ARIEL UNIVERSITY

MASTER THESIS

A Human-Centric Approach for Last-Mile Ridesharing

Author:
Chaya Levinger

Supervisor:
Dr. Amos Azaria
Dr. Noam Hazon

Department of Computer Science

September 24, 2019

Declaration of Authorship

I, Chaya Levinger, hereby declare that this thesis entitled, "A Human-Centric Approach for Last-Mile Ridesharing" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

Ariel University

Abstract

Faculty of Natural Sciences
Department of Computer Science

Master

A Human-Centric Approach for Last-Mile Ridesharing

by Chaya Levinger

Transportation services play a crucial part in the development of modern smart cities. In particular, on-demand ridesharing services, which group together passengers with similar itineraries, are already operating in several metropolitan areas. These services can be of significant social and environmental benefit, by reducing travel costs, road congestion and CO_2 emissions. The deployment of autonomous cars in the near future will surely change the way people are traveling. It is even more promising for a ridesharing service, since it will be easier and cheaper for a company to handle a fleet of autonomous cars that can serve the demands of different passengers.

A special case in which ridesharing services are most applicable, is the last mile variant. In this variant it is assumed that all the passengers are positioned at the same origin location (e.g. an airport), and each have a different destination. In this thesis we focus on the last mile variant. We believe that a human-centric approach is needed for a wide-spread adaptation of ridesharing services. We thus investigate the human-centric approach from three aspects.

First, we analyze the assignment problem where the objective is to maximize the user satisfaction. We argue that user satisfaction should be the main objective when trying to find the best assignment of passengers to vehicles and the determination of their routes. Moreover, the model of user satisfaction should be rich enough to capture the traveling time, cost, and other factors as well. We show that it is more important to capture a rich model of human satisfaction than peruse an optimal performance. That is, we developed a practical algorithm for assigning passengers to vehicles, which outperforms brute-force assignment algorithms that use a simpler satisfaction model.

The second part of this thesis investigates another major aspect of ridesharing, which is the splitting of the ride cost among the passengers fairly. Deciding how to split the ride cost is significant and directly impacts passenger satisfaction. For this matter we use the Shapley value, which is one of the most significant solution concepts in cooperative game theory, for fairly splitting the cost of a shared ride.

We consider two scenarios. In the first scenario there exists a fixed priority order in which the passengers are dropped-off (e.g. elderly, injured etc.), and we show a method for efficient computation of the Shapley value in this setting.

In the second scenario there is no predetermined priority order. We show that the Shapley value cannot be efficiently computed in this setting. However, extensive simulations reveal that our approach for the first scenario can serve as an excellent proxy for the second scenario, outperforming other known proxies.

The last part of this thesis investigates the drop-off order aspect. Since we adopt a human-centric approach, we would like the passengers to be able to set the drop-off

order according to their needs. We thus present a VCG based mechanism for setting the drop-off order that obtains the value of time from each of the passengers and outputs a drop-off order. The mechanism is both efficient and truthful. We provide simulations showing that the overhead cost paid to the mechanism is reasonable, and is significantly lower than the commission charged by companies providing ride-sourcing services.

Acknowledgements

I would like to thank Ariel University for the fellowship which support me while conducting my research.

In addition, I would like to special thank Dr. Amos Azaria and Dr. Noam Hazon, who escorted me (and still) throughout the process.

My deep thanks to my supportive family, my loving and supportive husband Ariel and my wonderful kids Dvir and Miryam.

This work was supported in part by Voltswagen Stiftung under grant "EC-RIDER: Explainable AI Methods for Human-Centric Ridesharing".

Dedicated with love, to my dear and supportive grandfather.
May he have a good and long life.

Contents

List of Figures

List of Tables

List of Abbreviations

VRP	**V**ehicle **R**outing and scheduling **P**roblem
VRPB	**V**ehicle **R**outing **P**roblems with **B**ackhauls
VRPPD	**V**ehicle **R**outing **P**roblems with **P**ickups and **D**eliveries
DARP	**D**ial-**A**-**R**ide **p**roblem
VCG	**V**ickrey **C**larke **G**roves
MAE	**M**ean **A**bsolute **E**rror
MSE	**M**ean **S**quared **E**rror
RMSE	**R**oot **M**ean **S**quared **E**rror

Chapter 1

Introduction

The National Household Travel Survey performed in the U.S. in 2009 [68] revealed that approximately 83.4% of all trips in the U.S. were in a private vehicle (other options being public transportation, walking, etc.). The average vehicle occupancy was only 1.67 when compensating for the number of passengers. This extremely low average vehicle occupancy entails a very large number of vehicles on the road that collectively contribute to carbon dioxide emissions, fuel consumption, air pollution and an increase in traffic load, which in turn requires additional investment in enlarging the road infrastructure. In recent years, ride hailing services such as Uber and Lyft have gained popularity and an increasing number of passengers use these services as one of their main means of transportation [79]. Both Uber and Lyft are now also offering ridesharing options, and other companies, such as Super-Shuttle and Via, are explicitly targeted at customers who want to share their ride.

The deployment of autonomous vehicles in the near future will have a significant impact on the way people are traveling. The implication of this revolutionary way of transportation is not fully known nowadays [33], but it is safe to claim that autonomous vehicles will have a positive effect on the development of ridesharing services. Indeed, it will be easier and cheaper for a company to handle a fleet of autonomous vehicles that can serve the demands of different passengers. It can also rule-out some negative human-driver factors, such as driver's fatigue from the long travels and the driver's inconvenience from having multiple pick-up and drop stops along his route.

We note that the term ridesharing is used in the literature with different meanings. As we will later define, we consider only the setting where the vehicle operator does not have any preferences or predefined origin and destination. Instead, the vehicle's route is determined solely by the passengers' requests.

In this thesis we focus on the last-mile setting [17], i.e., it is assumed that all the passengers are positioned at the same origin location (e.g. airport), where all the vehicles are also located, and must be taken to their final destination. This setting is very important, as any mass public transportation solution must also deal with the last (and first) mile setting. While large aircrafts and trains may transport many people at once, each person must eventually arrive at her final destination.

Ridesharing has a true potential for improving the quality of life for many people [18], and it is part of the general concept of sharing economy that is being evolved nowadays. However, despite both Uber and Lyft offering ridesharing options, not many users elect to share their rides with additional passengers [42, 14]. Following the statement by Carnegie [15, p. 37], "There is only one way to get anybody to do anything. And that is by making the other person want to do it.", we believe that the key ingredient required for a widespread adaptation of ridesharing is to focus on user satisfaction. We therefore investigate a comprehensive human-centric approach for ridesharing.

In this thesis we analyze user satisfaction from three aspects: In Chapter 2, we analyze the assignment problem. We collect information from human about the parameters affecting passenger satisfaction, and develop a learning-based function that weighs all of these parameters and ranks the passenger's satisfaction. In addition, we develop an efficient algorithm for the assignment of passengers to vehicles. We show that our algorithm, with the function we have developed, outperforms the brute-force algorithm that uses a simpler satisfaction function.

