied Demand in Louisville with *reactive relocation*, *post trip* enabled trigger and *KDE sampling* as ending zone selection technique. 27
4.12 Total number of relocations in Louisville with *reactive relocation*, *post trip* enabled trigger and *KDE sampling* as ending zone selection technique. 28
4.13 Satisfied Demand in Louisville with *reactive relocation*, *post trip* enabled trigger and *Delta* as zone selection technique. 29
4.14 Total number of relocations in Louisville with *reactive relocation*, *post trip* enabled trigger and *Delta* as zone selection technique. . . . 29
4.15 Average number of vehicles moved per relocation in Louisville with *reactive relocation*, *post trip* enabled trigger and *Delta* as zone selection technique. 30
4.16 Satisfied Demand in Louisville with *proactive relocation* and *KDE sampling* as ending zone selection technique. 31
4.17 Total number of relocations in Louisville with *proactive relocation* and *KDE sampling* as ending zone selection technique. 32
4.18 Satisfied Demand Difference in Louisville with *proactive relocation* and *KDE sampling* as ending zone selection technique. 32
4.19 Profit Difference in Louisville with *proactive relocation* and *KDE sampling* as ending zone selection technique. 33
4.20 Satisfied Demand in Louisville with *proactive relocation* and *Delta* as zone selection technique. 34
4.21 Total number of relocations in Louisville with *proactive relocation* and *Delta* as zone selection technique. 35
4.22 Satisfied Demand Difference in Louisville with *proactive relocation* and *Delta* as zone selection technique. 35
4.23 Profit Difference in Louisville with *proactive relocation* and *Delta* as zone selection technique. 36
4.24 Average number of vehicles moved per relocation in Louisville with *proactive relocation* and *Delta* as zone selection technique. 37
4.25 Satisfied Demand in Minneapolis with *proactive relocation* and *KDE sampling* as ending zone selection technique. 38
4.26 Satisfied Demand Difference in Minneapolis with *proactive relocation* and *KDE sampling* as ending zone selection technique. 39
4.27 Profit Difference in Minneapolis with *proactive relocation* and *KDE sampling* as ending zone selection technique. 40
4.28 Satisfied Demand in Minneapolis with *proactive relocation* and *Delta* as zone selection technique. 40
4.29 Satisfied Demand Difference in Minneapolis with *proactive relocation* and *Delta* as zone selection technique. 41

4.30 Profit Difference in Minneapolis with *proactive relocation* and *Delta* as zone selection technique. 42
4.31 Average number of vehicles moved per relocation in Minneapolis with *proactive relocation* and *Delta* as zone selection technique. . . . 42
4.32 Satisfied Demand in Kansas City with *proactive relocation* and *KDE sampling* as ending zone selection technique. 43
4.33 Satisfied Demand Difference in Kansas City with *proactive relocation* and *KDE sampling* as ending zone selection technique. 44
4.34 Profit Difference in Kansas City with *proactive relocation* and *KDE sampling* as ending zone selection technique. 45
4.35 Satisfied Demand in Kansas City with *proactive relocation* and *Delta* as zone selection technique. 45
4.36 Satisfied Demand Difference in Kansas City with *proactive relocation* and *Delta* as zone selection technique. 46
4.37 Profit Difference in Kansas City with *proactive relocation* and *Delta* as zone selection technique. 47
4.38 Average number of vehicles moved per relocation in Kansas City with *proactive relocation* and *Delta* as zone selection technique. . . . 47

5.1 Satisfied Demand in Louisville with different relocation algorithms and $n_w = 1$. 49
5.2 Satisfied Demand in Louisville with different relocation algorithms and $n_w = 10$. 49

Acronyms

API Application Programming Interface

CNN Convolutional Neural Network

DRRP Dynamic Repositioning and Routing Problem

EDA Exploratory Data Analysis

FFVSS Free-Floating Vehicle Sharing Systems

GHG Greenhouse gases

KDE Kernel Density Estimation

LCA Life-Cycle Assessment

LSTM Long Short-Term Memory

MDS Mobility Data Specification

OMF Open Mobility Foundation

QR Quick Response

RNN Recurrent Neural Network

Chapter 1

Introduction

1.1 The rising of micromobility

In the past few years, the term micromobility became increasingly popular in the field of transportation. It usually refers to the usage of lightweight single-person vehicles for short trips [1].

This new way to travel has a great potential in terms of sustainability:

- it can reduce Green House Gases (GHG) emissions, by lowering the use of private cars. In particular, micromobility aims to provide a valid alternative for first-mile and last-mile trips and can also act as a complement to transit [2].

- It can ensure a reliable and equitable service, using sustainable business models and labor practices. It can also be a tool to provide equity through population, adopting measures like low-income fare and focused vehicle deployment over underserved geographical areas [3].

- It is able to enhance the human experience, being an enjoyable and fun way to travel. So, it can influence transport habits, increasing even more the shift from private car driving.

However, these potentials have not yet been fully expressed. New policies and planning actions are needed to make micromobility truly successful and sustainable, thus involving both private companies and regulators [2][4].

Bike and e-bikes were considered the micromobility mode par excellence, but, recently, a new vehicle is invading city streets: the e-scooter.

1.2 What is an e-scooter?

An electric kick scooter (e-scooter) is a stand-up scooter powered by a small electric motor (Figure 1.1). An initial "kick" is required to enable the motor and than the amount of power can be adjusted, typically with a small lever on the handlebar (Figure 1.2).

First shared e-scooters were consumer-grade vehicles, that were not designed for intensive use. For this reason, they used to have problems like insufficient range, and, consequently, relevant charging needs and long out-of-service times. Also, they suffered damage from being transported and left outdoors. In terms of life-cycle GHG emissions, one of the highest impacts came from materials and manufacturing [5], and an high demand of new e-scooter could have cast doubts on their effective sustainability.

Fortunately, e-scooter sharing companies invested on custom vehicles designs and now they are deploying more robust and durable e-scooters, with a grater range and swappable batteries [4]. However, we can assume that materials and manufacturing still represent a big contribution to total amount of emissions.

Figure 1.1: A Lime e-scooter [6].

Figure 1.2: E-scooter handlebar with dashboard, QR code and power lever [7].

1.3 E-scooter sharing companies

Recently, many companies are deploying e-scooters in cities throughout the world. They are offering shared dockless e-scooters as a new mode of micromobility and they are growing fast. In the last few years, e-scooter sharing operator giants, such as Bird and Lime, were capable to attract more than $1 billion of investment alone (Figure 1.3)[4].

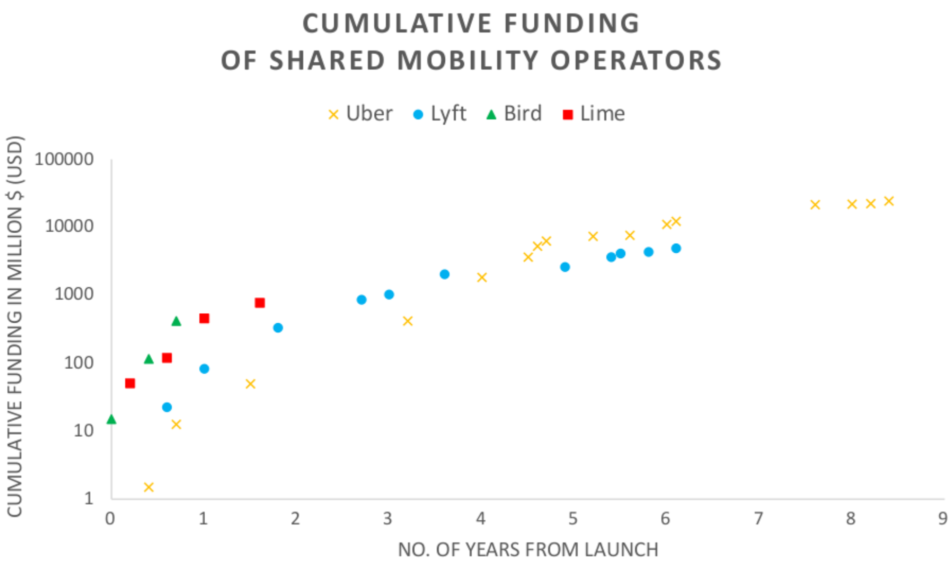

Figure 1.3: Cumulative funding of shared mobility unicorns over time [4].

The huge growth of these companies may also be due to the ease of use of their services. Renting a dockless shared e-scooter is as simple as a click on an app. Users must download a specific app on their smartphone, register, and select a payment method. In the main app view, a map shows where free e-scooters are located (Figure 1.4). When a user approaches a vehicle, he can start a new rental either directly from the app or scanning the QR code printed on the e-scooter, which uniquely identifies the vehicle (Figure 1.2).

1.4 Side effects of e-scooter sharing boom

As mentioned before, micromobility has great potentials, but, if not properly regulated, it can lead also to negative effects.

In the case of e-scooters, two big questions can be asked:

- Do e-scooters really reduce private motor vehicle use and traffic congestion?

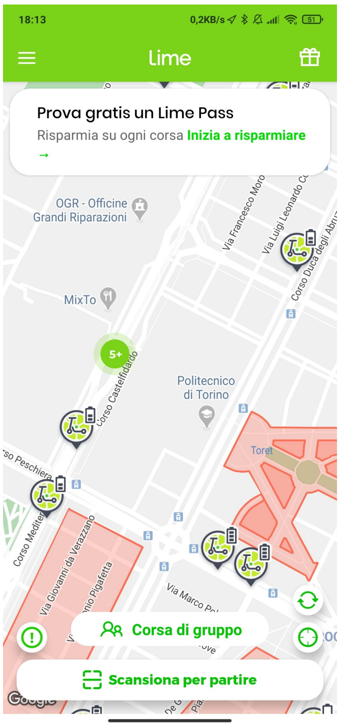

Figure 1.4: Example of e-scooter sharing app.

- Do they effectively contribute to GHG reduction?

We can say from surveys like [3] and other studies [2] that e-scooters are partially replacing personal cars and other transportation modes which implies the use of motorized vehicles (e.g., taxi, ride-haling). However, they require additional vehicles for distribution and relocation throughout the city. Hence, reducing transportation distance and using more efficient vehicles for this purpose, is mandatory to have a net reduction of GHG emissions [5]. Moreover, optimizing relocations can increase system utilization, thus reducing even more private motor vehicle use.

Considering congestion, if from one side we have less car traffic on streets, on the other hand we have a new vehicle populating streets and, more importantly, congesting also sidewalks. Pedestrian comfort and safety is reduced by e-scooter riding and parking in the middle of sidewalks. Both [3] and an analysis on big social data in [8] show that a big number of complaints is about vehicles being improperly parked. So, reducing the number of e-scooters in the city, can mitigate this issue. As a matter of fact, city regulators usually define a maximum number of deployable vehicles. Hence, try to utilize less e-scooter and try to maximise the utilization of each one of them, can be a reasonable goal for an e-scooter sharing company.

In addition, using less e-scooters can lead to a lower demand of new vehicles, thus reducing the big environmental impact given by materials and manufacturing, as mentioned in section 1.2.

1.5 In this thesis

In this work, we are going to analyze different relocation strategies, with the aim of understanding:

- if it is useful and profitable to relocate, and in which measure;
- how different relocation strategies affect an e-scooter sharing system.

For this purpose, we extended the software implemented in [9] - a data-driven, discrete-event simulator for Free-Floating Vehicle Sharing Systems (FFVSS) - to support relocation algorithms.

When possible, we updated the datasets that were already present in [9], and we introduced new public datasets, that were freely available online, reaching a total amount of 7 North-American cities.

From these datasets, we extract a demand model, that we can use as input for the simulator. Then, we generate a supply model, which defines the scenario in which the system will operate. This allows us to test the system in different conditions and to define parameters that give us the possibility to choose between different relocation strategies.

We collect a wide range of statistics from each simulation campaign, and we post-process them to compare different scenarios and to be able to analyze also costs and revenues of the system.

The thesis is organized as follows. In Chapter 2 we will discuss existing works on vehicle relocation techniques. In Chapter 3 we will do a brief EDA on newly integrated data and we will introduce the demand model, the simulator and the actual relocation strategies. In Chapter 4 we will deep dive into results obtained by simulations. Finally, in Chapter 5 we will sum up most relevant considerations emerging from results and we will introduce possible future work.

Chapter 2

Background

E-scooter sharing recently emerged as a new trending topic inside transportation research field. First findings and first collected data emerged from city pilots like [3] and from surveys like [10]. The first presents an analysis on data submitted by scooter companies through APIs, data collected by a customers survey, staff reports and customers focus groups observations. The latter is a survey on e-bike and e-scooter sharing, completed by Canadian stakeholders and municipal representatives, with the aim of analysing key perspectives for integrating micromobility within the existing transportation network. More recently [8] expanded the concept of surveys, by performing a big analysis on Twitter crawled data, and reported interesting indices, like trending topics, positive and negative feelings about e-scooter sharing systems, stakeholders characterization and system operations feedbacks.

Given the growth of such a new transportation mode, researchers tried to forecast e-scooter competition with other transport modes [11], and estimated that e-scooters would replace at most 32% of carpool, 13% of bike, and 7.2% of taxi trips.

Researchers also compared scooter-sharing with bike-sharing systems. In [12][4] and also in our previous work [9], it is possible to see that e-scooter usage temporal pattern is quite different to bike-sharing. Bike usage exhibits two distinct morning and evening peaks, suggesting primarily commuting usage. E-scooter riders, instead, are more likely to ride in the middle of the day and on weekends, suggesting social, shopping, and other recreational use.

Moreover, researchers questioned about actual sustainability and economic implications of current scooter-sharing systems and micromobility in general [4][13], also conducting a full LCA to address actual e-scooter environmental impact [5], and found that e-scooter manufacturing and deployment are the most contributing voices to GHG emissions.

Talking about vehicle deployment, relocation is a widely covered topic in the mobility research field. Although e-scooter is a relatively recent transport mode, we can find a lot of literature on relocation for car-sharing and also for bike-sharing, which is the most similar transport mode to e-scooters.

The first kind of approach is to schedule relocations at fixed times (e.g., at night), to rebalance the system [14]. Another possibility is to use time-independent decision rules that are only a function of the current system state [15].

Then, we can take in consideration the dinamicity of the system, introducing time dependency. This led to the use of online optimization approaches, to solve a newly defined problem called Dynamic Repositioning and Routing Problem (DRRP) [16]. There is a lot of existing work on this topic, and different formulations are proposed. However, these approaches, even the best performing ones, have a computation time in the order of minutes [17], and cannot be used in our scenario, in which we want to simulate potentially more than one month of system operations in a reasonable time.

Recently, an emerging topic in bike-sharing system optimization of bike relocation strategies, is the use of graph mining and deep learning. For example [18] uses a Convolutional Neural Network (CNN) to identify mobility patterns from unbalanced pair of stations - taking adjacency matrix snapshots of unbalanced sub-graphs -, and tries to predict future patterns through a Long Short-Term Memory Recurrent Neural Network (LSTM RNN).

In this thesis, we will approach to the DRRP without considering the relocation routing optimization part, and we will introduce greedy algorithms to be able to give a sub-optimal solution to the rebalancing problem in a reasonable computation time.

In our previous work [9], we proposed the methodology used in this thesis (Section 3.2) to translate open data describing e-scooter sharing trips into a demand model able to generalize their usage. Moreover, we found that, in order to satisfy the demand, we need a large number of e-scooters, and, in this thesis, we will introduce relocation operations to find out if it is possible to increase satisfied demand, or, possibly, to maintain the same amount of satisfied demand with a lower number of vehicles.

Chapter 3

Methodologies and Tools

In this chapter we will introduce the tools that we used and the strategy that we adopted. However, the most important tool was the coding language itself, and everything was implemented using *Python* [19], and libraries like *SimPy* [20], *Pandas* [21], *GeoPandas* [22] and *scikit-learn* [23].

3.1 Datasets

In this section, we are going to introduce a brief Exploratory Data Analysis (EDA) on the datasets that are now available and ready to be used with the simulator (see Table 3.1 for complete details of used datasets).

In [9] only data from the city of Minneapolis, MN and from the city of Louisville, KY where available. Now, we have more months for Louisville and we introduced 5 brand new cities: Austin, Norfolk, Kansas City and Chicago from USA and Calgary from Canada.

These municipalities make their data freely available online, using a standard API format called Mobility Data Specification (MDS), defined by the Open Mobility Foundation (OMF)[38]. In *Ref.* column of Table 3.1, a reference to the web page of each dataset is reported.

To avoid privacy issues and to not expose company practices, these datasets are anonymized and original raw data has been discretized both in terms of time and space. As we will see in Section 3.2, we need a continuous-time trace of events and coordinates, to be able to generate our demand model. So we disaggregated raw data as follows.

Starting and ending times of raw trips are quantized using different resolutions (see column *Time Res.* of Table 3.1). To provide an exact starting time estimation, to be used during the demand model estimation, we assume a local stationary process and we extract a timestamp from a uniform distribution. Given a Dataset

City	Months		N.Trips	Size (MB)	OD Type	Time Res.	Ref.
	from	to					
Louisville, KY	2018-08	2020-01	300 550	60.36	Rounded Coordinates	15 min	[24]
Louisville, KY	2020-02	2020-07	32 962	3.53	Rounded Coordinates	15 min	[25]
Minneapolis, MN	2019-05	2019-05	39 757	3.00	Centerline ID	30 min	[26]
Minneapolis, MN	2019-06	2019-06	123 696	11.65	Centerline ID	30 min	[27]
Minneapolis, MN	2019-07	2019-07	176 276	16.96	Centerline ID	30 min	[28]
Minneapolis, MN	2019-08	2019-08	223 729	21.55	Centerline ID	30 min	[29]
Minneapolis, MN	2019-09	2019-09	249 773	24.13	Centerline ID	30 min	[30]
Minneapolis, MN	2019-10	2019-10	177 853	17.13	Centerline ID	30 min	[31]
Minneapolis, MN	2019-11	2019-11	49 467	4.76	Centerline ID	30 min	[32]
Austin, TX	2018-04	2020-11	9 997 823	1894.19	Census Tract GEOID	15 min	[33]
Norfolk, VA	2019-07	2020-10	651 703	35.79	Census Tract GEOID	1 hour	[34]
Kansas City, MO	2019-06	2020-04	371 089	66.63	Rounded coordinates	15 min	[35]
Chicago, IL	2019-06	2019-10	710 839	183.25	Census Tract GEOID or Community Area Number	1 hour	[36]
Calgary, AB	2019-07	2019-09	482 021	106.53	Hexagonal grid	1 hour	[37]

Table 3.1: E-scooter Trips Data Sources - Datasets and fleets specifications.