In Chapter 3, we analyze the cost allocation problem. We use the Shapley value and show that when passengers are prioritized, it can be calculated in polynomial time. In contrast, when no such order exists, we show that the Shapley value cannot be calculated in polynomial time unless P = NP, but we show that computing the Shapley value with a priority order can serve as an excellent proxy.

In Chapter 4, we would like the passengers to be able to set the drop-off order according to their needs. We therefore propose a VCG-based mechanism for determining the drop-off order. Our mechanism collects information from the passengers regarding their preferences, and determines the optimal drop-off order. We show that our mechanism is both truthful and efficient.

Chapter 2

Human Satisfaction as the Objective for Assignment

2.1 Introduction

The basic challenge of a ridesharing service is how to assign the passengers' requests for a ride to vehicles and define the routes for the fleet of vehicles in an optimal manner. This problem belongs to the generic class of Vehicle Routing and scheduling Problems (VRPs), which have been extensively studied over the past 50 years, mainly in the operation research and transportation science communities. Several variants with different characteristics have been developed. For example, the initial formulation of the VRP assumes that the environment is *static*, i.e., all requests are known before-hand and do not change thereafter [21]. The more complex variants, including the rideharing problem, are *dynamic*, where real-time requests are gradually revealed along the service operating time. In this setting the assignment of passengers to vehicles and the determination of vehicles' routes may be adjusted when they are already in transit [66, 73]. Arguably, a major criterion that characterizes each variant of the VRP is the objective function. It is very common to consider objectives from the service provider's perspective, for example, minimizing the total distance travelled [70, 1], minimizing the fleet size [22, 71], or maximizing the service provider's profit [13, 59]. However, as noted by Cordeau and Laporte[20], one should be interested not only in minimizing the operating costs for the service provider but also in maximizing the quality of the service and the user satisfaction.

Many works integrate quality of service and user satisfaction considerations as additional constraints of the problem. For example, a time window restricts the waiting time a passenger is willing to face before being picked up [37, 24], and it is usually combined with a bound on the maximum user ride time [56]. In addition, there are several works that combine the aforementioned operational objectives with the objective of maximizing the user satisfaction (or its antonym, minimizing the user inconvenience). The common interpretation for user satisfaction is the minimization of the total user on-board (ride) time and the total user waiting time [65], the extra riding time due to ridesharing [44], or the amount of deviations from desired departure and arrival times [29, 82]. Notably, Lyu et al. [47], allow the users to provide a set of constraints and to indicate the importance of each of the following features: walking distance, waiting time, extra travel time, and travel fare. Lyu et al. make the assumption that the user preferences are specified by each user in each ride request, and they provide an algorithm that uses this information to maximize user satisfaction. However, to the best of our knowledge, there are no works in the ridesharing domain that develop a general user satisfaction function, and exclusively focus on maximizing it.

Our basic claim in this chapter is that the user satisfaction should be the main objective of the ridesharing service. Moreover, the model of user satisfaction should be rich enough to capture the complex inter-dependencies among several factors. In the last-mile setting, we develop a method for modeling and maximizing a complex user satisfaction function.

One approach for handling a rich objective function is to treat its factors as multiple objectives. Indeed, there are several methods in the literature on VRP for handling multiple objectives. The most common approach is to aggregate the objectives into a single weighted-sum objective function [49], and advanced utility model may be used for modeling the interactions between the objectives [43]. Additional strategies define hierarchical objective function [69], or return a set of non-comparable solutions which do not weakly dominate each other [60, 50]. Since our rich objective function models user satisfaction, we propose a different, human-centric, approach. Specifically, we investigate machine learning methods for modeling the rich satisfaction function from real humans. Clearly, it is unrealistic to elicit the exact user satisfaction for each passenger and every ride, and we thus propose to build a general model for user satisfaction, which is based on multiple features. These features include both ride specific features (e.g. cost, travel time) and person specific features (e.g. age, gender) and thus two people may obtain different satisfaction levels from similar rides.

Interacting with humans and learning human behavior is a very complex task. Research into humans' behavior has found that people often deviate from what is thought to be the rational behavior, since they are affected by a variety of (sometimes conflicting) factors: a lack of knowledge of one's own preferences, the effects of the task complexity, framing effects, the interplay between emotion and cognition, the problem of self-control, the value of anticipation, future discounting, anchoring and many other effects [77, 46, 6, 12]. Therefore, algorithmic approaches that use a pure theoretically analytic objective often perform poorly with real humans [61, 8, 53]. On the other hand, several works have demonstrated that a machine learning approach, which builds upon psychological factors and human decision-making theory, is essential for developing a good model of true human behavior. The human behaviour model is in turn required for successfully implementing algorithms that interact with humans [30, 36, 75, 67, 7, 4, 34]. We therefore ran experiments with actual humans and build a deep learning based function to estimate user satisfaction. We introduce Simsat, an algorithm for assigning passengers to vehicles while maximizing a complex user satisfaction function as the objective. We show that Simsat outperforms brute-force assignment methods that use a simpler objective function, indicating that it is more important to obtain a richer model of user satisfaction, than improving the performance of the assignment algorithm.

2.2 Related Work

We will briefly review the current literature on the broad class of Vehicle Routing and scheduling Problems (VRPs), to place our assignment problem in an appropriate context. The VRP was first introduced by Dantzig and Ramser[21]. The growing body of research on routing problem over the past 50 years has led to the development of several research communities, which sometimes denote the same problem types by various names. In particular, the traditional VRP and some of its extensions deal with finding an optimal set of routes for a fleet of vehicles to traverse in order deliver or pickup some goods to a given set of costumers. We refer to the

comprehensive survey of Parragh, Doerner, and Hartl[57] on this class of problems, which they denote by Vehicle Routing Problems with Backhauls (VRPB). A more recent survey, that also defines a taxonomy to classify the various variants of VRP by 11 criteria, is given by Psaraftis, Wen, and Kontovas[66]. A second class of problems, that is denoted by Parragh et al. as Vehicle Routing Problems with Pickups and Deliveries (VRPPD), deal with all those problems where goods are transported between pickup and delivery customers. We refer to the survey of Parragh, Doerner, and Hartl[58] on this class of problems. One subclass of VRPPD compromises the dial-a-ride problem (DARP), where the goods that are transported are passengers with associated pickup and delivery points. As noted by Cordeau and Laporte[20], the DARP is distinguished from other problems in vehicle routing since transportation cost and user inconvenience must be weighed against each other in order to provide an appropriate solution. Therefore, the DARP typically includes more quality constraints that aim at capturing the user's inconvenience. We refer to a recent survey on DARP by Molenbruch, Braekers, and Caris[49], which also makes this distinction.