$\mathcal{D}$ with a time granularity ΔT, and a discrete start time $a_s(i)$, the distribution will be like:

$$U\left[a_s(i) - \frac{\Delta T}{2}, a_s(i) + \frac{\Delta T}{2}\right] \tag{3.1}$$

For georeferencing, instead, we can see from column *OD Type* in Table 3.1, that different datasets use different methods to aggregate starting and ending locations of trips. Sometimes coordinates are simply rounded. For example, in the case of Louisville, they are rounded at 3 decimals - which corresponds to a resolution of 80 m - and they can be used as they are provided, because the resolution is still lower than the space granularity used for simulations (as we will see, our space bin will be 200 m wide). Sometimes, instead, we have a reference to a street or a big three-dimensional shape, which can be a simple geometrical shape - like an hexagon for Calgary - or a more complex figure - like Community Areas or Census Tracts. In all these other cases, we need to disaggregate the data, assuming a uniform distribution over the shape and extracting a point every time we need to define a starting or ending point of a trip.

Once normalized, cleaned from missing elements, and disaggregated, the data was organized by month, giving us the possibility to get a broad picture of how collected trips are distributed in time, both in terms of number of trips (Figure 3.1) and size (Figure 3.2).

We can clearly see how much Austin overcomes other cities, with a peek number

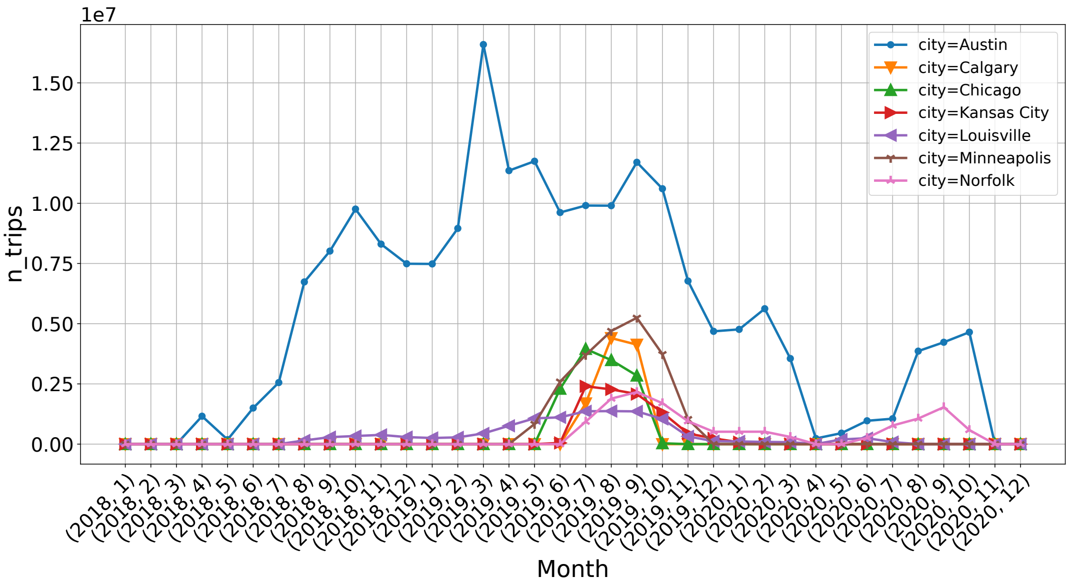

Figure 3.1: Number of collected trips per month for each city.

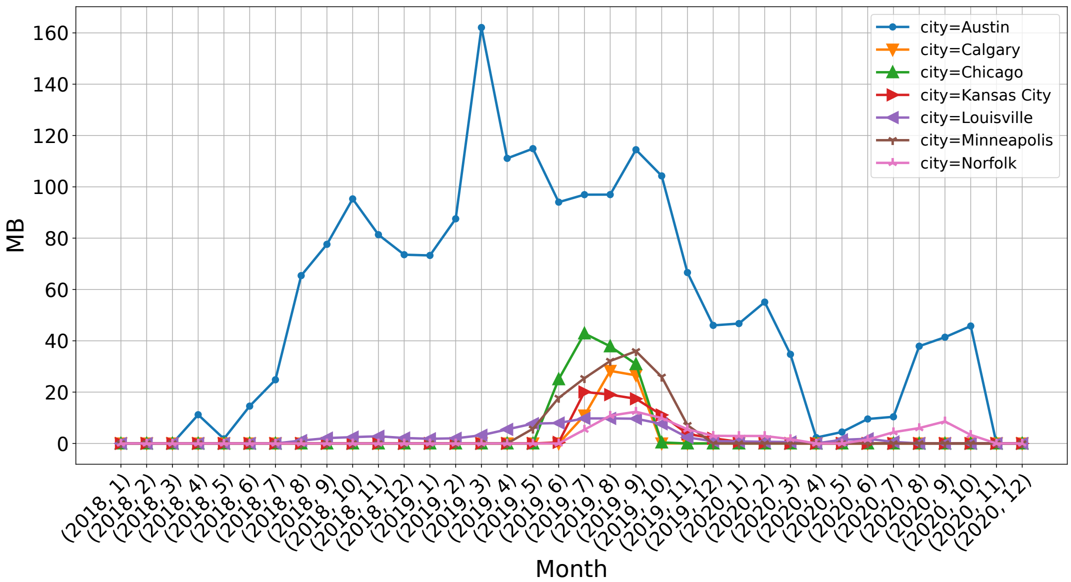

Figure 3.2: Size of collected trips per month for each city.

of trips per month of over 15 millions, proving to be the city with the highest demand between those selected. Louisville, instead, is at the opposite, also because is a smaller city. An interesting observation that we can made is that, regardless of the city, we have a massive drop of demand, starting during last months of 2019, and more or less prosecuting for all following months. This is obviously related to COVID-19 pandemic and restriction measures, that inevitably precluded the access to and reduced the demand of services like shared e-scooters.

In Figure 3.3 and 3.4, we plotted the same data in a different way. Using a logarithmic scale to color each month depending on the amount of present trips, we can clearly understand in which period of time collected data is placed, and how trips are distributed along these periods. We have to specify that March and April 2020, of Kansas City and Norfolk respectively, are not missing months, but they are available months with no registered trips. This shows even better how difficult the situation was for such companies and for the economy in general, during those months.

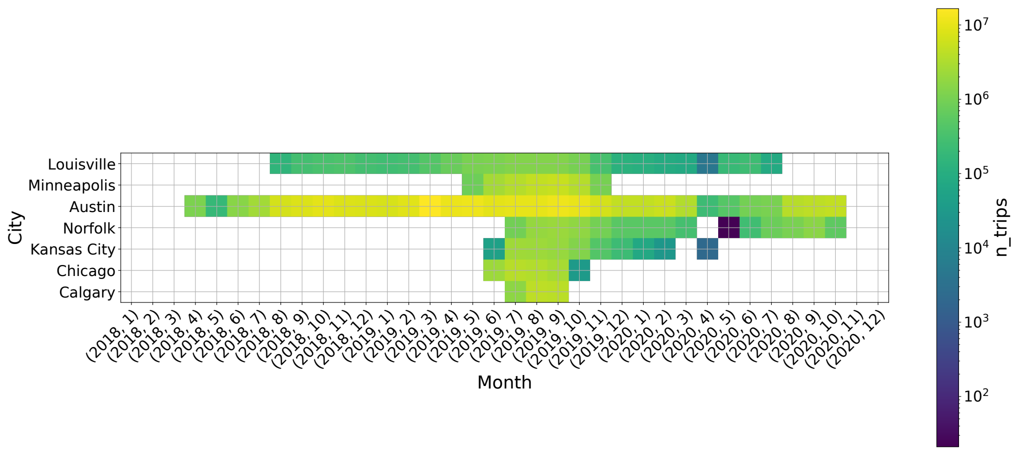

Figure 3.3: Number of collected trips per month for each city in logarithmic scale.

For the remaining part of the thesis, we will only focus on three cities. We chose Louisville and Minneapolis, to continue with the analysis started in [9], and we included also Kansas City, because of its similarity to Louisville, both in terms of demand and raw data granularity.

3.2 Demand model

Given pre-processed data as a trace, we extract a demand model, which then will be given as input to the simulator. The goal of the model is to generalize input data, giving us the possibility to generate new *synthetic* trips.

For this purpose, we extract a model based on two components: we use Poisson processes for time modeling and Kernel Density Estimation (KDE) for spatial

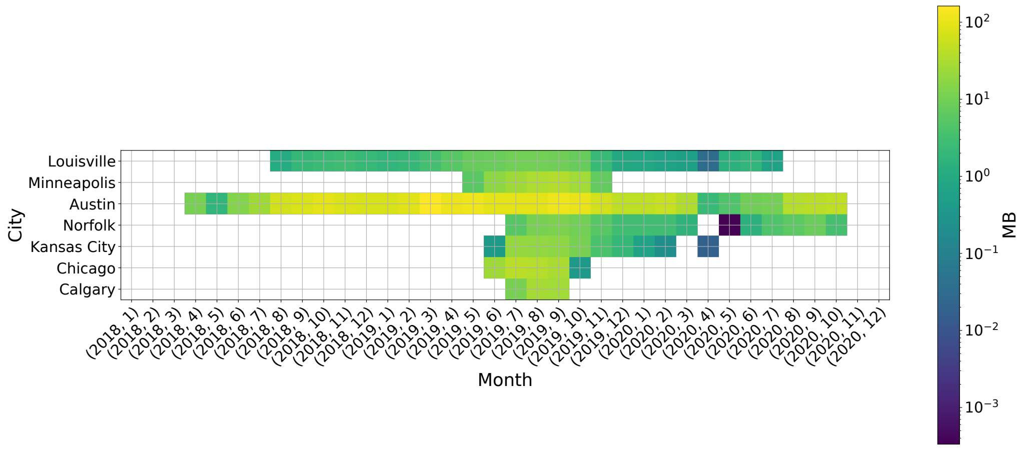

Figure 3.4: Size of collected trips per month for each city in logarithmic scale.

generalization, as in [9].

- Time model: Our time model is represented by hourly arrival rates. Given that e-scooter temporal pattern is a lot different between weekend and weekdays - as mentioned in Chapter 2 -, we define a total amount of 48 rates - 24 hours for both weekdays and weekends. We assume that the time between arrivals can be expressed by an exponential random variable. So, we fit the Poisson rate of each time slot to the average number of bookings occurring in the trace during the same period of time. The result is an inhomogeneous Poisson process, which is a commonly accepted model for independent service requests of a very large population [39].

- Spatial model: We generalize over space using KDE. For this purpose, we leverage Kernel Density estimator from *scikit-learn*[23], with a Gaussian kernel and a 2 x 2 identity matrix as bandwidth. First, we divide the city into 200 m x 200 m squares, generating a matrix of *city zones*. Then, we fit a four-dimensional KDE on origin-destination zone couples, where each zone is represented by its two indexes inside the matrix. We do this fitting 48 times, one for each time slot for weekdays and for weekends. Thus, we have a spatial representation of mobilty patterns between different zones during each hour of the day, for weekdays and for weekends.

For the purpose of this thesis, we will extract a demand model based on months from July to September 2019, because they are the only three months that are present for all cities (as can be seen from Figure 3.3).

3.3 Simulator and assumptions

We run our experiments with a discrete-event simulator, based on SimPy [20]. It receives two inputs:

- a configuration, or a set of configurations. It defines both general and running parameters. General parameters include the simulated city, simulated months, and the side length of zones (i.e., 200 m, in our case, as specified before). Running parameters, instead, define the scenario and the details about the vehicle sharing system, hence the supply model. They also specify relocation strategies and parameters.
- A demand model, as described in section 3.2.

Once the city is initialized, we have a fleet F of vehicles randomly distributed throughout the city, and the simulator starts generating new booking requests thanks to the demand model. Each new booking request is marked as *satisfied* if the user can find a vehicle in the request zone or in one of neighboring zones. The vehicle must have enough charge to support the entire trip. If the request is satisfied, the vehicle is considered *unavailable* for the entire simulated duration of the trip. Once the trip has finished and before the vehicle becomes *available* again, we check if recharging is needed.

For the purpose of this thesis, we assume that our e-scooter sharing system adopts custom made vehicles, as mentioned in section 1.2. So, even if our software gives us the possibility to simulate different charging strategies, we assume that:

- our only charging strategy is *battery swapping*;
- our e-scooters have a capacity $C = 425\,\mathrm{W\,h}$ and an energy efficiency of $k = 11\mathrm{W\,h\,km^{-1}}$;
- our system has an unlimited number of charging workers;
- each worker takes an average time of 30 minutes to reach an e-scooter and an average time of 5 minutes to swap the battery (this timings derive from previous simulations, presented in [9]).

Moreover, for all simulations that we did, we set 1 million requests per month as target. Given the average rate of booking requests extracted from the trace, the simulator computes a multiplicative factor before each simulation. It then multiplies the average rate at which simulated trips are yielded, by such multiplicative factor, with the aim of generating a total number of requests as near as possible to the selected target.

3.4 Relocation algorithms

In this section we will introduce the different algorithms that we implemented to manage e-scooter relocation.

They are divided in two categories: *reactive* and *proactive* approaches. With reactive strategies we decide if and how to relocate only when a certain condition is *triggered* during the simulation. With proactive strategies we decide how and when to relocate with a *schedule*.

3.4.1 Terminology

Before we describe the algorithms, we introduce some terminology:

- **Relocation strategy**: it defines the main structure of a relocation algorithm.

- **Zone selection technique**: it better characterises a relocation algorithm, defining which method is used for selecting starting and/or ending zone of a relocation. Sometimes, it is associated with parameters.

- **Relocation scheduling**: it is a recurrent procedure that every simulated hour generates a *relocation schedule*.

- **Relocation schedule**: it is a list of e-scooter relocations. Each item of this list includes both pick up and drop off zones and a proposed number of vehicles to be relocated. The maximum length of the list varies between different strategies. Its final length can be lower, because the zone selection technique can decide that a relocation between two zones should relocate zero vehicles, and, in that case, the relocation is not added to the list.

- **Relocation triggers**: they are points in the code where we decide if we have to start a relocation process or not. The decision is made by checking if current zone (i.e., the one in which the vehicle that triggered the check is currently located) is present inside the relocation schedule as a pick up zone. We have two *relocation triggers* inside the simulator: one is positioned right after the charging process finishes (**Post charge**) and one after a trip is completed, immediately before the vehicle is made available again (**Post trip**). This triggers are per-vehicle: the vehicle that triggered the relocation process is the first one that is relocated. If the relocation strategy allows more then one vehicles to be relocated at the same time, other random vehicles are picked up from the zone where the first vehicle triggered the process.

3.4.2 Strategies

Now, we can proceed with the description of actual algorithms. As reactive approaches we have:

- **Magic relocation**: we use it as an *upper bound* strategy, because theoretically is not possible to outperform it. So, it can be used as a measure of how large is the room for improvement and how well other strategies perform with respect to this margin. It works like that: for each booking request, if no e-scooters are available inside requesting zone and neighboring zones, a *magic* relocation is triggered. This means that surrounding zones are scanned, proceeding by concentric squares, and the e-scooter with the highest sufficient state of charge to satisfy the trip, in the innermost square, is instantly moved to the requested zone, without simulating the actual relocation trip.

- **Reactive relocation**: we use relocation triggers as *reaction* points, in which we check relocation needs. Hence, the decision criteria is the relocation schedule, that, in this case, we use as a list of reasonable relocation that can be done. Its maximum length, with this algorithm, is set to an upper bound represented by half the number of the total available zones in which the city is divided.

As the only proactive approach, we have:

- **Proactive relocation**, which simply every hour, performs *all* relocations given by the relocation scheduling, which, this time, acts as a list of relocations to be done. With this algorithm, the maximum length of such list is dynamically derived by the number of simulated workers that are free at the moment of list computation.

3.4.3 Zone selection techniques

It is possible to see that, for all the strategies (with the obvious exception of *magic relocation*) relocation scheduling is the only means by which we make our decision. In particular, the zone selection technique that we use to generate the schedule, plays a key role in the performance. So, here are the different techniques that we propose and compare:

- **Aggregation**: it selects as pick up (drop off) zone the one with maximum (minimum) number of vehicles.

- **KDE sampling**: it is used only to select drop off zones. It samples a new trip origin-destination couple from the *scikit-learn* [23] Kernel Density estimator mentioned in Section 3.2. More precisely, it uses the estimator of the next hour as a prediction model, and it selects as drop off zone the origin zone sampled from the estimator. This technique is very simple and effective, but, even if it is still stochastic, it uses as prediction model, the same model that we use to generate new trips.

- **Delta**: it is the most complex technique that we propose. It uses the number of current available vehicles in each zone, as a proxy for current state S. It uses average origin counts (O) and average destination counts (D), to calculate a prediction of the total outcoming or incoming flow of vehicles in a zone at a given hour of a given type of day (i.e., weekday or weekend). O and D are derived directly from the trace, computing the average out-flow and in-flow of vehicles from a zone at a given hour of a given day type, and they are used as a prediction model. The predicted flow at hour i of day of type d for zone z is then computed as the difference between $O(d, i; z)$ and $D(d, i; z)$. A positive flow means that the predicted number of vehicles that will depart from a zone at a given hour, will be higher than the predicted number of vehicles that will arrive. The strategy selects as pick up (drop off) zone the one with the lowest negative (highest positive) *delta* (Δ), which is the difference between predicted flow for next hour(s) and current state, for a given time t (in hours) and a given zone z, as can be seen in (3.2). Thus, an higher delta means that a shortage of e-scooters is more probable. For example, it can mean that we predict an high positive out-coming flow from a zone, and we know from S that there are not enough e-scooters.

$$\Delta(d, t; z) = \frac{\sum_{i=t+1}^{t+W} O(d, i; z) - D(d, i; z)}{W} - S(d, t; z) \tag{3.2}$$

  This is the only strategy for which we can specify a window width W, to be able to take into consideration more then just one hour in the next future.

  This is also the only strategy that allows us to relocate more than one e-scooter at a time, with a suggested number of relocated vehicles that is given by Δ itself.

3.4.4 Performance metrics and cost assumptions

To evaluate our algorithms, we will consider two performance metrics:

- **Satisfied Demand Difference**: it is the difference in terms of satisfied demand between a system operating with given algorithm and the same system with no active relocation operations.

- **Profit Difference**: similarly to the previous one, it is the difference in terms of profit between a simulated scenario with a relocation algorithm and the same scenario without relocation. However, we are not considering the actual profit of an e-scooter sharing operator, because in that case we should consider also costs that are idependent from relocation. We consider as *profit* the difference between total revenues and relocation costs, and than we compute the difference between the *profit* of the strategy and the *profit* without relocation. Such operation allows us to take in consideration both performance benefits and side effects in terms of costs of doing relocation.

To be able to define system costs and revenues, we made these assumption:

- a relocation worker cost of 18 $ per hour,
- a relocation van with a leasing cost of 400 $ per month and a consumption of 7 l/100km,
- a diesel price in USA of 0.65 $/l,
- a fixed e-scooter unlock fee of 1 $,
- and an e-scooter rent fee of 0.30 $/min.

Chapter 4

Results

In this Chapter, we will discuss about obtained results for three different cities: Louisville, Minneapolis and Kansas City. At first, we will discuss about how different algorithms performed in the case of Louisville, and then we will see how the best performing ones behave with the other cities.

As mentioned in Chapter 3 we will use months from July to September 2019 to generate our model. Then we simulate one month of operations and analyze results.

4.1 Magic relocation - Louisville

At first, we present results obtained by *magic relocation* strategy (Figures 4.1-4.2).

In Figure 4.1, we plotted both percentage patterns of satisfied demand with *magic relocation* and with no relocation at all. On the X axis we have the fleet size, which corresponds to the cardinality of the F set (i.e., $|F|$), which is the set of simulated vehicles. We can see from the graph that, with no relocation, even with an high number of deployed vehicles (i.e., 2000) we can reach at most 70% of satisfied demand. Hence, there is a big margin for improvement. On the other hand, as expected, *magic relocation* is always capable to satisfy a big portion of the demand, except for an initial transient in which the number of deployed vehicles is too low. This is because a minimum number of deployed e-scooters is needed to satisfy all booking requests, although we are magically putting the vehicle in front of the user every time it is needed.

In next figures, when we will plot satisfied demand for other relocation algorithms, we will always use these two lines as a reference, to understand how much the considered algorithm is performing well, with respect to not having relocation, and with respect to the upper bound represented by *magic relocation*.

In Figure 4.2, we plotted the total number of relocation operations performed,

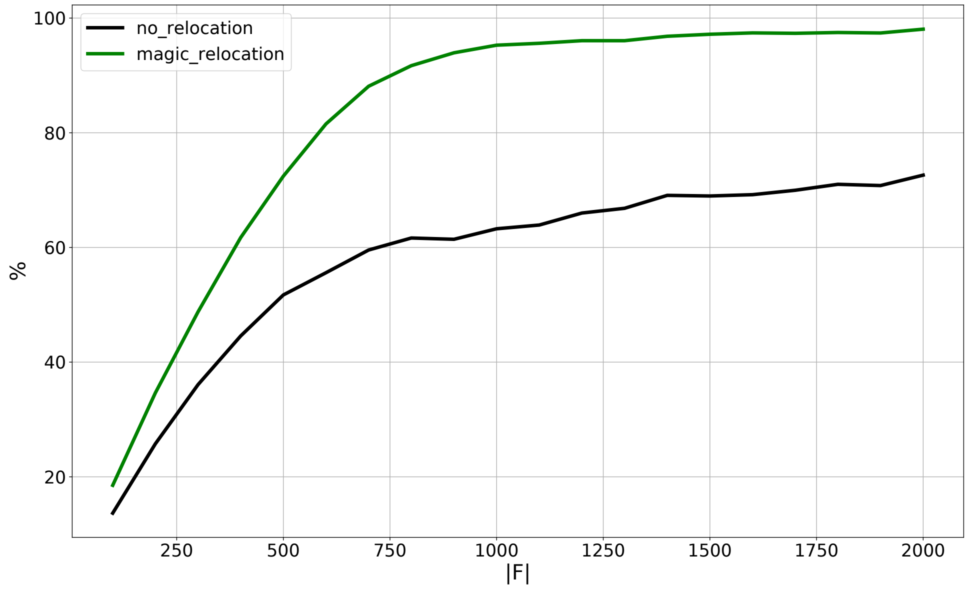

Figure 4.1: Satisfied Demand in Louisville with *magic relocation*.

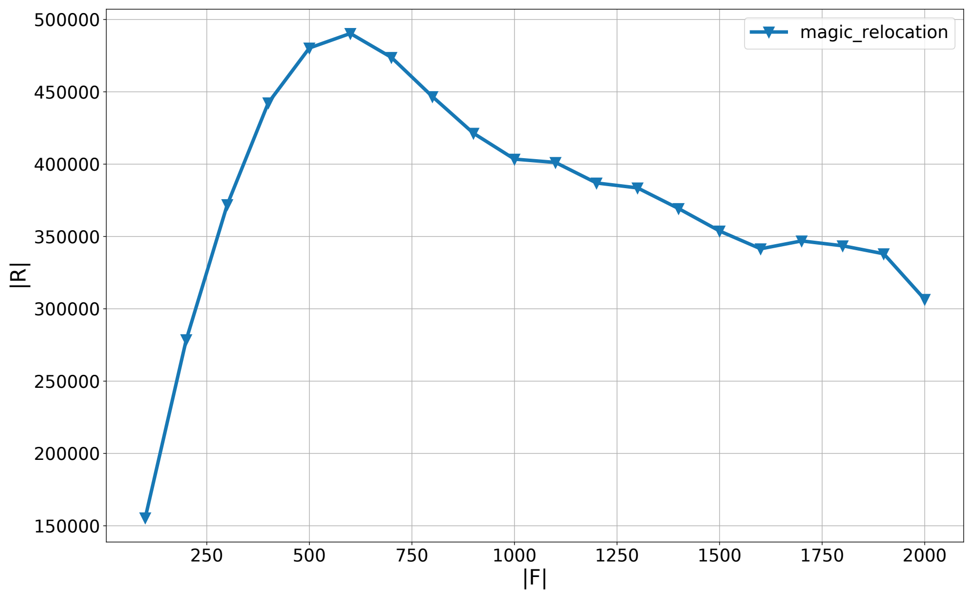

Figure 4.2: Total number of relocations in Louisville with *magic relocation*.

depending on the total number of vehicles deployed. Each operation produces a tuple with relocation details, that includes: relocation date, start time and end time, number of vehicles moved, pick up and drop off zone ids, traveled distance and process duration. We call R the set containing all relocation tuples. So, the cardinality of R represents the total number of relocation operations.

With *magic relocation*, this number is not so relevant per se, but it can be used to have an idea of the order of magnitude of the number of relocations that is needed to satisfy the entire demand with such a strategy.

As expected, the number of relocations is higher with less vehicles, because more relocations are needed to satisfy the demand, which is spreaded throughout the city. The initial steep ascending part of the graph, is due to the fact that here we are not satisfying the entire demand, so, most of the times in which the system receives a booking request, a *magic* relocation is needed, but the final number of relocations still depends on the number of available vehicles. In other words, in this section, with more vehicles we do more relocations to satisfy the same demand, which is not yet fully satisfied.

4.2 Reactive relocation - Post charge - Louisville

In this section, we present results obtained from a first version of the *reactive relocation*, with *post charge* as the only one enabled trigger. This means that, after each charge, we check the *relocation schedule*: if there is a proposed relocation in the *relocation schedule* that has as pick up zone the same zone where the vehicle has just finished charging, we perform such relocation. If we planned to relocate more than one e-scooter, we relocate the one that triggered the action and we randomly pick up the remaining number of e-scooters from the set of vehicles that are currently available in that zone.

4.2.1 With Aggregation and KDE sampling techniques

At first, we analyze how the *reactive relocation* algorithm performs with *aggregation* as pick up zone selection technique and *KDE sampling* as drop off zone selection technique (Figures 4.3-4.5).

We can see from Figure 4.3 that this version of the algorithm is not performing so well, and, most importantly, it only outperforms the scenario with no relocation when we set the number of workers to 10, which is the maximum value that we considered. Its performance is almost proportional to the number of relocation workers and slightly dependent on the number of deployed vehicles.

As we can see from Figure 4.4, the number of performed relocations has a pretty similar behaviour. Hence, we can say that: there is a limit to the number of relocation that can be performed in a month by a certain number of workers; the performance of the system is strictly related to the number of relocation that we perform; given that with these zone selection techniques we are always moving

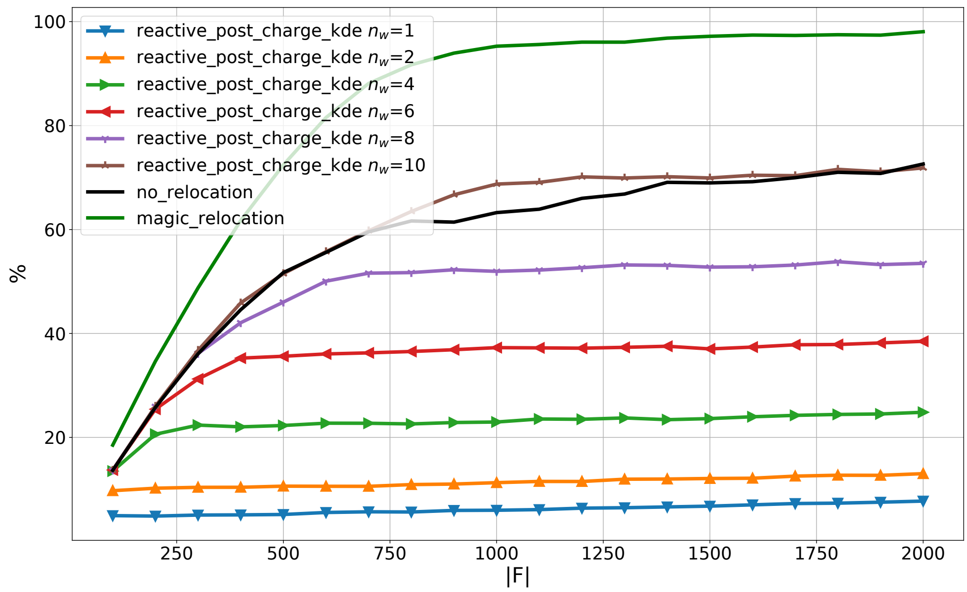

Figure 4.3: Satisfied Demand in Louisville with *reactive relocation*, *post charge* enabled trigger and *KDE sampling* as ending zone selection technique.

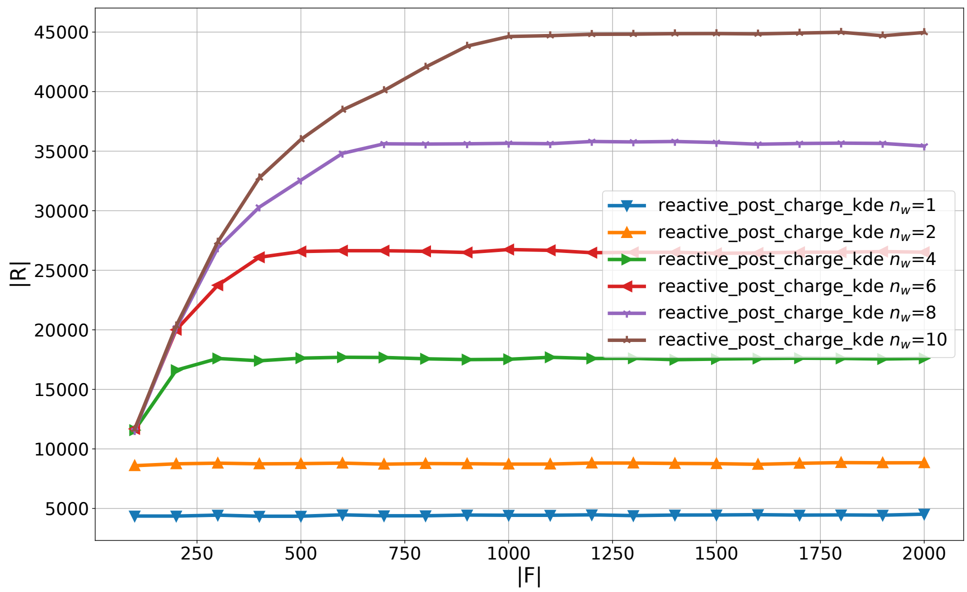

Figure 4.4: Total number of relocations in Louisville with *reactive relocation*, *post charge* enabled trigger and *KDE sampling* as ending zone selection technique.

only one vehicle for each relocation, there is a minimum number of relocations required to obtain a better performance than without relocation, and such number of reloactions can be reached only with a high number of workers.

In Figure 4.5, we present a contour plot of our first performance metric applied to this algorithm. We show how the metric is affected by different fleet sizes and different numbers of relocation workers at the same time. We will always use the same colour scale for plots of this kind through the entire Chapter, in order to easily do comparisons between different algorithms.

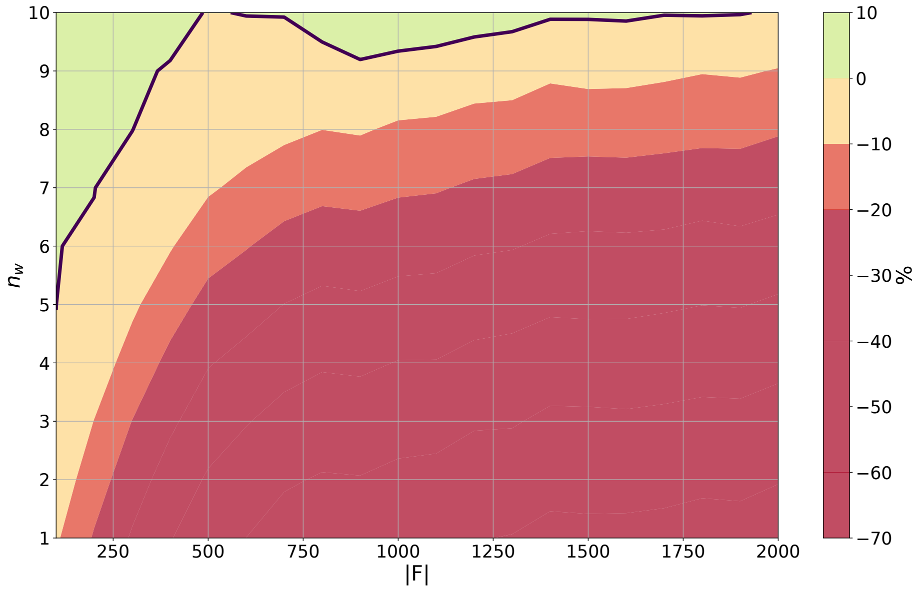

Figure 4.5: Satisfied Demand Difference in Louisville with *reactive relocation*, *post charge* enabled trigger and *KDE sampling* as ending zone selection technique.

We can see that, accordingly to the previous plots, the percentage difference with respect to not having relocation, is positive only with a high number of workers, with an interesting improvement when having less than 10 workers, but very few vehicles. Although this kind of behaviour with a very small fleet can be promising, looking again at Figure 4.3, we can see that, even if we are improving satisfied demand, we are in a range in which even magic relocation cannot do better than 19%. Moreover, when we consider costs and revenues, with the help of our *profit difference* metric, we found that we were always making a lower profit than without relocation.

Finally, we can conclude that this algorithm is not improving vehicle utilization and it is also not profitable for the e-scooter sharing company. Moreover, using *KDE sampling* technique, we are always moving only one vehicle per relocation, and this can be not the best possible solution.

4.2.2 With Delta technique

We can move now to the analysis of the same algorithm, with a different pick up and drop off zone selection technique, which is *Delta* technique (Figures 4.6-4.10).

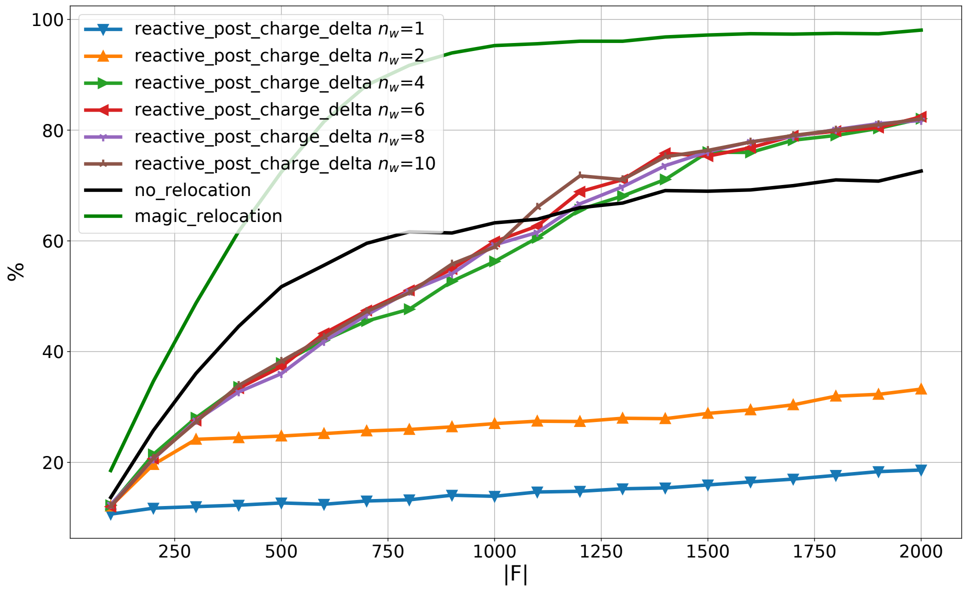

Figure 4.6: Satisfied Demand in Louisville with *reactive relocation*, *post charge* enabled trigger and *Delta* as zone selection technique.

As we can see in Figure 4.6, this version of the algorithm is performing well only with certain conditions. In particular, we can observe that the behaviour in terms of satisfied demand is pretty similar when considering any number of workers grater than or equal to 4. This is due to the fact that *Delta* technique proposes a limited number of relocations, which depends on the number of possible combinations between zones with positive *Delta* (i.e., more scooters than needed, see Section 3.4.3) and zones with negative *Delta*. Hence, we can say that 4 is the minimum number of workers needed to perform all relocations suggested by *Delta* technique, and we can also say that an higher number of workers is not needed and does not imply any changes in system performance.