A domain closely related to ridesharing is car-pooling. In this domain, ordinary drivers, may opt to take an additional passenger on their way to a shared destination. The common setting of car-pooling is within a long-term commitment between people to travel together for a particular purpose, where ridesharing is focused on single, non-recurring trips. Indeed, several works investigated car-pooling that can be established on a short-notice, and they refer to this problem as ridesharing [2, 51]. We stress that in our ridesharing assignment problem, similar to the DARP setting, there is a central organization that owns the vehicles, and they thus do not have their own travel plans. Indeed,Montazery and Wilson[51] investigated the matching of passengers to cars in the car-pooling domain, based on a user satisfaction function that they learn. However, they use this function to build a recommendation system that recommends a set of best possible matching to each passenger.

2.3 Basic Model and Notation

Informally, the assignment problem consists of a weighted graph, requests given by passengers, each with an origin and a destination that are both nodes in the graph, and a set of vehicles, each with a given capacity. All the vehicles are assumed to be operated by a central entity. In the last-mile setting we need to assign travel routes (on the graph) to vehicles, in order to satisfy the passenger requests while optimizing a given objective function. In our work, we concentrate on the objective of maximizing the user satisfaction function.

We assume to have a set of passengers, U, $|U|= n$, and a set of vehicles, V. Every user, $u \in U$, is assumed to have a travel time $t(u)$ that would take her to reach her destination had she received a direct ride. An assignment P determines which users are assigned to which vehicle, and the route for this vehicle. Let P^v be the set of passengers that are assigned by P to a vehicle $v \in V$, and let t_P^v be the total travel time of vehicle v according to P. Clearly, the travel time of some passengers according to P might be longer than their direct ride's travel time. Let $t_P(u)$ be the travel time of user u according to P.

We assume that the service provider incurs a fixed cost per minute of travel, M, that encapsulates the full operation cost including any desired revenue. For example, if the fuel, tolls and any maintenance costs are estimated at 0.90 dollars per

minute of travel, and the service provider commits to receiving only a 10% percentage of the user payment (as its revenue), M equals $\frac{\$0.90}{1-0.10} = \$1$. Consequently, let $c(u)$ be the cost that a user u would have paid had she received a direct ride, $c(u) = M \cdot t(u)$.

The cost for a user in a shared-ride is determined in proportion to the user's distance from the origin. More formally, let $c_P(u)$ be the cost of a user u according to an assignment P. The cost for a user u that is assigned to vehicle v according to P is determined by the proportional sharing pricing function [26], that is,

$$c_P(u) = \frac{c(u)}{\sum_{u' \in U, u' \in P^v} c(u')} \cdot (M \cdot t_P^v)$$

2.3.1 The Human Satisfaction Function

Our definition of the objective of the assignment problem is to find an assignment P that maximizes the satisfaction of all of the passengers. In order to derive the factors that define the satisfaction function we run a short survey on Mechanical Turk. We asked 30 subjects from the USA what the factors that affect their satisfaction of from using a ride-sharing service are and what factors affect other people's decision to prefer a ride-sharing service over a private taxi ride. We obtained the following 14 factors:

1. The cost of the ride.
2. Travel time.
3. Number of passengers in the vehicle.
4. Seat.
5. Cleanliness of the vehicle.
6. Comfort of the vehicle.
7. Politeness / Friendliness / Behavior / Care / Personality of driver and other passengers.
8. Waiting time.
9. Safety.
10. Laud music or smoking.
11. Own age.
12. Own gender.
13. Own income and wealth.
14. Own social level.

Clearly, some of these factors are not specific to a ride-sharing service, such as cleanliness of the vehicle or its comfort, and some of these factors are not relevant to our setting, such as waiting time. In addition, we felt that it is inappropriate to ask for the exact income and wealth, and we replace this factor with a general question regarding the working status. We therefore consider the following satisfaction factors in our work:

- Travel cost.
- Travel time.
- Number of passengers in vehicle.
- User's seat.
- Working status (employed or unemployed).
- User age.
- User gender.

A passenger may also have other dynamic preferences, which change from one ride to another, but asking the passenger in each ride about all of her preferences may be tedious. Our motivation in this work is to bring about the importance of considering the user satisfaction as an objective, even if it is not 100% accurate, and we experimentally support this claim.

2.4 Satisfaction Model Learned from Humans

In order to develop a more realistic human satisfaction model, we use machine learning techniques based upon data collected from humans. To this end, we solicited 414 human subjects from Mechanical Turk to obtain satisfaction level data. We set nationality to USA, approval rate to at least 98% and did not require the Turkers to be masters. Based on this data, we use deep learning to build a satisfaction model.

2.4.1 Data Collection

The subjects were first asked to provide the following personal details: year of birth, gender and whether they were employed or unemployed.

Our satisfaction model tries to predict the relative satisfaction, that is, how much a passenger traveling by shared-ride is more or less satisfied than the same passenger traveling in a private ride. However, asking users to provide their relative satisfaction is unrealistic, and we thus split every travel scenario into two sub-parts. In the first part we asked the subjects to determine their satisfaction level from a direct private ride to some destination. In the second part the subjects were asked to determine their satisfaction level from a shared ride to the same destination. Specifically, in the first part of each scenario we described a direct private ride with a given time (random number between 5 minutes and one hour) and price (dollar per minute). In the second part of each scenario we described a shared ride to the same destination, where we varied the travel time and cost. Travel time of a shred ride can never be shorter than a direct private ride, and we thus uniformly sampled a number from $\{1, 1.1, 1.2, 1.3, 1.4, 1.5, 1.75, 2, 3, 4\}$ and multiplied it by the direct private travel time. The cost of a shared ride should be lower than the cost of a direct private ride. In the optimal sharing scenario, assuming a 4 passenger vehicle (excluding the driver), there could be 4 passengers traveling to the same destination; in this case the cost will be reduced by a factor of 4. We thus divided the direct private ride's cost by a number uniformly sampled from $\{1.1, 1.2, 1.3, 1.4, 1.5, 1.75, 2, 3, 4\}$. Note that we used a uniform distribution to insure that we collect data from a wide enough range, so that our satisfaction model will be more accurate. In addition, we

randomly sampled the number of additional passengers from $\{1, 2, 3\}$, and we randomly sampled the user's seat from {front passenger, middle back, right back, left back}. The subjects could choose one of seven satisfaction levels on a Likert scale [3], between very satisfied (7) to very dissatisfied (1) with the middle being 'neither satisfied nor dissatisfied' (4).

Each subject was asked a total of six travel scenarios (each with a private and a shared ride), and each subject received different scenarios. In order to eliminate subjects that may be selecting satisfaction levels at random, two out of the six scenarios were sanity check scenarios. In these scenarios the cost of the shared ride was *more* expensive than the private ride, and the travel time was longer. Since it is unreasonable for a person to be more satisfied with such a ride, being both longer and more expensive, we have disqualified any subject who expressed her satisfaction in this question to be higher than her satisfaction from the private ride of that scenario. 131 subjects have failed one of these sanity tests, and were removed from our analysis [1].

26 subjects refused to answer the personal questions and were eliminated from our analysis as well. Of the remaining 257 subjects 147 were females and 110 were males. Their age ranged from 19 to 67 with an average of 32.3 and a median of 31. Regarding their occupation status, 195 were employed while 62 were unemployed. In total, each of the 257 subjects had 4 real scenarios, resulting with 1028 data-points.

2.4.2 Deep-Learning Based Model

Using the collected data we consider deep learning based models with a varying depth to find a good satisfaction model, that is, a model that will accurately predict user satisfaction levels of a new user, based on different features of the user and the user trip. Specifically, the model predicts the relative satisfaction, i.e., how much a passenger traveling by shared-ride is more or less satisfied than the same passenger traveling in a private ride. Formally, let $s_p(u)$ denote the satisfaction level of a user from private ride, and $s_s(u)$ denote the satisfaction level of the same user from a shared ride, the relative satisfaction of the user in a shared ride is given by:

$$r_s(u) = \begin{cases} \frac{s_s(u)-s_p(u)}{7-s_p(u)}, & \text{for } s_p(u) < s_s(u) \\ \frac{s_s(u)-s_p(u)}{s_p(u)-1}, & \text{for } s_s(u) < s_p(u) \\ 0, & \text{for } s_p(u) = s_s(u). \end{cases}$$

Intuitively, the relative satisfaction measures how much the satisfaction of the user has increased (or decreased) from the shared ride in proportion to the maximal possible increase (or decrease). For example, if a user gave a satisfaction level of 5 to a private ride, and a satisfaction of 6 to a shared ride, the relative satisfaction is 0.5, since 6 is half the way from 5 (the baseline satisfaction in this example) to 7 (the maximal satisfaction in the scale). We note that the relative satisfaction value is between -1 and 1.

The input to our network is composed of the following 12 features (following Section 2.3.1): direct travel time ($t(u)$), direct travel cost ($c(u)$), the shared-ride travel time ($t_P(u)$), the shared-ride travel cost ($c_P(u)$), number of passengers in the vehicle ($|P^v|$), 4 seat indicators (front seat, left-back seat, right-back seat, middle-back seat), and the user's working status, age and gender. The output is a value between -1

[1] We use this sanity check for building our satisfaction model, as we believe that doing so results in a more accurate model. However, in Section 2.7, when evaluated against real human subjects, we do not disqualify any of the responses.

Model	Train-error	Validation error
No hidden layers	0.245	0.241
1 hidden layer	0.213	**0.226**
2 hidden layers	0.206	0.235
3 hidden layers	0.204	0.230

TABLE 2.1: Train and validation error for the different model depths. Values indicate the mean squared error (MSE) of each model.

and 1 that is a prediction for the relative satisfaction value. We used a fully connected architecture, with Rectified Linear Unit (ReLU) activation function between the hidden layers. ReLU is defined as:

$$ReLU(x) = \begin{cases} \text{x} & \text{if } x > 0 \\ 0 & \text{otherwise.} \end{cases}$$

Each hidden layer consisted of 100 neurons. The final activation level was set to tanh, as it returns values between -1 and 1. Since our problem is a regression problem in which our model is required to predict a value (rather than a class) we use the mean squared error as our loss function (and as our error measurement). The neural network depth varied from no hidden layers (i.e. only the tanh layer) to 3 hidden layers. We used Adam optimizer [40] with a learning rate of 0.001. We used early stopping [64], i.e., we used the validation set to determine when to stop training. We used 70% of the data-set for training, 15% for validation and 15% for test.

First, we extracted the weights learned by the no hidden layers model, in order to determine the contribution of each of the factors in use. The shared cost factor resulted in the highest absolute weight (-0.035), while the year of birth seemed to have the smallest influence (-0.0001). We then chose the number of hidden layers. Table 2.1 presents the results obtained by each of the models. Since the 1-hidden layer model performed best, we use it as our satisfaction function. This model achieved a mean squared error of 0.240 on the test set. Finally, we tested the contribution of the factors with the chosen 1-hidden layer model. Similar to our previous finding, removing the shared cost factor resulted in the highest increase in the mean squared error on the test set (0.357).

2.5 Assignment Algorithms

In order to demonstrate the importance of the user satisfaction, we introduce a stochastic algorithm for finding an assignment for our ridesharing problem. We assume that there are a sufficiently large number of vehicles so that any request could be satisfied. Indeed, since in this thesis we study the last-mile problem, this assumption holds in many real-world situations. For example, in many airports there is an agreement between the airport authorities and the official taxi company that ensures that there will always be a sufficient number of vehicles available.

2.5.1 Stochastic Merging-Based Satisfaction Algorithm (Simsat)

We propose a practical algorithm for assigning passengers to vehicles with the objective of maximizing the satisfaction of all of the passengers: Simsat (Algorithm 2). However, we begin with a deterministic variant of Simsat, M-Sat (Algorithm

1). M-Sat runs Floyd-Warshall [27, 80] on the graph at the initialization, to obtain the minimal travel time between every two vertices. M-Sat then runs its main procedure as follows. The algorithm starts with n vehicles and assigns every single passenger to a unique vehicle. It then considers all assignments P' that result from merging any pair of vehicles into a single vehicle. The assignments P' use the nearest neighbor algorithm [38] for ordering the passengers drop-offs based upon the Floyd-Warshall matrix (the greedy approach). For every new assignment P', M-Sat computes the difference in the satisfaction level, $s_{P'} - s_P$. The satisfaction of an assignment is computed with the function SatFunc, which returns the sum of satisfaction of all passengers in a given vehicle. M-Sat considers all the assignments P' where $s_{P'} - s_P$ is non-negative, and picks the assignment P' that maximizes $s_{P'} - s_P$. M-Sat continues selecting and merging pairs of vehicles until all assignments P' results in a decrease in the satisfaction level. We note that the efficiency of Algorithm 1 may heavily depend on the number of calls to SatFunc function. For clarity purpose, in each step Algorithm 1 re-computes the satisfaction of passengers in a given vehicle, and re-evaluates the satisfaction of many assignments that have been also considered in the previous steps. Therefore, in its current presentation Algorithm 1 calls the function SatFunc at most n^3 times. However, in practice, M-Sat stores the satisfaction values that the SatFunc function computes and the deltaSat values, and thus SatFunc is called at most $2n$ times.