In Figure 4.7, we have another confirmation of our deductions. With 1 or 2 workers, the number of performed relocations is limited by the capabilities of each worker, as it was for the previous algorithm (Figure 4.4). With an higher number of workers, the number of operations is limited by the algorithm itself: it increases

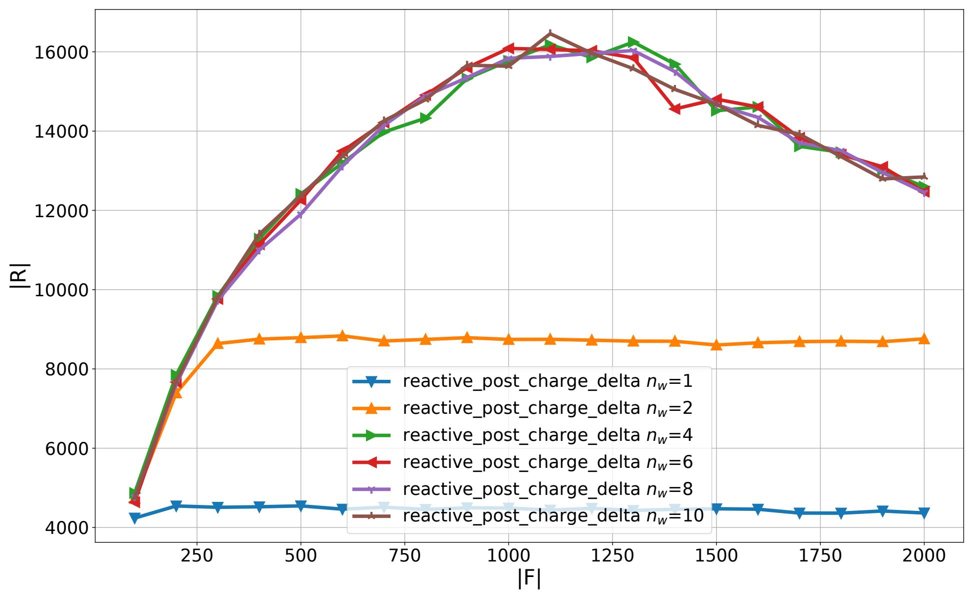

Figure 4.7: Total number of relocations in Louisville with *reactive relocation*, *post charge* enabled trigger and *Delta* as zone selection technique.

when the fleet is small, because there are increasingly more vehicles that need relocation, and it decreases when the number of simulated vehicles is high, because there are more and more vehicles that already satisfy the demand, without requiring relocation. This generates a concave trend in which the maximum is around 1100 vehicles, and, interestingly, corresponds to the point at which the algorithm starts working better than without relocation.

As expected, the *satisfied demand difference* (Figure 4.8) is in line with previous observations, and shows an important improvement area with a fleet size bigger than about 1100 vehicles and with more than 4 workers.

Also, when considering costs and revenues (Figure 4.9), we can see that there is a similar area of gain with respect to no relocation, but the tipping line is slightly shifted to the right, at about 1400 vehicles. This means that, for a fleet size between 1100 and 1400, with our cost assumptions, even if we are improving performance, we are profiting less than without relocation. In a real case scenario, it would be up to the e-scooter sharing company to evaluate if such a negative difference could be compensated by induced demand given by a better performing system. Even if further discussion can be made on this topic, it is out of the scope of this thesis.

With algorithms that use *Delta* as zone selection technique, we can analyze another important measure, that is the number of vehicles moved for each relocation

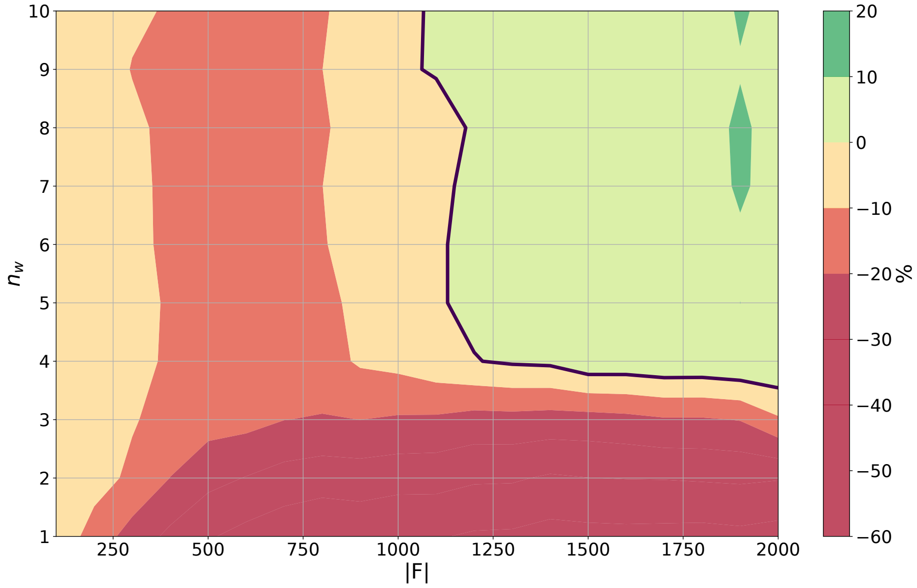

Figure 4.8: Satisfied Demand Difference in Louisville with *reactive relocation*, *post charge* enabled trigger and *Delta* as zone selection technique.

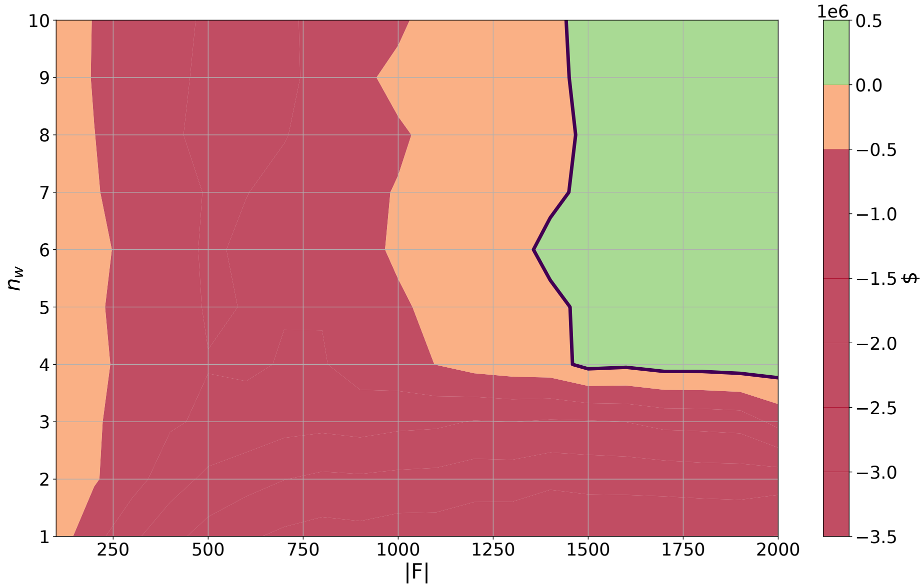

Figure 4.9: Profit Difference in Louisville with *reactive relocation*, *post charge* enabled trigger and *Delta* as zone selection technique.

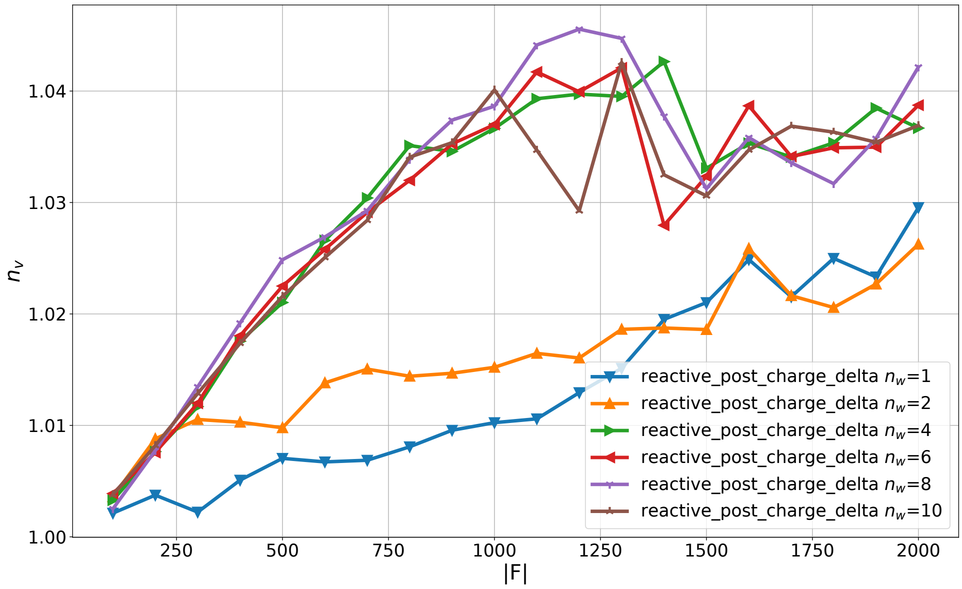

Figure 4.10: Average number of vehicles moved per relocation in Louisville with *reactive relocation*, *post charge* enabled trigger and *Delta* as zone selection technique.

(Figure 4.10). This measure is peculiar to small, light-weight vehicles such as e-scooters, and gives us the possibility to understand how much we are exploiting the capability to relocate more then one vehicle at a time. For example, with this algorithm, we can see that we always relocate about one vehicle on average per relocation. This means that probably we are not yet completely leveraging an usual van loading capability. However, we can use this information only in a qualitative way, because we are neither simulating nor optimizing relocation routing, and we do not have a real measure of how much we are loading a truck per relocation trip.

4.3 Reactive relocation - Post trip - Louisville

In this section, we present results obtained from a second version of the *reactive relocation*, with, this time, *post trip* as the only one enabled trigger. The purpose of this version is to understand how the algorithm behaves when a more frequent event, such as the end of a trip, is used to trigger relocation check.

4.3.1 With Aggregation and KDE sampling techniques

Also this time, we will try different zone selection techniques.

In this section, we start with *aggregation* as pick up zone and *KDE sampling* as drop off zone selection techniques (Figures 4.11-4.12).

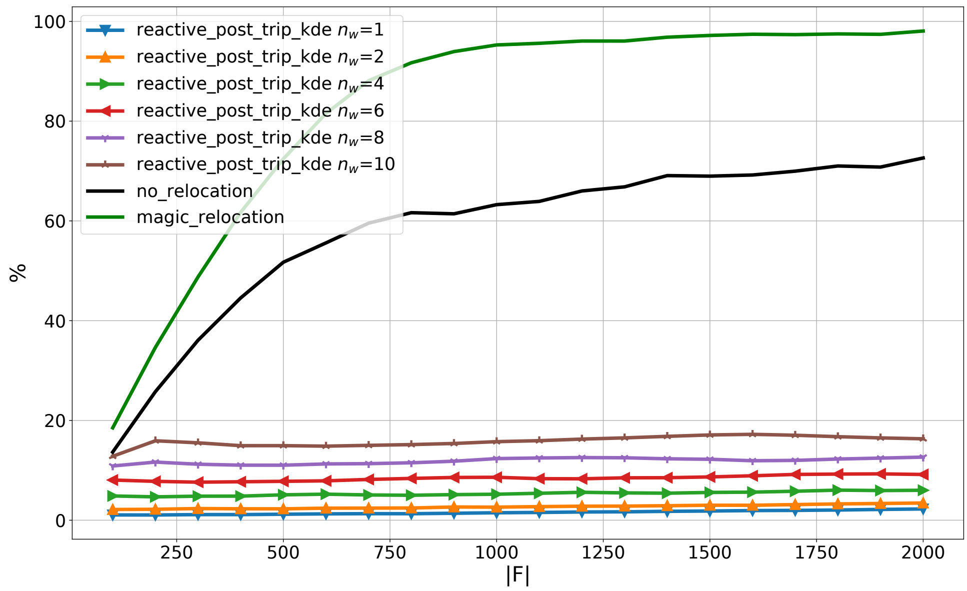

Figure 4.11: Satisfied Demand in Louisville with *reactive relocation*, *post trip* enabled trigger and *KDE sampling* as ending zone selection technique.

As we can see from Figure 4.11, this algorithm is not performing well at all, thus meaning that, probably, a too much frequent relocation check is counterproductive. Indeed, it is possible that we are moving e-scooter too much, randomly spreading them throughout the city, instead of relocating them in a reasonable way. We can see from Figure 4.12 that we are always saturating the capability of workers of doing relocations in parallel. So, the problem is not simply related to the number of relocation performed, but it is also related to the fact that we check relocation needs too many times, each time doing a relocation to a drop off zone that was previously computed by the KDE. This means that we are not sure that we are always doing only best (i.e., most needed) relocations. As a reminder, any time a relocation check is triggered, we start the relocation process, only if current position of the e-scooter that enabled the trigger, is present as a pick up zone for some relocation. So, it is possible that doing relocations depending on where each trip ends, is not the best way to decide which relocations should be performed.

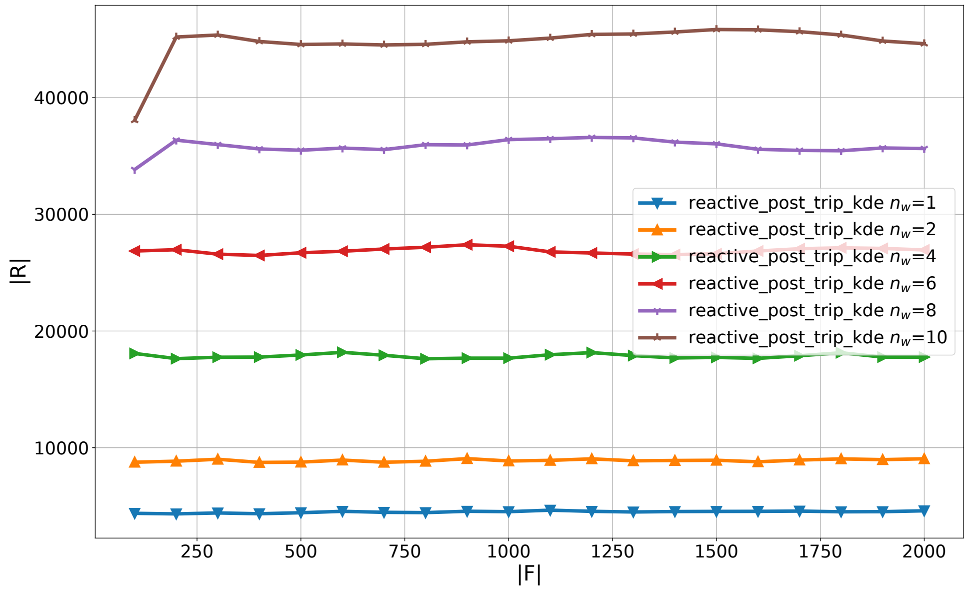

Figure 4.12: Total number of relocations in Louisville with *reactive relocation*, *post trip* enabled trigger and *KDE sampling* as ending zone selection technique.

4.3.2 With Delta technique

We now consider *delta* as zone selection technique (Figures 4.13-4.15), but also in this version, our *reactive relocation* does not have success with the *post trip* trigger.

As we can see from Figure 4.13, we are never outperforming simulations without relocation. However, with respect to previous version, we obtained an improvement, especially when we added more workers. This means that in general we obtain a better performance with *delta* technique, even when the strategy itself is not working so well.

Regarding the total number of relocations (Figure 4.14), when we simulated 1 or 2 workers, we always saturated their capabilities, but from 3 to 10, we obtained a slightly different behaviour, in which at first the number of operations grows with the number of vehicles, and then it saturates either to worker capability or to the

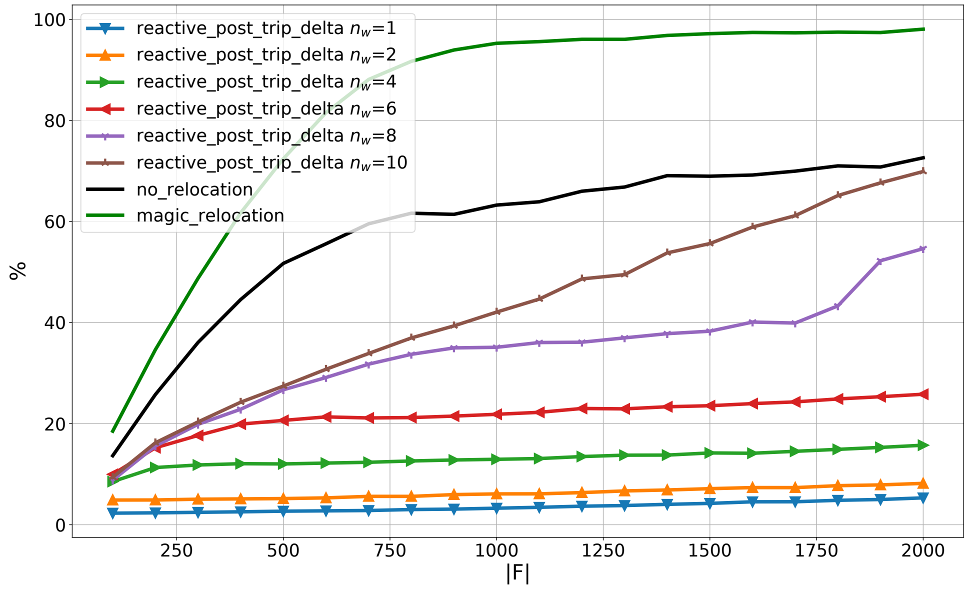

Figure 4.13: Satisfied Demand in Louisville with *reactive relocation*, *post trip* enabled trigger and *Delta* as zone selection technique.