Algorithm 1: M-Sat

```
Input: a graph with source vertex,
a set of passengers,
a satisfaction function that returns the sum of satisfaction of all passengers in a
  given vehicle (SatFunc).
Result: An assignment of all passengers to vehicles.
compute Floyd-Warshall on the graph
Cabs ← assign each passenger to a unique vehicle
while (true) do
    maxSat ← -1
    for every cabA, cabB from Cabs do
        mergedCab ← cabA ∪ cabB
        deltaSat ← SatFunc(mergedCab) - (SatFunc(cabA) + SatFunc(cabB))
        if deltaSat > maxSat then
            maxSat ← deltaSat
            maxCabA ← cabA
            maxCabB ← cabB
            maxMergedCab ← mergedCab
        end
    end
    if maxSat ≥ 0 then
        add maxMergedCab to Cabs
        delete maxCabA, maxCabB from Cabs
    end
    else
        return Cabs
    end
end
```

Simast begins with running M-Sat, as a baseline. It then repeats M-Sat's main

procedure multiple times (n^2), but instead of selecting the assignment P' that maximizes $s_{P'} - s_P$, it randomly picks an assignment P' proportionate to $e^{(s_{P'} - s_P)}$. Once Simsat completes all of the iterations, the assignment that yields the maximal total satisfaction is selected. The number of times the main procedure is repeated in Simsat can vary; the more times it is repeated the higher the expected performance. Therefore, Simsat is an any-time algorithm. As with M-Sat, Algorithm 2 is a non-optimized version of SimSat, since SimSat stores the satisfaction values that the SatFunc function computes and the deltaSat values from previous steps.

2.5.2 Brute-Force Algorithm

We use the following brute-force algorithm to compute the assignment that maximizes a given objective function (i.e., a possible user satisfaction model). First, the algorithm runs Floyd-Warshell on the graph. The algorithm then solves a coin-change problem [35, p. 171] to obtain all possible ways to split the number of passengers into vehicles. For example, when $n = 10$, we get $\{3, 3, 3, 1\}$, $\{2, 2, 2, 2, 2\}$, $\{4, 4, 2\}$ etc. For each splitting option the algorithm iterates over all possible assignments. We explicitly handle multiple vehicles with the same number of passengers, since it does not matter if a specific group of passengers travels in one vehicle or another. For example, for a group of $\{4, 3, 3, 2, 2, 2\}$, the algorithm first iterates over all assignments of 4 passengers (there are $\binom{n}{4}$ such assignments), then, recursively calls the assignment function with $\{3, 3, 2, 2, 2\}$ and the remaining passengers. The recursive call iterates over all possible assignments of three people to each of the next two vehicles, and preforms a recursive call with the remaining vehicles and passengers. For each vehicle, the algorithm computes all possible options for seating and for dropping off its passengers (this is done once for each set of users), and selects the seating and travel order that maximize the objective function. Clearly, this algorithm is computationally and memory extensive.

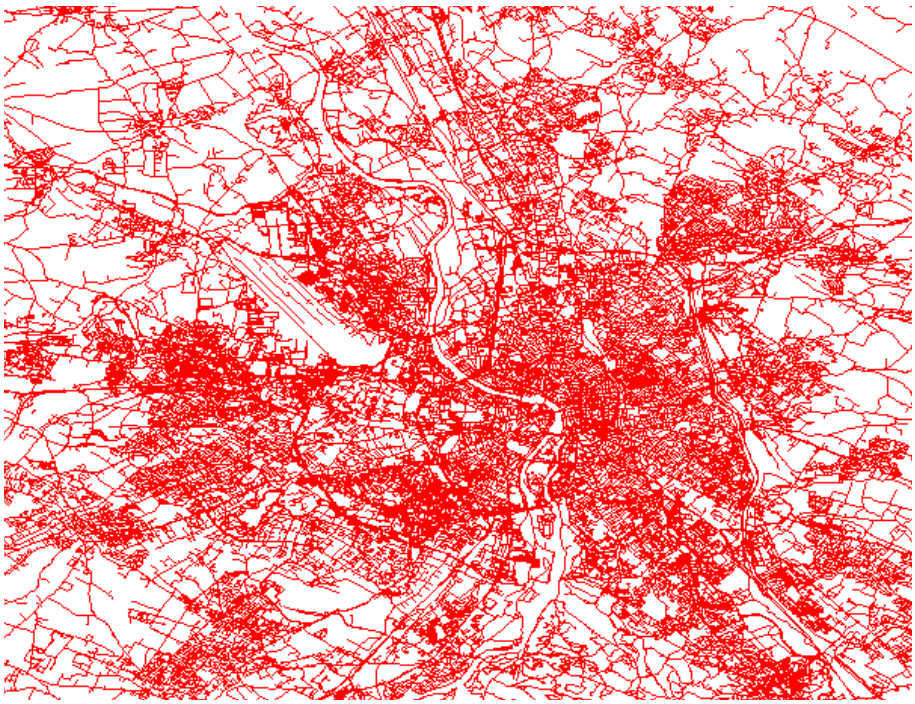

FIGURE 2.1: A graph created from a map of the city of Toulouse, France.

Algorithm 2: Simsat

Input: a graph with source vertex,
a set of passengers,
a satisfaction function that returns the sum of satisfaction of *all* passengers in a given vehicle (SatFunc).
Result: An assignment of all passengers to vehicles.

```
maxP ← the assignment returned by M-Sat
maxSat ← satisfaction of assignment maxP (i.e. s_maxP)
for 1 to n^2 do
    Cabs ← assign each passenger to a unique vehicle
    while (true) do
        DeltaSat[·, ·] ← -1
        for every cabA, cabB from Cabs do
            mergedCab ← cabA ∪ cabB
            deltaSat ← SatFunc(mergedCab) - (SatFunc(cabA) + SatFunc(cabB))
            if deltaSat ≥ 0 then
                DeltaSat[cabA,cabB] ← deltaSat
            end
        end
        if DeltaSat contains a non-negative value then
            choose cabA, cabB ∝ e^DeltaSat[cabA,cabB]
            add cabA ∪ cabB to Cabs
            delete cabA, cabB from Cabs
        end
        else
            break
        end
    end
    currSat ← satisfaction of assignment Cabs
    if currSat > maxSat then
        maxP ← Cabs
        maxSat ← currSat
    end
end
return maxP
```

2.6 Experimental Evaluation

2.6.1 Experimental Settings

For the data we use the graph of the city of Toulouse, France[2] as presented in Figure 2.1. We used the Toologize module of the MicroCity 1.15 GIS program[3] for generating the nodes from the city map. The generated graph includes the actual distances between the different vertices. The graph also includes the Toulouse-Blagnac airport. We cropped the graph to $40,000$ vertices, by running Dijkstra algorithm [23] starting at the airport, sorting all vertices by their distance from the airport, and removing all farther away vertices (including those that are unreachable).

Being a last-mile problem, we set the origin vertex to be the same for all passengers, the Toulouse-Blagnac airport. The destination vertices were randomly sampled for every passenger using a uniform distribution over all vertices (i.e. we uniformly sampled a number between 1 and the number of nodes in the graph). Note that since we use a uniform distribution, nodes within dense regions (e.g. the city center) were sampled more often as destinations. To convert the distances to travel times we set the average speed to 30 kph. We also set the cost per minute of travel, M, to $\$1$. The capacity of each vehicle was set to 4 passengers.