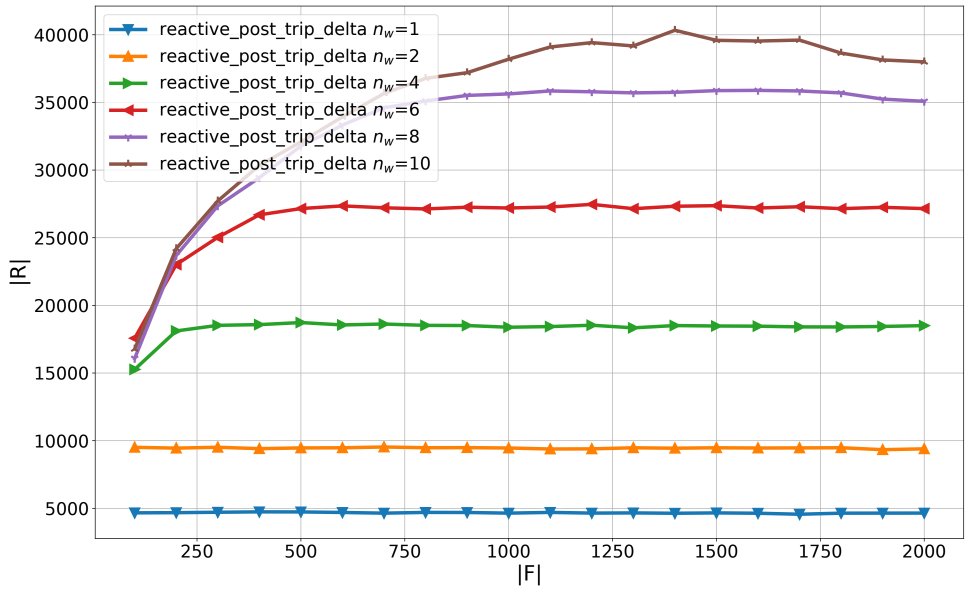

Figure 4.14: Total number of relocations in Louisville with *reactive relocation*, *post trip* enabled trigger and *Delta* as zone selection technique.

total number of relocations, as they are proposed by *delta*. The best performance is obtained when most of proposed relocations are performed, with $n_w = 10$. This means that, as mentioned before, given our implementation of reactive strategy, it is counterproductive to use a too frequently triggered check to perform relocation. It can be more useful, instead, to get the top n needed relocations from *delta* and perform them in a proactive way, as we will see in next Sections.

In Figure 4.15, we reported the average number of vehicles moved for each relocation. We can see that, as it was with *post charge* trigger, we are not yet completely leveraging the van loading capability, with an average number of vehicles that is always about one.

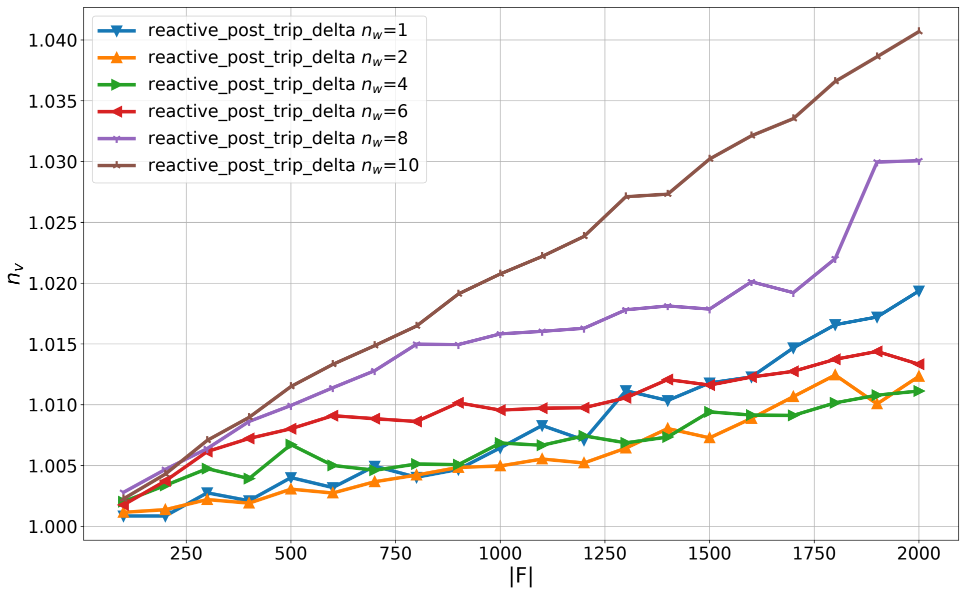

Figure 4.15: Average number of vehicles moved per relocation in Louisville with *reactive relocation*, *post trip* enabled trigger and *Delta* as zone selection technique.

4.4 Proactive relocation - Louisville

In this section, we are going to analyze how *proactive relocation* strategy performs.

4.4.1 With Aggregation and KDE sampling techniques

As for previous algorithms, we analyze at first how the strategy performs with *aggregation* and *KDE sampling* as pick up and drop off zone selection techniques respectively (Figures 4.16-4.19).

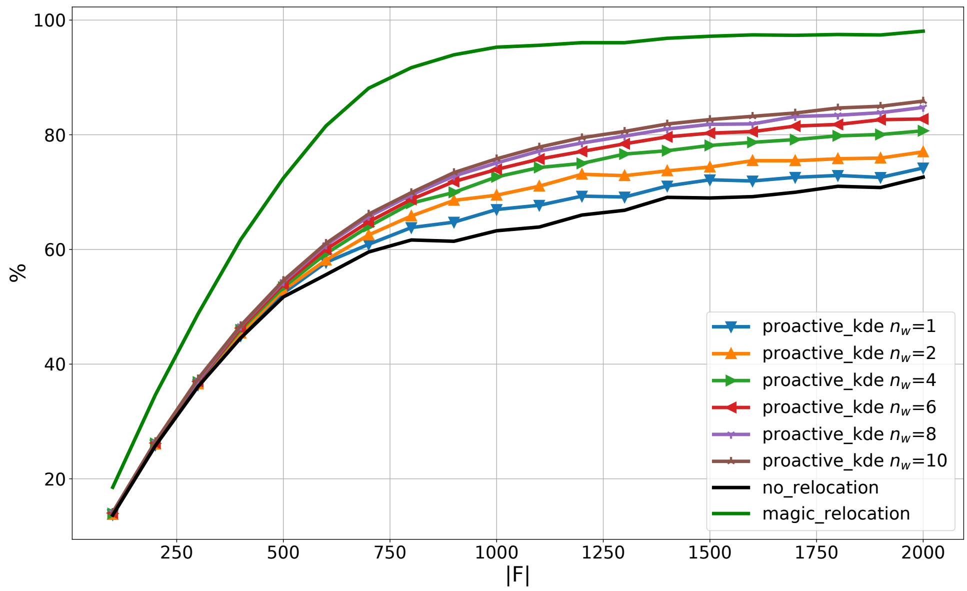

Figure 4.16: Satisfied Demand in Louisville with *proactive relocation* and *KDE sampling* as ending zone selection technique.

We can see from Figure 4.16 that this strategy is performing quite well, with a best improvement of about 15% on satisfied demand when considering the highest amount of workers that we tested, and of about 5% when considering the smallest.

As a proactive approach, we run a fixed number of relocations for each hour, resulting in a constant total amount of relocations that only depends on the number of workers, that can relocate vehicles in parallel (Figure 4.17).

As we can see from Figure 4.18, the *satisfied demand difference* with respect to not having relocation is already quite impressive. It increases with the number of workers and the fleet size. It is always positive, meaning that, at worst, the algorithm has the same performance than without relocation. The maximum achieved *satisfied demand difference* is about 16%. However, from our perspective, the most important

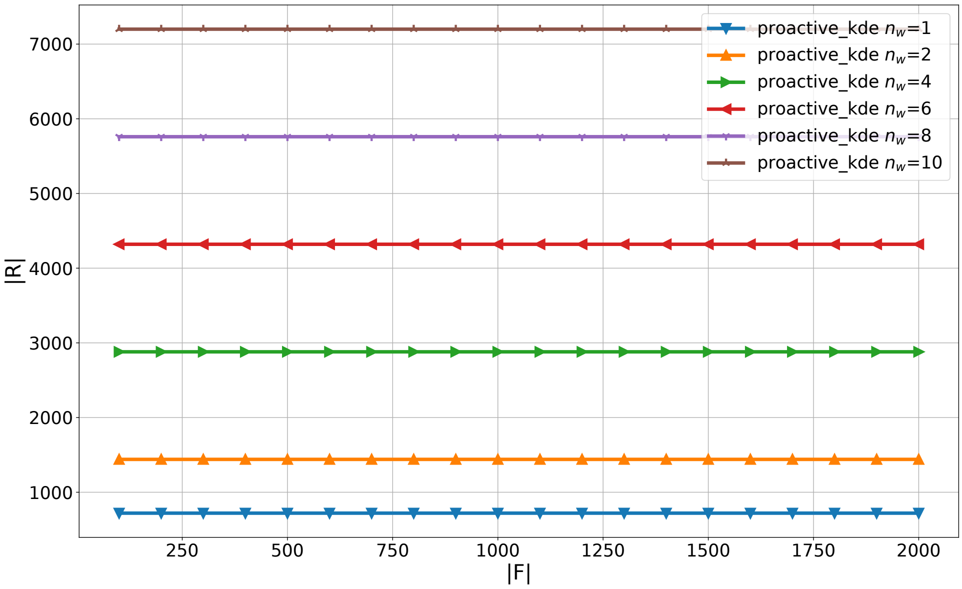

Figure 4.17: Total number of relocations in Louisville with *proactive relocation* and *KDE sampling* as ending zone selection technique.

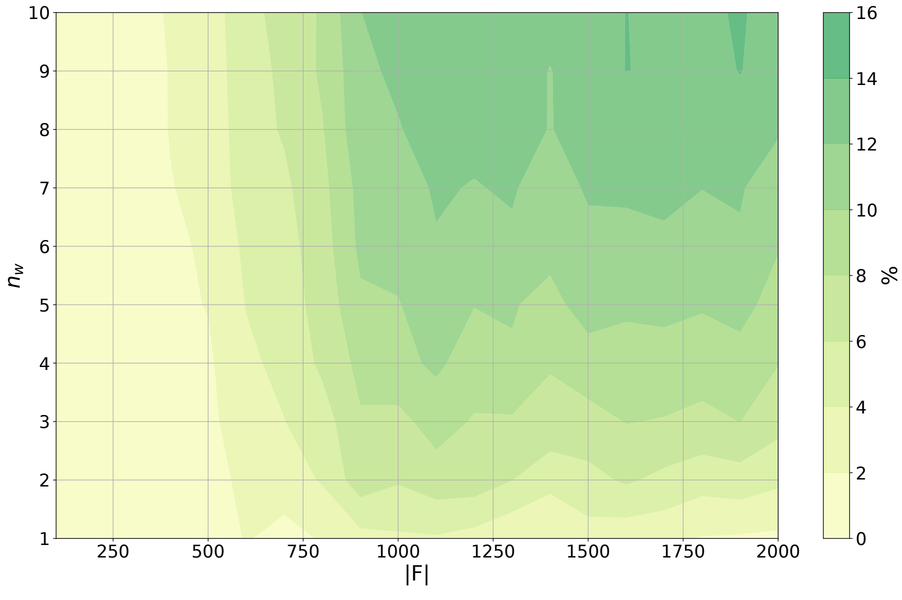

Figure 4.18: Satisfied Demand Difference in Louisville with *proactive relocation* and *KDE sampling* as ending zone selection technique.

achievement is that this algorithm can improve system performance even with an amount of vehicles that is less or equal than the number of vehicles actually deployed in the city (i.e., 850). This means that we can satisfy the same demand with less vehicles, thus reducing vehicles production and deployment.

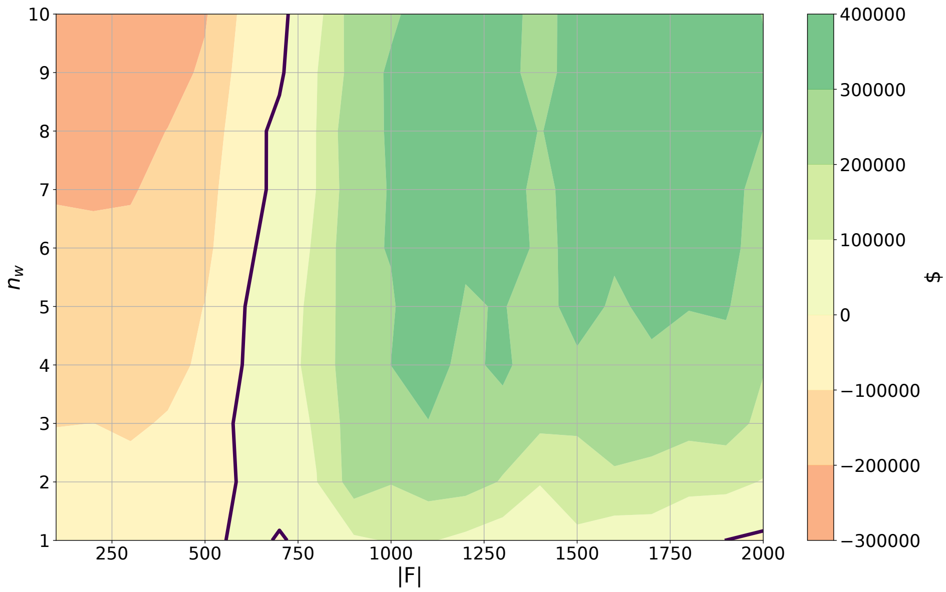

Figure 4.19: Profit Difference in Louisville with *proactive relocation* and *KDE sampling* as ending zone selection technique.

In Figure 4.19, we can see that, also in terms of *profit difference*, this version of the *proactive relocation* is already performing well, reaching a total difference in profit of about 400k$ more than without relocation.

Differently from *satisfied demand difference*, here we have both zones with positive and negative differences, with a well defined boundary at about 600 vehicles. This means that a trade-off should be made between satisfied demand gain, profit difference and number of deployed vehicles, also keeping in consideration ecological aspects and limits imposed by regulators. In any case, from our point of view, it is interesting to see that, given our 1 million simulated requests target, also with a fleet size equal or slightly lower than the real world fleet, it is possible to make a slightly higher profit than without relocation.

However, we are still moving one vehicle per relocation. This is not optimal, and can be solved applying *delta* technique to this algorithm, as it is shown in next section.

4.4.2 With Delta technique

We can finally present our best performing algorithm, which is *proactive relocation* with *delta* as both zone selection techniques (Figures 4.20-4.24).

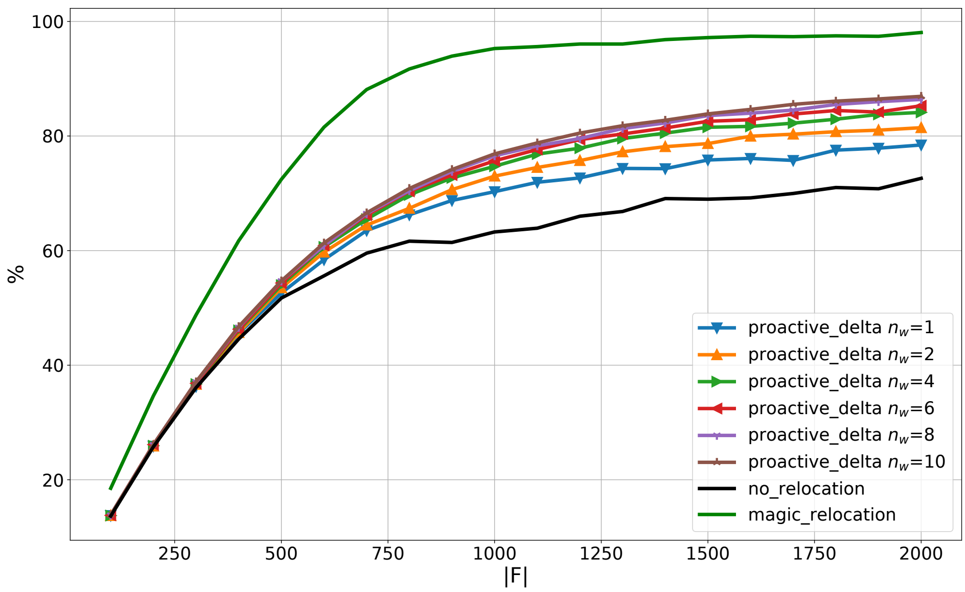

Figure 4.20: Satisfied Demand in Louisville with *proactive relocation* and *Delta* as zone selection technique.

As we can see from Figure 4.20, this algorithm, at its best, is capable of raising the satisfied demand from about 70%, without relocation, to about 85%. Similarly to the previous one, it works increasingly better with a bigger fleet and an higher number of workers (Figure 4.22). However, we can clearly see from Figure 4.20 that with a constant step increase in the number of workers, the satisfied demand does not increase constantly as well, meaning that there is limited room for improvement. Moreover, the biggest difference here, is that, even with only one worker we have a leap of improvement of about 8%. In other words, there is already a big gap between performing relocation with a single worker and not doing relocation at all.

We can see from Figure 4.21 that, similarly to the version with *KDE sampling*, we have a constant total number of relocations which depends only on the number of workers, as expected.

Given our costs and revenues assumptions, we can see from Figure 4.23 that the behavior of this version of the algorithm is very similar to the previous one. However, here we can reach an even higher *profit difference* of about 500k$ more. We also have a similar boundary between positive and negative profit difference.

Also here, the most important takeaway, is that we can obtain an important

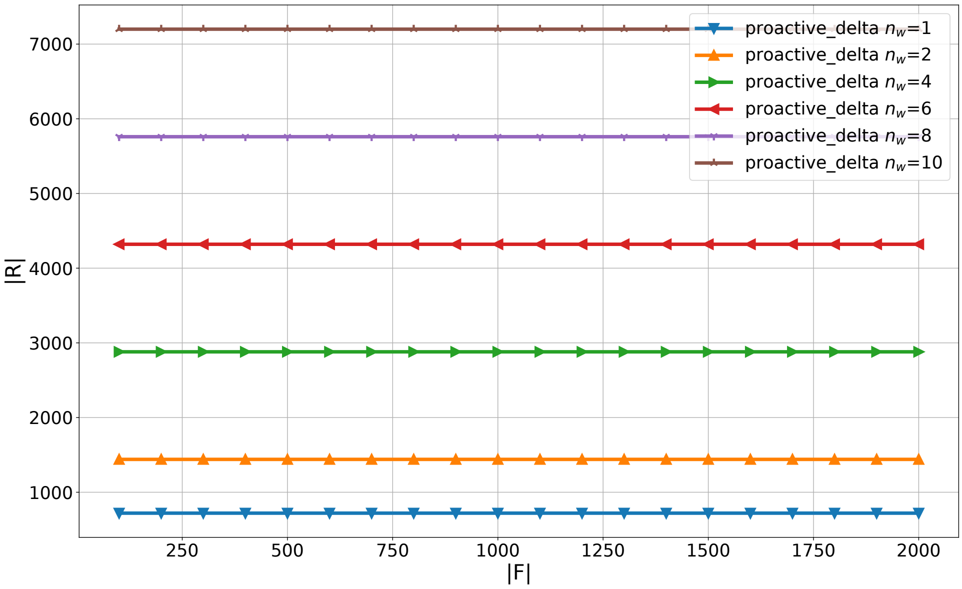

Figure 4.21: Total number of relocations in Louisville with *proactive relocation* and *Delta* as zone selection technique.