2.6.2 Simulation Results

In this section we compare the performance of the following five methods in terms of relative user satisfaction (as obtain from the satisfaction model) in simulation:

1. **Optimal:** The brute-force algorithm with the learned satisfaction function as the objective function.
2. **M-Sat:** with the learned satisfaction function.
3. **Simsat:** with the learned satisfaction function.
4. **Cost:** The brute-force algorithm that uses the travel cost only as the objective function. Note that this algorithm does not require enumerating over the possible seating options.
5. **Time:** An algorithm that uses the travel time only as the objective function. Clearly, this algorithm has trivial behavior; it assigns a private vehicle to each passenger.

All the methods were evaluated with the learned satisfaction function, regardless of the function actually used by the method. Since the brute-force algorithm is computationally expensive, we did not run Optimal and Cost methods with more than 12 passengers. However, we evaluated the performance of M-Sat, Simsat and Time methods with up to 30 passengers.

Figures 2.2 and 2.3 present our results. The results were obtained by averaging over 1000 samples of passenger destinations. Note that the Time assignment yields a constant user satisfaction of 0 since it assigns a private vehicle to each and every passenger. Due to the high volume of the data, all differences between *any* two methods are statistically significant ($p < 0.0001$).

[2]obtained from https://www.geofabrik.de/data/shapefiles_toulouse.zip
[3]https://microcity.github.io/index.html

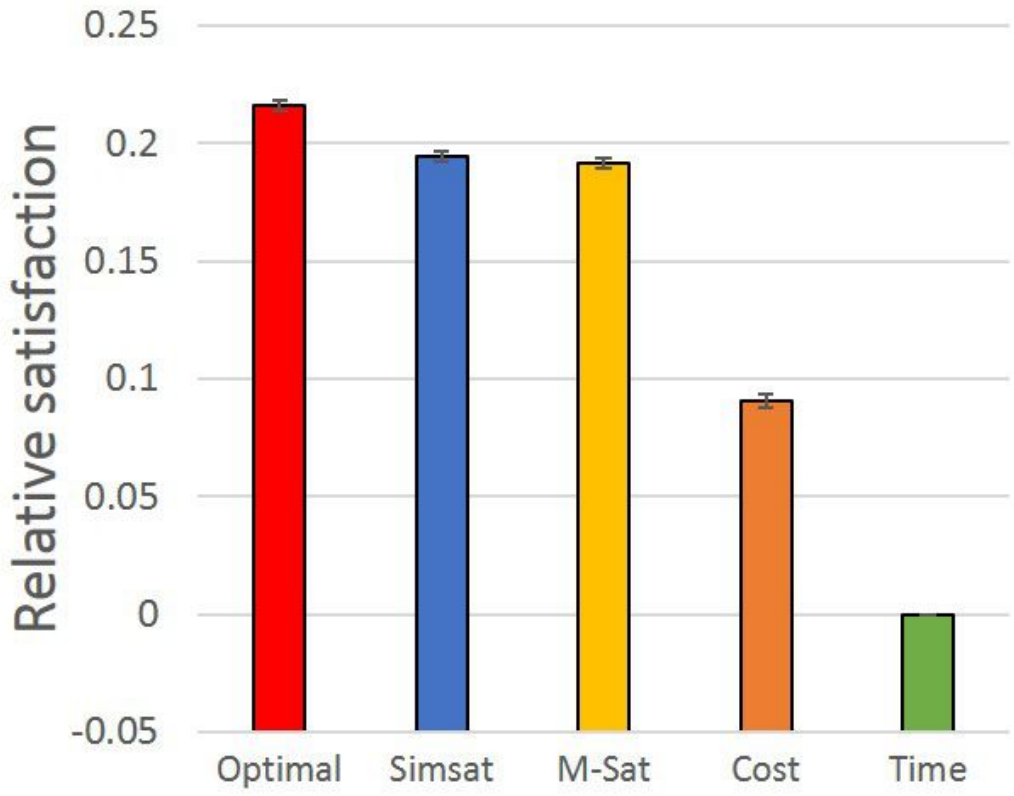

FIGURE 2.2: Relative satisfaction in simulation for each of the assignment methods, averaged on 1000 assignments and 2-12 passengers. Error bars present 95% confidence interval.

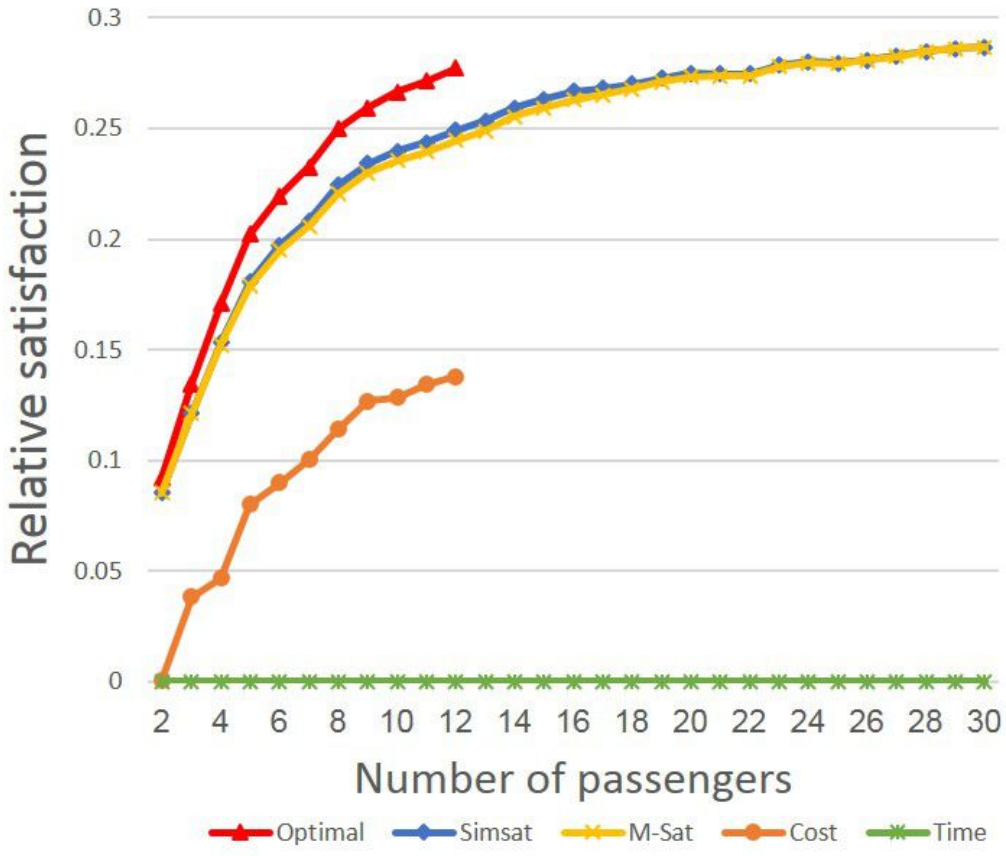

FIGURE 2.3: Relative satisfaction in simulation for each of the assignment methods, averaged on 1000 assignments, for 2-30 passengers.

As depicted in the figures, our satisfaction oriented assignment method (Simsat) obtains results that are very close to the optimal assignment. Simsat's average satisfaction level is much closer to the optimal assignment than that of the Cost and Time assignments, which are brute-force assignments that use a simpler user-satisfaction model. Figure 2.4 depicts the average running time of Simsat and M-sat in comparison to that of Optimal and Cost, which are brute-force algorithms. The running time of the Cost assignment is slightly lower than that of the Optimal assignment since the Cost assignment does not enumerate over the possible seating options. Clearly, the performance of both Simsat and M-sat is much better than that of the brute-force algorithms and both achieve very reasonable execution times and thus can be used in practice.

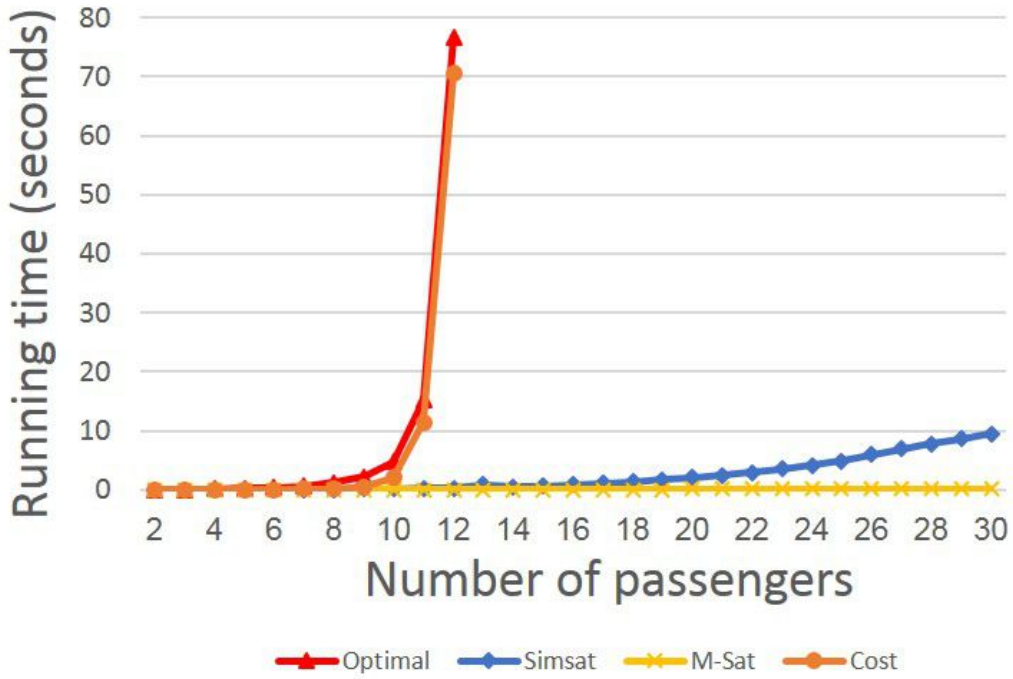

FIGURE 2.4: Running time (in seconds) for a single assignment, averaged on 1000 assignments, for 2-30 passengers.

We also tested the efficiency of the methods in terms of travel time and cost. The results are shown in Table 2.2. As expected, the Time method resulted in the lowest average travel time, but with the highest average cost. Similarly, the Cost method resulted in the lowest average cost, but with the highest average travel time. Our methods, as well as the optimal method, seem to balance the travel time and travel cost. Note that our methods do not aim at balancing the travel time and cost explicitly. Instead, they try to maximize the user satisfaction, and as a result they generate assignments that balance the travel time and cost.

In summary, our results indicate that it is more important to obtain a richer model of user satisfaction, than improving the performance of the assignment algorithm. That being said, we do not disregard the importance of improving the performance of the assignment algorithm and do intend to pursue additional algorithms that may perform better.

	Average Time	Average Cost
Optimal	16:31 minutes	\$10.07
Simsat	16:02 minutes	\$10.30
M-Sat	16:00 minutes	\$10.36
Cost	18:21 minutes	\$9.33
Time	14:38 minutes	\$14.64

TABLE 2.2: Travel time and cost, averaged on 1000 assignments and 2-12 passengers.

2.7 Evaluation With Humans

In this section we compare the performance of Simsat against that of Cost and Time assignments, in terms of satisfaction, as reported by human subjects. To this end, we uniformly sampled destinations for 12 passengers from the map of the city of Toulouse. The passengers were assigned to vehicles according to Simsat, Cost, and Time methods. This process was repeated 21 times, resulting in 252 travel scenarios, that is, 252 passengers and their destinations.

We recruited 84 human subjects from Mechanical Turk. We set nationality to USA, approval rate to at least 98% and did not require the Turkers to be masters. Each subject was asked a total of 3 travel scenarios, where in each travel scenario the user was asked about her satisfaction from the 3 possible assignments: that of Simsat, that of Cost and that of Time (which is in fact a private ride). Overall, each user was asked 9 questions about her satisfaction. The subjects could choose one of seven satisfaction levels on a Likert scale, between very satisfied (7) to very dissatisfied (1) with the middle being 'neither satisfied nor dissatisfied' (4).

Figure 2.5 presents our results. The results were obtained by averaging over the 252 travel scenarios. As depicted by the figure, Simsat has significantly outperformed both the Cost and Time methods in terms of user satisfaction ($p < 0.05$).

2.8 Discussion

There are two key insights from our results. First, people are more satisfied (on average) from using a satisfaction oriented ride-sharing service, than using a private ride (i.e, the Time assignment). More importantly, people are more satisfied (on average) from using a satisfaction oriented ride-sharing service, than using a cost oriented ride-sharing service (i.e, the Cost assignment). A possible critique of our model is that it is hard to learn a good satisfaction model of people, and having a single satisfaction function that aggregates all the factors is not reasonable. Another possible critique is that there are several factors that could affect the user's satisfaction, so how are we sure that we chose the correct ones? Our results Section 2.7 prove that our approach is valid and useful: even if we did not choose the best factors and even if a single satisfaction function is too simplistic, our evaluation with humans shows that using our satisfaction oriented assignment with a greedy algorithm outperforms a brute-force algorithm with cost-based assignment.