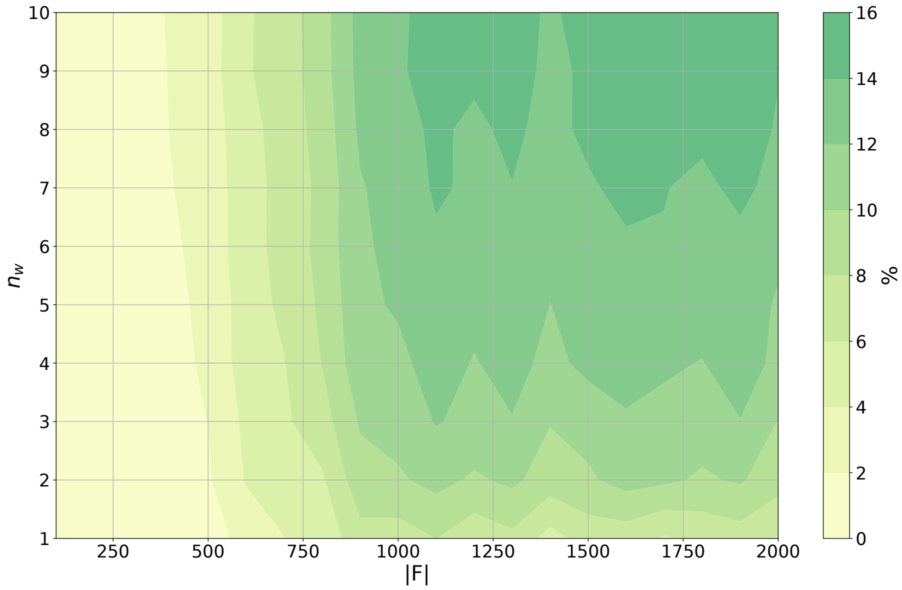

Figure 4.22: Satisfied Demand Difference in Louisville with *proactive relocation* and *Delta* as zone selection technique.

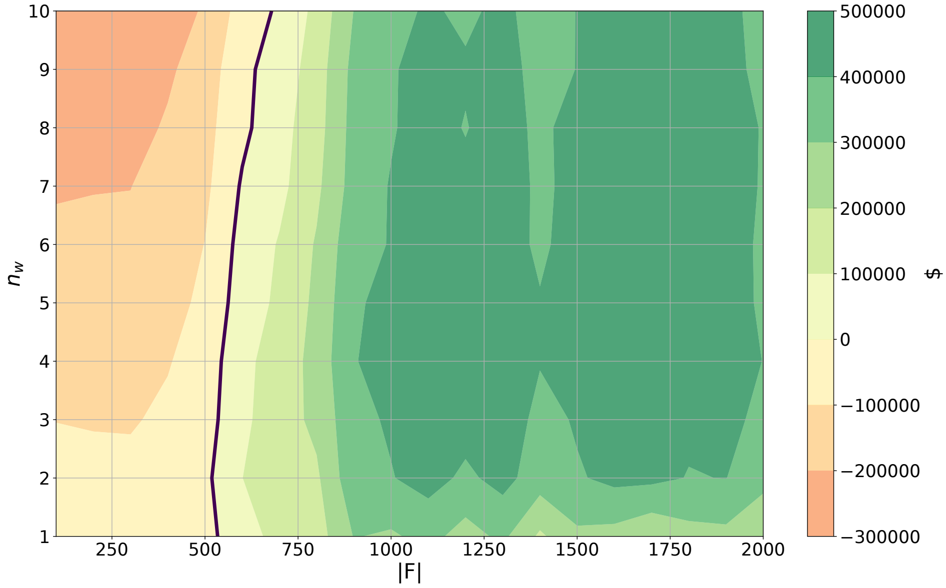

Figure 4.23: Profit Difference in Louisville with *proactive relocation* and *Delta* as zone selection technique.

improvement in performance and profit even with a fleet that is equal or slightly lower than the real world one (i.e., 850). The main difference with respect to the previous version, is that here we can obtain high profit gains also with few workers, because of the leap mentioned before, which can be explained with the number of vehicles relocated.

In Figure 4.24, indeed, we can see that we are finally better exploiting the capability of relocating more than one e-scooter at the same time, even if the average number of vehicles moved at each relocation is still quite low. We can see from the plot that it has an initial transient, where it depends on the fleet size. Then, once it reaches its maximum, it starts slowly decreasing with the increase of the number of deployed vehicles, reflecting the lowering need of relocation. But, the most important thing to notice is that the average depends a lot on the number of workers. Indeed, with one worker we obtain the highest average, and this is the reason why, even with one worker, the system already performs a lot better.

Now that we know that the best performing algorithms between the ones that we proposed, are the algorithms that use a proactive strategy, in the next Sections we are going to analyze how such strategy performs with other cities.

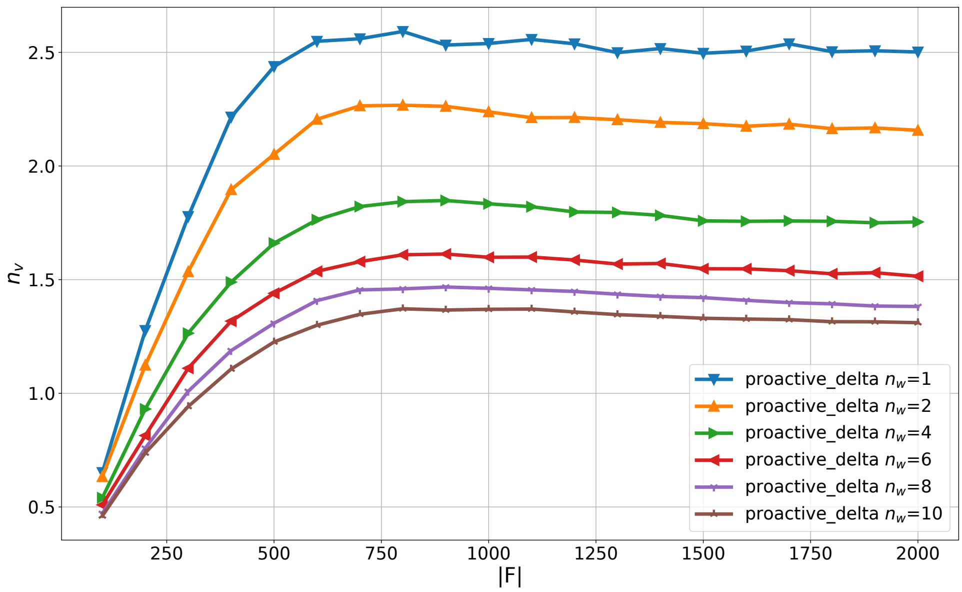

Figure 4.24: Average number of vehicles moved per relocation in Louisville with *proactive relocation* and *Delta* as zone selection technique.

4.5 Proactive relocation - Minneapolis

In this Section we will present how proactive relocation performed with data from the city of Minneapolis. As mentioned before, we generated our demand model starting from the same months as for Louisville (i.e., from July to September 2019), and we simulated one month of operations.

4.5.1 With Aggregation and KDE sampling techniques

As for Louisville, firstly, we analyze how the strategy performs with *aggregation* and *KDE sampling* as pick up and drop off zone selection techniques respectively (Figures 4.25-4.27).

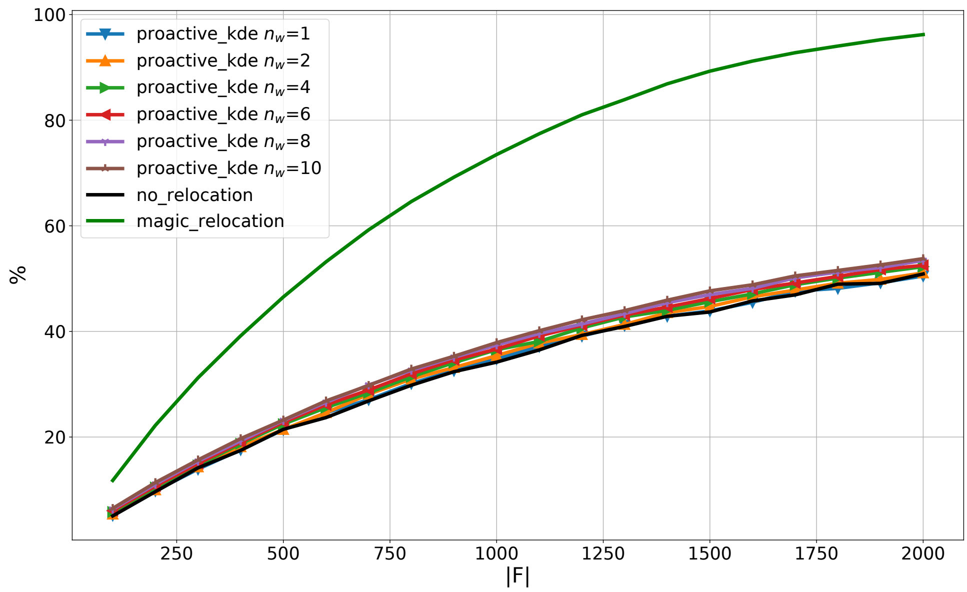

Figure 4.25: Satisfied Demand in Minneapolis with *proactive relocation* and *KDE sampling* as ending zone selection technique.

First of all, we can clearly see from Figure 4.25 that also here there is a big potential margin of improvement, considering that with 2000 vehicles, without relocation, only about 50% of the demand is satisfied. However, our algorithm struggles to improve system performance.

Indeed, we can see also from Figure 4.26 that performance improvement with respect to no relocation is constantly below 5%. And, obviously, this behaviour also translates in a poor gain in terms of profit, only with a limited number of parameter combinations (e.g., around 1000 vehicles, with 2 to 4 workers)(Figure

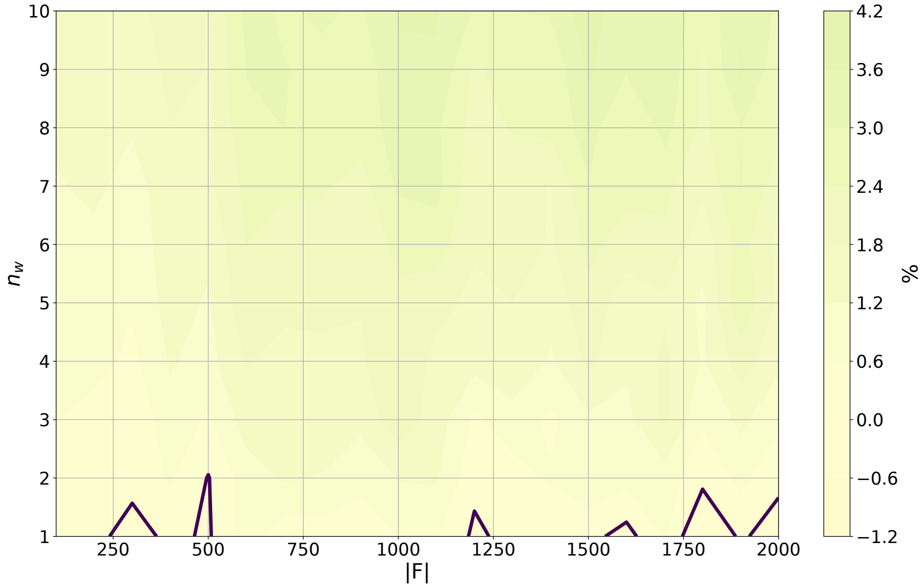

Figure 4.26: Satisfied Demand Difference in Minneapolis with *proactive relocation* and *KDE sampling* as ending zone selection technique.

4.27).

As a matter of fact, also with magic relocation we satisfy a larger number of requests only with bigger fleets. This can be due to the fact that Minneapolis has a larger operating area than Louisville and could require an higher fleet size range, to obtain a similar behaviour to the one obtained with Louisville.

However, we think that another explanation to the poor performance relies on data. Datasets provided by the city of Minneapolis are spatially aggregated by street ids (i.e., Centerline IDs, see Table 3.1), while data from Louisville has rounded coordinates. Moreover, for Minneapolis we have a time resolution of 30 minutes, while data from Louisville has a 15 minutes time bin. It is possible that, with such a level of aggregation, it is not so straight forward to disaggregate the data, as we did with a random uniform probability. Further study is required to understand how different types of aggregation affect our simulator and datasets in general, and this is, for sure, an important aspect to consider for future improvements.

4.5.2 With Delta technique

Like previous analysis, we now consider *delta* as zone selection technique (Figures 4.28-4.31).

As expected, it generally performs slightly better than the version with *KDE*

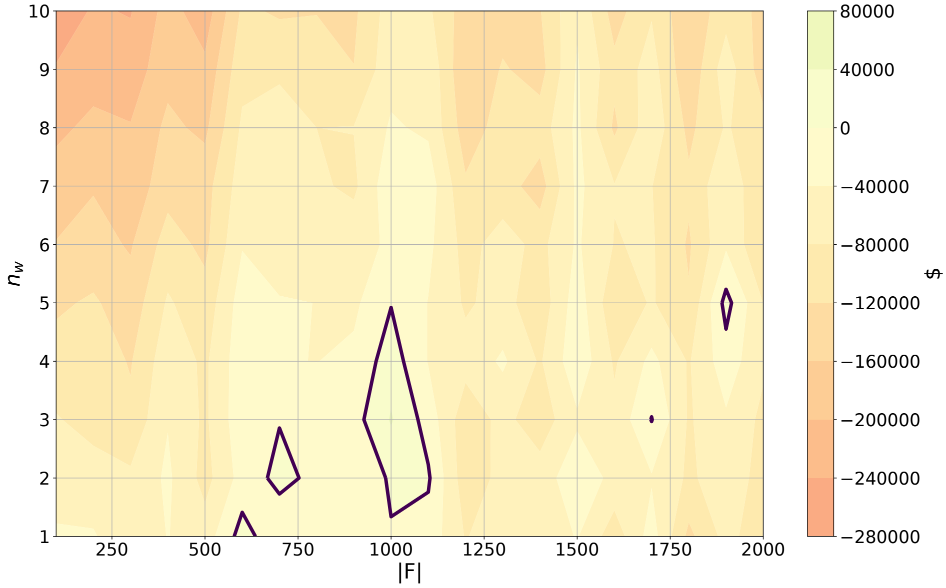

Figure 4.27: Profit Difference in Minneapolis with *proactive relocation* and *KDE sampling* as ending zone selection technique.

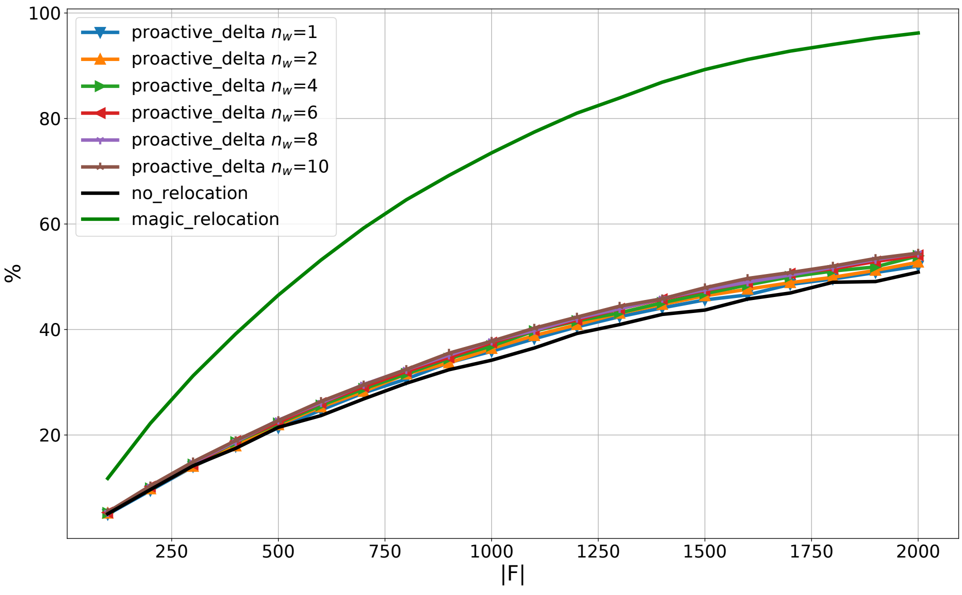

Figure 4.28: Satisfied Demand in Minneapolis with *proactive relocation* and *Delta* as zone selection technique.

sampling, but also here, the algorithm has some difficulty improving performance (Figures 4.28 and 4.29).

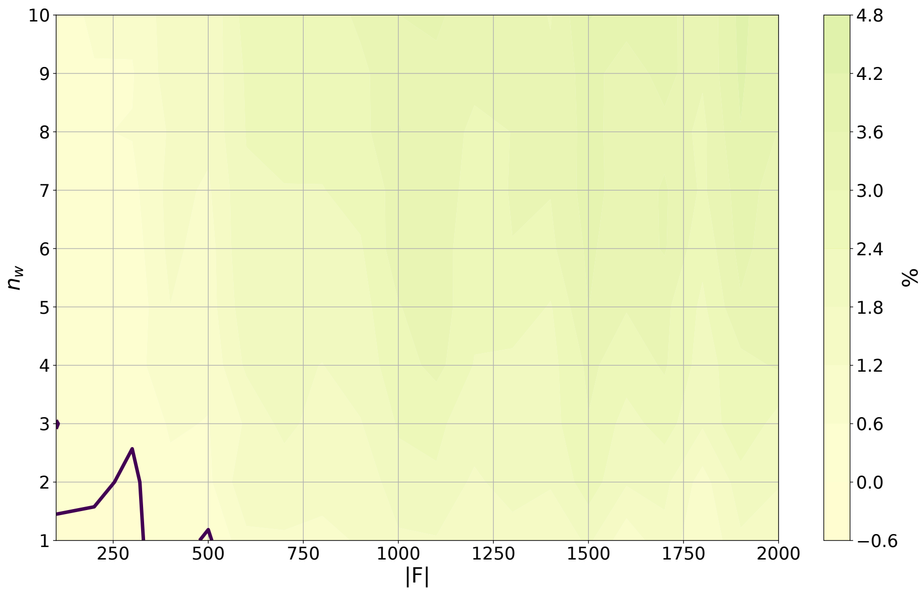

Figure 4.29: Satisfied Demand Difference in Minneapolis with *proactive relocation* and *Delta* as zone selection technique.

Only when we consider costs and revenues (Figure 4.30), we have a more similar behaviour to the expected one, with a clearer range of parameters with which the system has an higher profitability than without relocation. This means that the algorithm is not performing so well in general, but, at least, *delta* technique is working as expected, doing better relocations, moving more than one vehicle at a time (Figure 4.31).

As an explanation of why, even with *delta* technique, we have only a small improvement in terms of system performance, we can apply here the same considerations about dataset aggregation that we made for the previous version with *KDE sampling*.

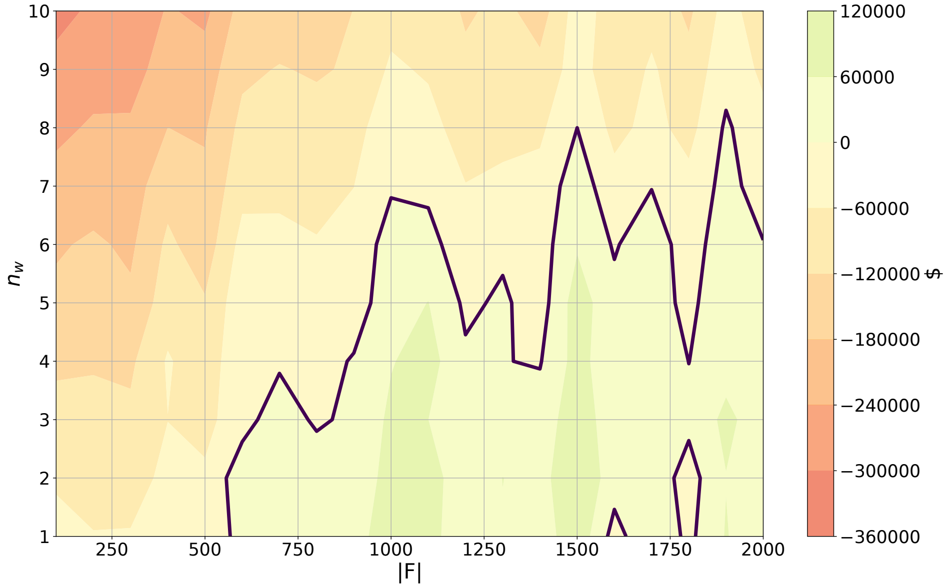

Figure 4.30: Profit Difference in Minneapolis with *proactive relocation* and *Delta* as zone selection technique.

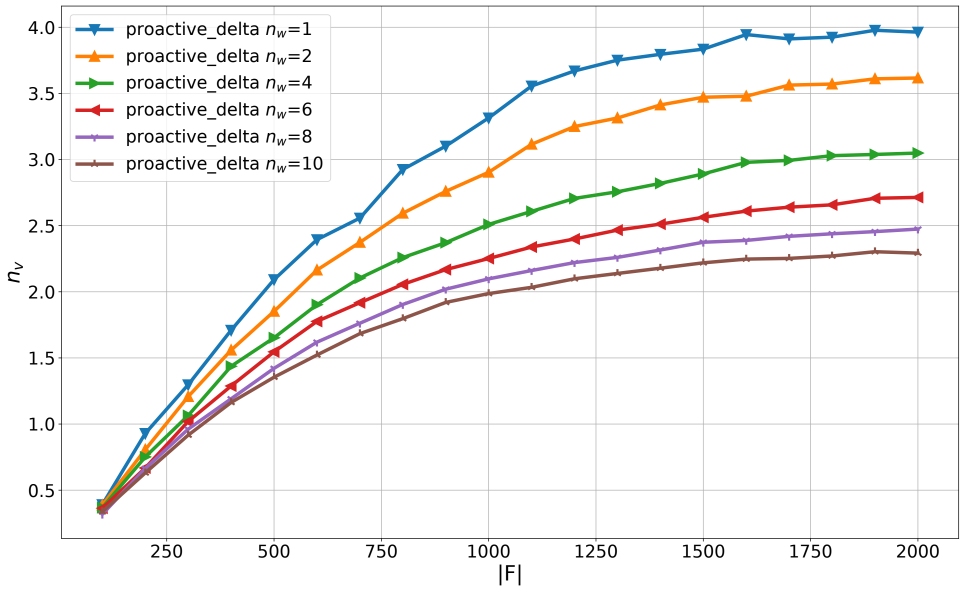

Figure 4.31: Average number of vehicles moved per relocation in Minneapolis with *proactive relocation* and *Delta* as zone selection technique.

4.6 Proactive relocation - Kansas City

The third and last city that we present is Kansas City, and also here we did simulations with both versions of proactive relocation, using the same time period for demand model generation and the same simulation duration. However, an important thing to notice is that data from Kansas City has the same spatial and time aggregation used by Louisville, so it should not present the same problems encountered with Minneapolis.

4.6.1 With Aggregation and KDE sampling techniques

As always, we start with *aggregation* and *KDE sampling* as zone selction techniques (Figures 4.32-4.34).

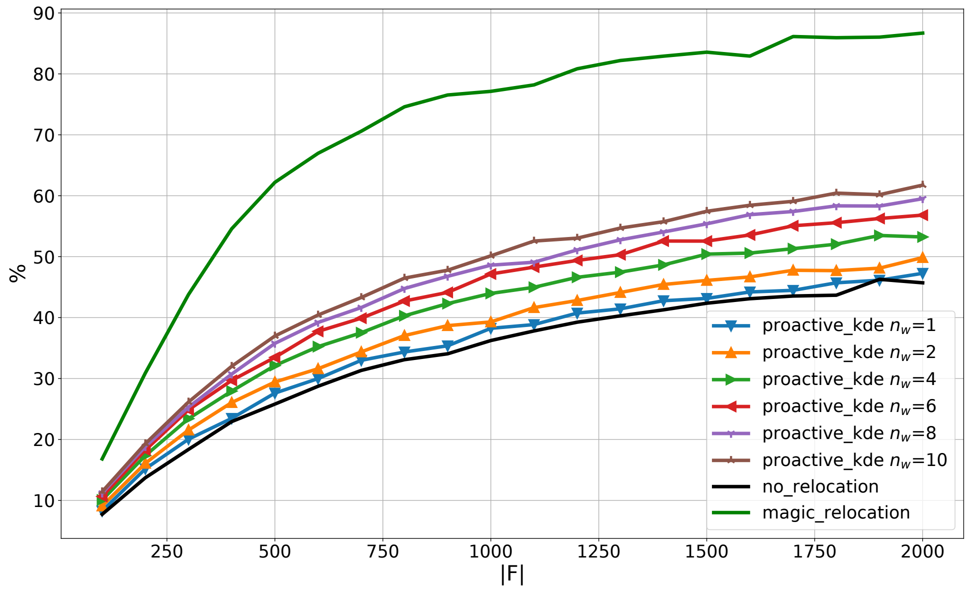

Figure 4.32: Satisfied Demand in Kansas City with *proactive relocation* and *KDE sampling* as ending zone selection technique.

We can see from Figure 4.35 that Kansas City, without relocation, has a lower satisfied demand than Louisville, so there is an even larger margin of improvement.

As expected, this time we have a very similar behaviour to the one that we had with Louisville, with even a higher maximum improvement in terms of satisfied demand (i.e., about 17%) (Figures 4.32 and 4.33).

The most important difference, here, is that a single worker has more difficulty doing all the work alone. It seems that at least 2 workers are needed to obtain a

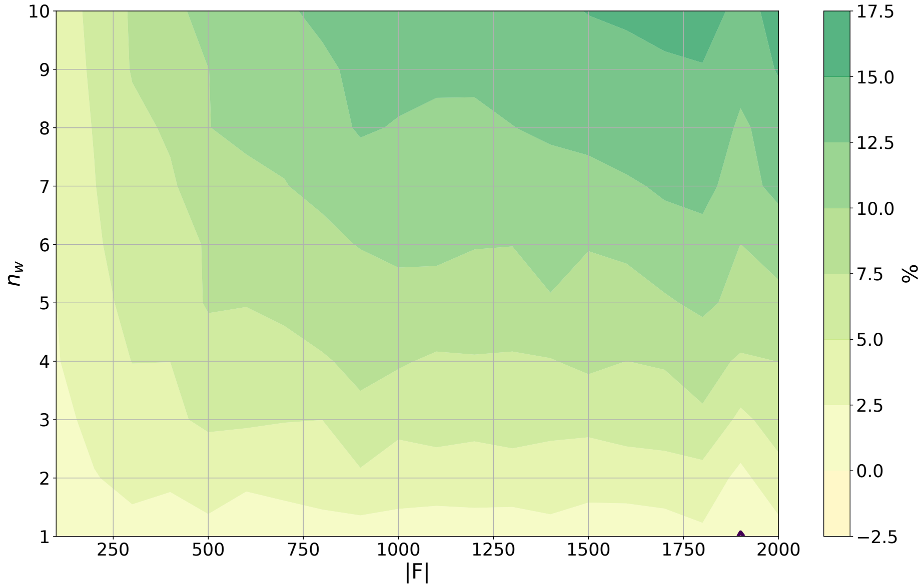

Figure 4.33: Satisfied Demand Difference in Kansas City with *proactive relocation* and *KDE sampling* as ending zone selection technique.

significant improvement in performance, and, definitely, it is partially due to the fact that here we are moving only one scooter per relocation, not leveraging the full worker potential.

Considering costs and revenues (Figure 4.34), we can see that, in terms of maximum gain, we are pretty close to Louisville, but, the most important difference, here, is that we have a positive difference even with very few vehicles, meaning that the algorithm is consistently improving performance, even in worst conditions.

As for Louisville, from our perspective, the most important takeaway here is that we can have the same system performance as without relocation, with fewer deployed vehicles, thus positively contributing to sustainability.

4.6.2 With Delta technique

Finally, we do our last comparison between cities, simulating proactive strategy with *delta* as pick up and drop off zone selection technique, in Kansas City (Figures 4.35-4.38).

As we can see from Figure 4.35, also here we have a very similar behaviour to the one obtained with Louisville. There is also the same gap between no relocation and relocation with one worker. However, as for the previous version of the algorithm, here we have a higher maximum improvement of about 20% with 10 workers.

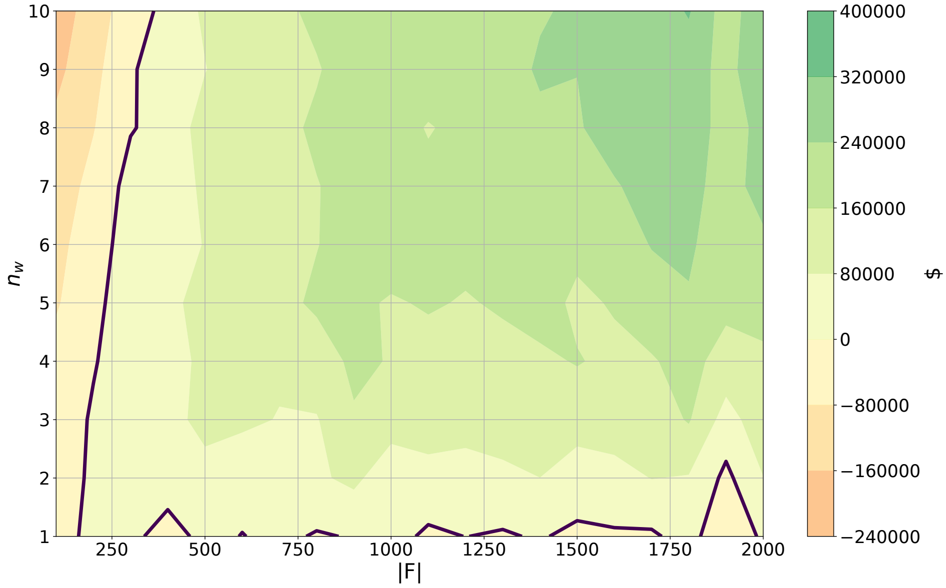

Figure 4.34: Profit Difference in Kansas City with *proactive relocation* and *KDE sampling* as ending zone selection technique.

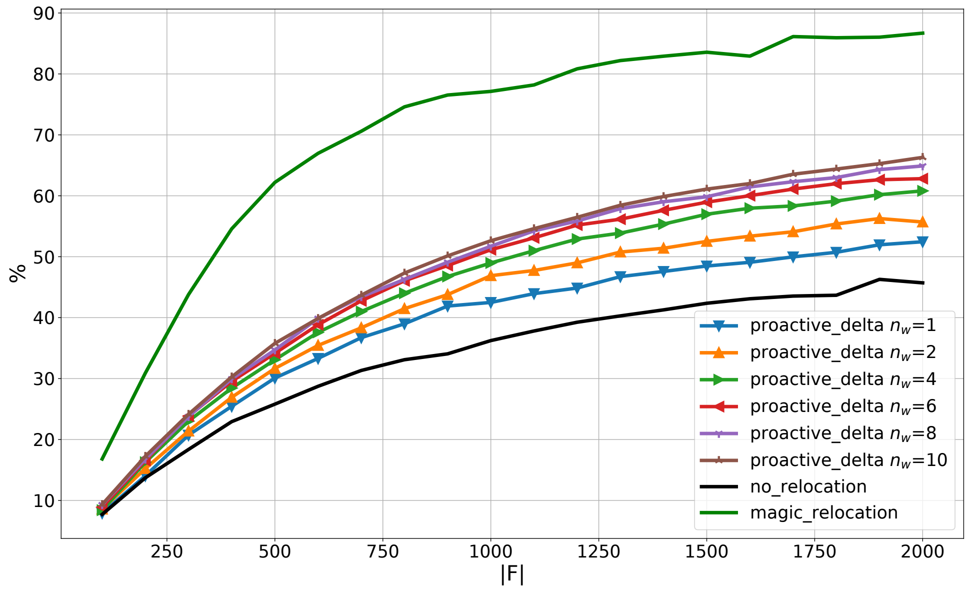

Figure 4.35: Satisfied Demand in Kansas City with *proactive relocation* and *Delta* as zone selection technique.

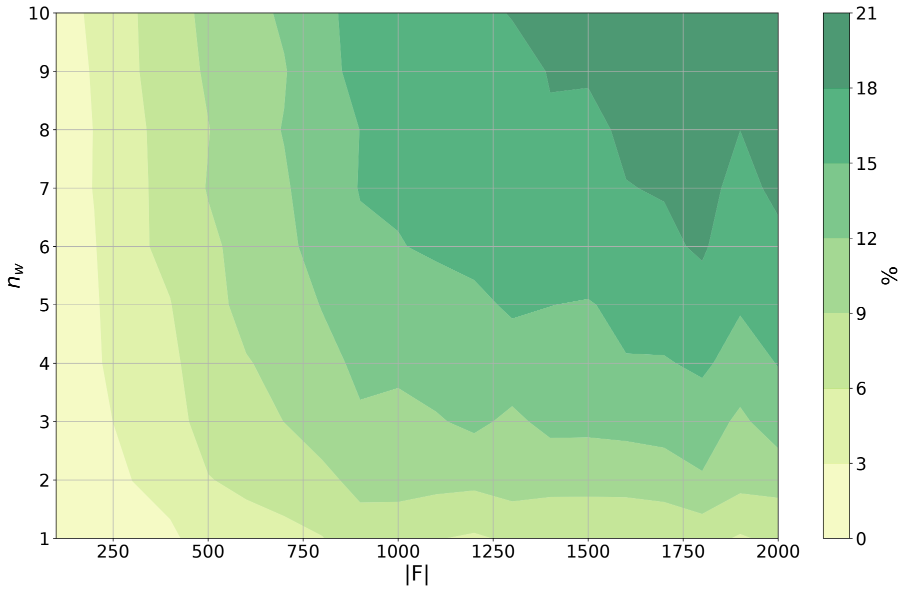

Figure 4.36: Satisfied Demand Difference in Kansas City with *proactive relocation* and *Delta* as zone selection technique.

In Figure 4.36, we can see that *satisfied demand difference* increases with more vehicles and more workers, as expected, and that there are no simulations in which we obtained a negative difference.

Profit difference reflects what we have seen in last plots (Figure 4.37), with a profit gain even with few vehicles and an higher maximum profit gain of more than 500k$.

Finally, we can observe from Figure 4.38 that we are moving a slightly higher average number of vehicles for each relocation, with respect to Louisville. This means that Kansas City presents more unbalancing issues than Louisville, that *delta* technique tries to solve, giving as output an higher suggested number of vehicles to be moved for a single relocation, on average.

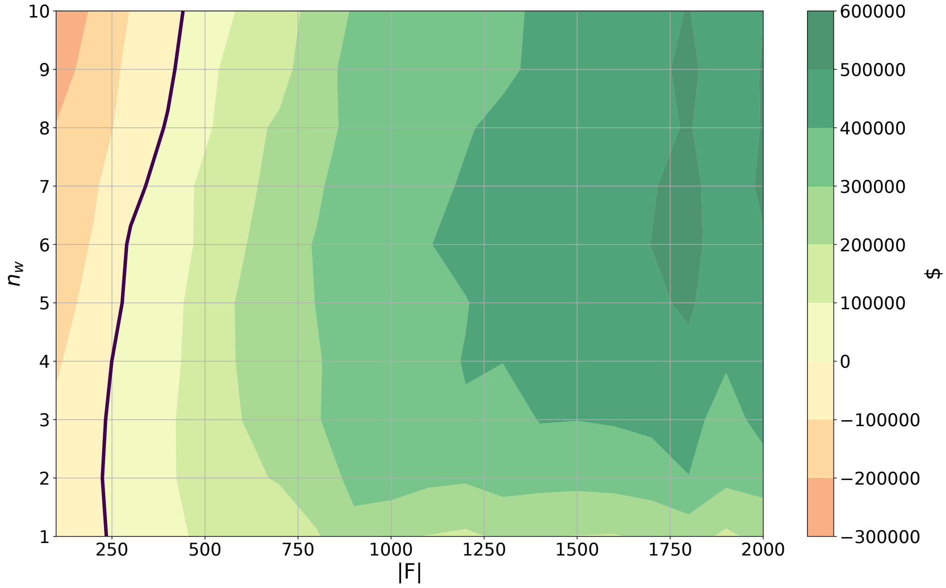

Figure 4.37: Profit Difference in Kansas City with *proactive relocation* and *Delta* as zone selection technique.

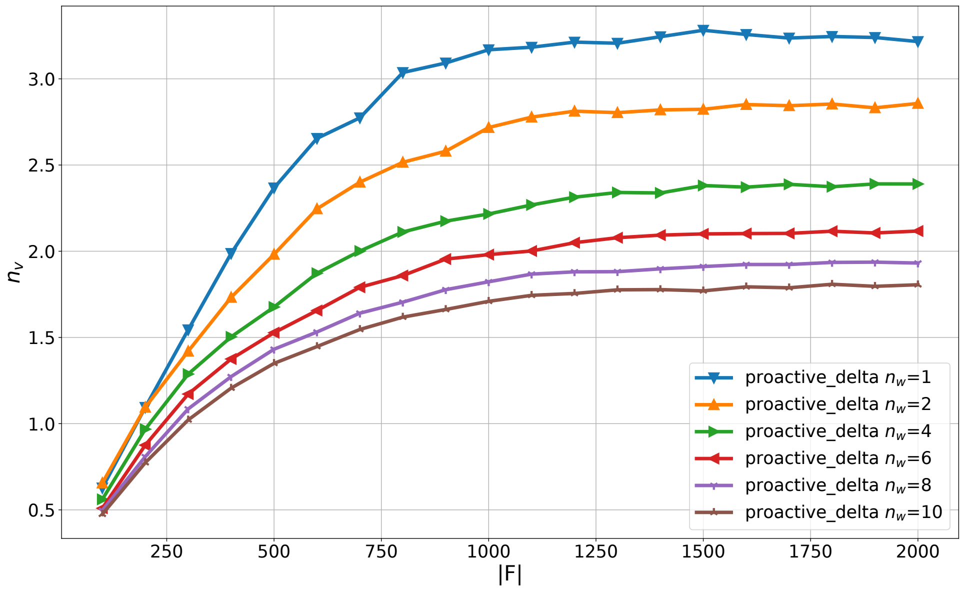

Figure 4.38: Average number of vehicles moved per relocation in Kansas City with *proactive relocation* and *Delta* as zone selection technique.

Chapter 5

Conclusions

In this thesis, we extended an existing data-driven, discrete-event simulator for FFVSS, in order to manage vehicle relocation. In particular, we focused on lightweight electric vehicles, and, for our purposes, we chose e-scooters.

They represent an increasingly popular mode of transportation, and an emerging topic in transportation research field. They have the potential to reduce private car usage, but their effective impact on the environment is still not so clear. The manufacturing phase is one of the most polluting voices in their life-cycle assessment list of contributes. Hence, we studied different relocation algorithms, to see if there is the possibility to better utilize each vehicle, reducing the deployed fleet size.

We proposed two main approaches: *reactive* and *proactive*. In Figures 5.1 and 5.2, we summarized algorithms performances in terms of satisfied demand in the case of Louisville, grouping them by the number of hired workers (i.e., $n_w = 1$ and $n_w = 10$ respectively).

Reactive relocation proved to be effective only when the action that triggers the check for relocation needs, is not too much frequent. Indeed, the trigger positioned at the end of charging procedure (*post charge*) outperformed the one placed at the end of each trip (*post trip*). Moreover, *reactive relocation* with *post charge* trigger, outperformed simulations without relocation only with a high number of workers and vehicles.

Proactive relocation was the best strategy overall, with a maximum improvement of about 15% of satisfied demand in Louisville and 20% in Kansas City, and it was the only one strategy to be able to outperform no relocation even with only one worker, with a maximum improvement in that case of about 8%.

We then made a comparison between different cities, analyzing the performance of *proactive relocation*. We found that dataset aggregation has a huge impact on the simulations: cities with minimal-invasive types of aggregation present coherent results, while cities with stronger aggregation techniques can present result that are difficult to be interpreted.

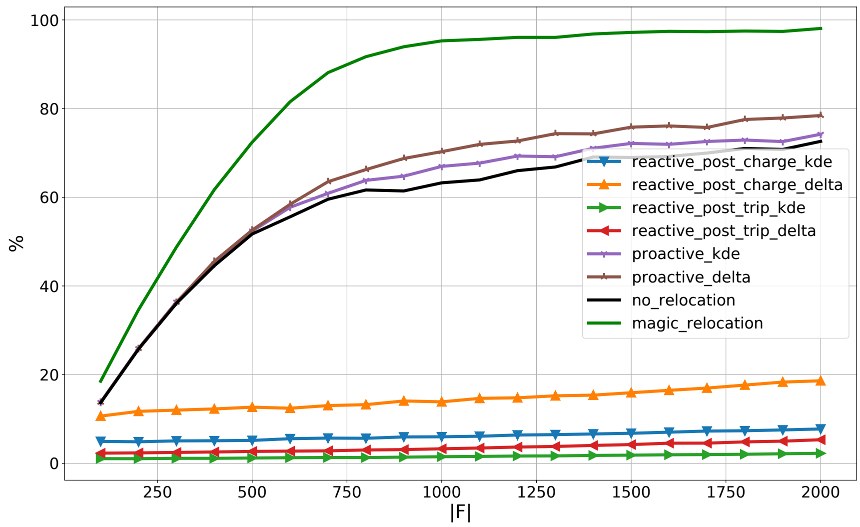

Figure 5.1: Satisfied Demand in Louisville with different relocation algorithms and $n_w = 1$.

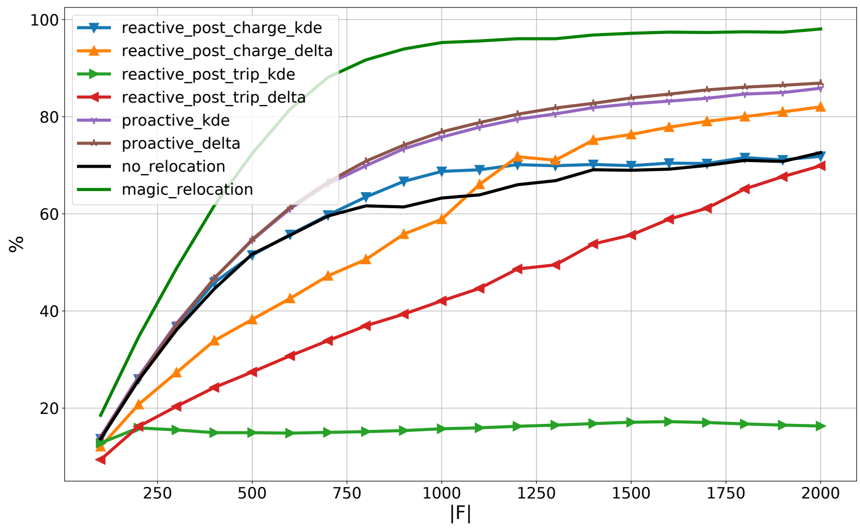

Figure 5.2: Satisfied Demand in Louisville with different relocation algorithms and $n_w = 10$.

In the future, there are different improvements that can be done and different research paths that can be taken. The first immediate path could be to further study data integration, trying to better understand how different aggregation techniques affect data usability.

Another focusing point could be the *delta* technique itself, because we did not touched the time window, but a dedicated study can be done to tune such parameter and validate the algorithm performance, also introducing other more complex methods of computing flow prediction.

Moreover, we are not simulating relocation routing. Such improvement could lead us to a more complex algorithm, a better performance analysis and a more precise cost estimation.

Finally, we can say that the most important takeaway of this work is that reducing or keeping low the number of deployed e-scooter is possible, thanks to *proactive* operations. Such strategies have the capability of making the entire system more profitable and more sustainable. They can be guided by a greedy solution that tries to solve the dynamic relocation part of the DRRP problem, in a very short time, thus giving us the possibility to simulate months of operations in a matter of hours.

Bibliography

[1] *Micromobility | Definition of Micromobility by Merriam-Webster*. URL: https://www.merriam-webster.com/dictionary/micromobility. (accessed: 18.02.2021) (cit. on p. 1).

[2] Michael McQueen, Gabriella Abou-Zeid, John MacArthur, and Kelly Clifton. «Transportation Transformation: Is Micromobility Making a Macro Impact on Sustainability?» In: *Journal of Planning Literature* 0.0 (0), p. 0885412220972696. DOI: 10.1177/0885412220972696. eprint: https://doi.org/10.1177/0885412220972696. URL: https://doi.org/10.1177/0885412220972696 (cit. on pp. 1, 4).

[3] Portland Bureau of Transportation. *2018 E-Scooter Findings Report*. URL: https://www.portlandoregon.gov/transportation/article/709719. (accessed: 18.02.2021) (cit. on pp. 1, 4, 6).

[4] Annie Chang, Luis Miranda-Moreno, Lijun Sun, and Regina Clewlow. «Trend or Fad? Deciphering the Enablers of Micromobility in the U.S.» In: (July 2019) (cit. on pp. 1–3, 6).

[5] Joseph Hollingsworth, Brenna Copeland, and Jeremiah X Johnson. «Are e-scooters polluters? The environmental impacts of shared dockless electric scooters». In: *Environmental Research Letters* 14.8 (Aug. 2019), p. 084031. DOI: 10.1088/1748-9326/ab2da8. URL: https://doi.org/10.1088/1748-9326/ab2da8 (cit. on pp. 2, 4, 6).

[6] *Lime-S San Diego 2018 - Lime (azienda) - Wikipedia*. URL: https://it.wikipedia.org/wiki/Lime_(azienda)#/media/File:Lime-S_San_Diego_2018.jpg. (acessed: 22.02.2021) (cit. on p. 2).

[7] *Lime S QR Code - Lime (azienda) - Wikipedia*. URL: https://it.wikipedia.org/wiki/Lime_(azienda)#/media/File:Lime_S_QR_Code.jpg. (acessed: 22.02.2021) (cit. on p. 2).

[8] Yunhe Feng, Dong Zhong, Peng Sun, Weijian Zheng, Qinglei Cao, Xi Luo, and Zheng Lu. *Micromobility in Smart Cities: A Closer Look at Shared Dockless E-Scooters via Big Social Data.* 2020. arXiv: 2010.15203 [cs.SI] (cit. on pp. 4, 6).

[9] A. Ciociola, M. Cocca, D. Giordano, L. Vassio, and M. Mellia. «E-Scooter Sharing: Leveraging Open Data for System Design». In: *2020 IEEE/ACM 24th International Symposium on Distributed Simulation and Real Time Applications (DS-RT).* 2020, pp. 1–8. DOI: 10.1109/DS-RT50469.2020.9213514 (cit. on pp. 5–8, 11–13).

[10] Karl Tracksdorf Samantha Leger Dave McLaughlin. *Leading the Charge on Canadian E-bike Integration: A Discussion on the Emerging and Unchartered Role of Micromobility.* URL: https://www.wsp.com/en-CA/insights/ca-micromobility. (accessed: 16.03.2021) (cit. on p. 6).

[11] Mina Lee, Joseph Y. J. Chow, Gyugeun Yoon, and Brian Yueshuai He. *Forecasting e-scooter substitution with direct and access trips by mode and distance in New York City.* 2021. arXiv: 1908.08127 [econ.GN] (cit. on p. 6).

[12] Rui Zhu, Xiaohu Zhang, Dániel Kondor, Paolo Santi, and Carlo Ratti. «Understanding spatio-temporal heterogeneity of bike-sharing and scooter-sharing mobility». In: *Computers, Environment and Urban Systems* 81 (2020), p. 101483. ISSN: 0198-9715. DOI: https://doi.org/10.1016/j.compenvurbsys.2020.101483. URL: https://www.sciencedirect.com/science/article/pii/S0198971519305812 (cit. on p. 6).

[13] Maria Lucchi James Fong Paul McDermott. *Micro-Mobility, E-Scooters and Implications for Higher Education.* URL: https://upcea.edu/micro-mobility-e-scooters-and-implications-for-higher-education/. (accessed: 16.03.2021) (cit. on p. 6).

[14] Tal Raviv and Ofer Kolka. «Optimal inventory management of a bike-sharing station». In: *IIE Transactions* 45.10 (2013), pp. 1077–1093. DOI: 10.1080/0740817X.2013.770186. eprint: https://doi.org/10.1080/0740817X.2013.770186. URL: https://doi.org/10.1080/0740817X.2013.770186 (cit. on p. 7).

[15] D. Jorge, G. H. A. Correia, and C. Barnhart. «Comparing Optimal Relocation Operations With Simulated Relocation Policies in One-Way Carsharing Systems». In: *IEEE Transactions on Intelligent Transportation Systems* 15.4 (2014), pp. 1667–1675. DOI: 10.1109/TITS.2014.2304358 (cit. on p. 7).

[16] Supriyo Ghosh, Pradeep Varakantham, Yossiri Adulyasak, and Patrick Jaillet. «Dynamic Repositioning to Reduce Lost Demand in Bike Sharing Systems». In: *J. Artif. Int. Res.* 58.1 (Jan. 2017), pp. 387–430. ISSN: 1076-9757 (cit. on p. 7).

[17] Joseph Warrington and Dominik Ruchti. «Two-stage stochastic approximation for dynamic rebalancing of shared mobility systems». In: *Transportation Research Part C: Emerging Technologies* 104 (2019), pp. 110–134. ISSN: 0968-090X. DOI: https://doi.org/10.1016/j.trc.2019.04.021. URL: https://www.sciencedirect.com/science/article/pii/S0968090X18314104 (cit. on p. 7).

[18] D. Ljubenkov, F. Kon, and C. Ratti. «Optimizing Bike Sharing System Flows Using Graph Mining, Convolutional and Recurrent Neural Networks». In: *2020 IEEE European Technology and Engineering Management Summit (E-TEMS)*. 2020, pp. 1–6. DOI: 10.1109/E-TEMS46250.2020.9111707 (cit. on p. 7).

[19] *Welcome to Python.org*. URL: https://www.python.org/. (accessed: 19.02.2021) (cit. on p. 8).

[20] *Overview — SimPy 4.0.2.dev1+g2973dbe documentation*. URL: https://simpy.readthedocs.io/en/latest/. (accessed: 19.02.2021) (cit. on pp. 8, 13).

[21] *pandas - Python Data Analysis Library*. URL: https://pandas.pydata.org/. (accessed: 19.02.2021) (cit. on p. 8).

[22] *GeoPandas 0.8.0 — GeoPandas 0.8.0 documentation*. URL: https://geopandas.org/. (accessed: 19.02.2021) (cit. on p. 8).

[23] *scikit-learn: machine learning in Python — scikit-learn 0.24.1 documentation*. URL: https://scikit-learn.org/stable/. (accessed: 19.02.2021) (cit. on pp. 8, 12, 15).

[24] *Dockless Vehicle Trips - Block Level - 2018-08 to 2020-01 - Dockless Vehicles | Louisville Metro Open Data*. URL: https://data.louisvilleky.gov/dataset/dockless-vehicles/resource/e36546f6-888b-4e66-8a87-9b68cab471e6#%7B%7D. (accessed: 19.02.2021) (cit. on p. 9).

[25] *Dockless Trips 02/01/2020-07/08/2020 - Dockless Vehicles | Louisville Metro Open Data*. URL: https://data.louisvilleky.gov/dataset/dockless-vehicles/resource/449c6bd1-cd4e-404a-ad80-76895fef91a0#%7B%7D. (accessed: 19.02.2021) (cit. on p. 9).

[26] *Motorized Foot Scooter Trips May 2019 | Open Minneapolis*. URL: https://opendata.minneapolismn.gov/datasets/motorized-foot-scooter-trips-may-2019. (accessed: 19.02.2021) (cit. on p. 9).

[27] *Motorized Foot Scooter Trips June 2019 | Open Minneapolis*. URL: https://opendata.minneapolismn.gov/datasets/motorized-foot-scooter-trips-june-2019. (accessed: 19.02.2021) (cit. on p. 9).

[28] *Motorized Foot Scooter Trips July 2019 | Open Minneapolis.* URL: `https://opendata.minneapolismn.gov/datasets/motorized-foot-scooter-trips-july-2019`. (accessed: 19.02.2021) (cit. on p. 9).

[29] *Motorized Foot Scooter Trips August 2019 | Open Minneapolis.* URL: `https://opendata.minneapolismn.gov/datasets/motorized-foot-scooter-trips-august-2019`. (accessed: 19.02.2021) (cit. on p. 9).

[30] *Motorized Foot Scooter Trips September 2019 | Open Minneapolis.* URL: `https://opendata.minneapolismn.gov/datasets/motorized-foot-scooter-trips-september-2019`. (accessed: 19.02.2021) (cit. on p. 9).

[31] *Motorized Foot Scooter Trips October 2019 | Open Minneapolis.* URL: `https://opendata.minneapolismn.gov/datasets/motorized-foot-scooter-trips-october-2019`. (accessed: 19.02.2021) (cit. on p. 9).

[32] *Motorized Foot Scooter Trips November 2019 | Open Minneapolis.* URL: `https://opendata.minneapolismn.gov/datasets/motorized-foot-scooter-trips-november-2019-1`. (accessed: 19.02.2021) (cit. on p. 9).

[33] *Shared Micromobility Vehicle Trips | Open Data | City of Austin Texas.* URL: `https://data.austintexas.gov/Transportation-and-Mobility/Shared-Micromobility-Vehicle-Trips/7d8e-dm7r`. (accessed: 19.02.2021) (cit. on p. 9).

[34] *Scooters | Open Data Portal - City of Norfolk, VA Open Data.* URL: `https://data.norfolk.gov/Government/Scooters/wqxq-hhe6`. (accessed: 19.02.2021) (cit. on p. 9).

[35] *Microtransit (Scooter and Ebike) Trips | Open Data KC | data.kcmo.org.* URL: `https://data.kcmo.org/Transportation/Microtransit-Scooter-and-Ebike-Trips/dy5n-ewk5`. (accessed: 19.02.2021) (cit. on p. 9).

[36] *E-Scooter Trips - 2019 Pilot | City of Chicago | Data Portal.* URL: `https://data.cityofchicago.org/Transportation/E-Scooter-Trips-2019-Pilot/2kfw-zvte`. (accessed: 19.02.2021) (cit. on p. 9).

[37] *Shared Mobility Pilot Trips | Open Calgary.* URL: `https://data.calgary.ca/Transportation-Transit/Shared-Mobility-Pilot-Trips/jicz-mxiz`. (accessed: 19.02.2021) (cit. on p. 9).

[38] *openmobilityfoundation/mobility-data-specification: A data standard to enable communication between mobility companies and local governments.* URL: `https://github.com/openmobilityfoundation/mobility-data-specification`. (accessed: 19.02.2021) (cit. on p. 8).

[39] Aditya Krishna Menon and Young Lee. «Predicting Short-Term Public Transport Demand via Inhomogeneous Poisson Processes». In: *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. CIKM '17. Singapore, Singapore: Association for Computing Machinery, 2017, pp. 2207–2210. ISBN: 9781450349185. DOI: `10.1145/3132847.3133058`. URL: `https://doi.org/10.1145/3132847.3133058` (cit. on p. 12).