There is another general critique on our approach; one may wonder why we focus only on user satisfaction for achieving a widespread adaption of ride-sharing services. Indeed, there are several other factors that affect the adaptation of ride-sharing services, for example, education and technology advances. However, it is a

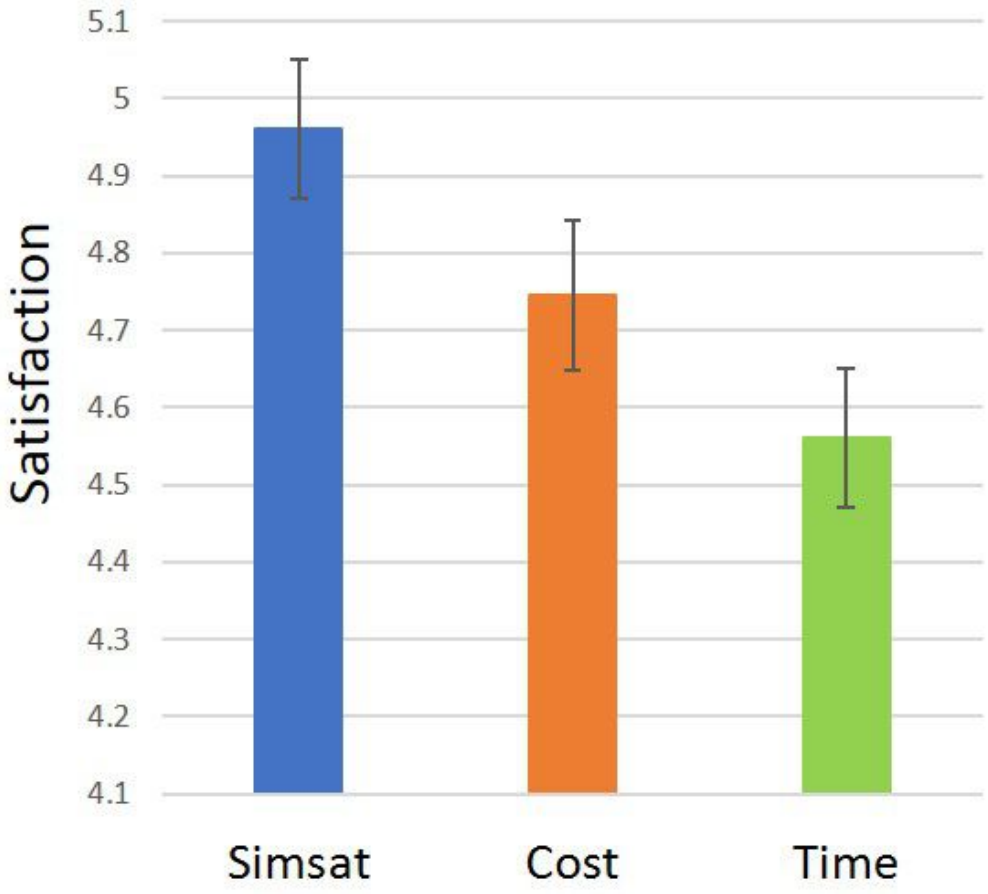

FIGURE 2.5: Average satisfaction for each of the three assignment methods, as reported by the human subjects. Error bars present 95% confidence interval.

well known marketing concept that customer satisfaction has a very strong impact on adaptation, loyalty and retention (see, for example, [74]).

2.9 Conclusions and Future Work

The importance of our work lies in the fact that we exclusively concentrate on a rich and realistic general function of user satisfaction as the objective, which is (arguably) the most important aspect to consider for achieving a widespread adaption of ridesharing services. We use deep learning to model user satisfaction based upon data collected from actual human subjects. We present a satisfaction oriented assignment method (Simsat), and show that it outperforms brute-force assignments using simpler user-satisfaction models. These results indicate that it is more important to obtain a richer model of user satisfaction, than improving the performance of the assignment algorithm.

In future work we intend to extend our model from the last-mile setting to the more general ridesharing scenario, where people may have different origins. We also intend to build a game that will simulate an actual ride for the subjects; this should allow us to obtain more exact satisfaction levels. This game could include additional travel information such as the other passengers in the trip, and allow the subject to select her seat when entering a vehicle. Since users will be playing the game more than once, the satisfaction model can be further improved by personalization, taking into account user's feedback on previous rounds.

Chapter 3

Fairly Splitting the Ride Cost using the Shapley Value

3.1 Introduction

Most works in the domain of ride-sharing are dedicated to the assignment of passengers to vehicles, or to planning optimal drop-off routes [66, 5, 49]. In this chapter we study a fair allocation of the cost of the shared ride in the last mile variant [17]. We concentrate on the Shapley value [72] as our notion of fair cost allocation. The Shapley value is widely used in cooperative games, and is the only cost allocation satisfying efficiency, symmetry, null player property and additivity. The Shapley value has been even termed the most important normative division scheme in cooperative game theory [81]. However, the Shapley value depends on the travel cost of a ride of each subset of the passengers. Therefore, as stated by Özener and Ergun [55], "In general, explicitly calculating the Shapley value requires exponential time. Hence, it is an impractical cost-allocation method unless an implicit technique given the particular structure of the game can be found".

There are two possible general structures of the last-mile ride-sharing problem. In some cases there is a priority order in which the passengers are dropped-off. Such prioritization may be attributed to the order in which the passengers arrived at the origin location, or the frequency of passenger usage of the service; the latter is similar to the different boarding groups on an aircraft. Other rationales for prioritization may include urgency of arrival or priority groups in need (e.g. elderly, disabled, pregnant women, and the injured). Clearly, in such cases, the prioritization is preserved when determining the travel cost of a ride with a subset of the passengers. We denote this problem as the *prioritized ride-sharing problem*. Indeed, in some scenarios there is no predetermined prioritization order. In such cases it is assumed that a ride with a subset of the passengers is performed using the shortest (or cheapest) path that traverses their destinations. We denote this problem as the *non-prioritized ride-sharing problem*.

The prioritized and the non-prioritized ride-sharing problems are closely related to traveling salesman games [63]. In these games, a service provider makes a round-trip along the locations of several sponsors, where the total cost of the trip should be distributed among the sponsors. Specifically, the prioritized ride-sharing problem is similar to the fixed-route traveling salesman game, also known as routing game [83], while the non-prioritized ride-sharing problem is similar to the traveling salesman game. Most of the works on traveling salesman games concentrated on finding an element of the core, a solution game concept which is different from the Shapley value. One exception is the work of Yengin [83], who has studied the Shapley value of routing games and has conjectured that there is no efficient way for computing the Shapley value in routing games.

In this chapter, we show an efficient computation of the Shapley value for the prioritized ride-sharing problem. Our method is based on smart enumeration of the components that are used in the computation of the Shapley value. Furthermore, our approach can be generalized to routing games, and we thus also provide an efficient way for computing the Shapley value in routing game. We then move to analyze the non-prioritized ride-sharing problem and show that, unless P=NP, there is no polynomial time algorithm for computing the Shapley value. Fortunately, we show through extensive simulations that when the given travel path is the shortest path the Shapley value of the prioritized ride-sharing problem can be used as an excellent proxy for the Shapley value of the non-prioritized ride-sharing problem.

We note that the context of our work is that the assignment of the passengers to the vehicle has already been determined, either by a ride-sharing system or by the passengers themselves, and we only need to decide on the cost allocation. Since we focus on the case where the assignment has already been determined, we do not consider the ability of passengers to deviate from the given assignment and join a different vehicle, which is acceptable since either they want to travel together or no other alternative exists.

To summarize, the contributions of this chapter are two-fold:

1. We show an efficient method for computing the Shapley value of each user in a shared-ride when the priority order is predetermined. Our solution entails that the Shapley value can be computed in polynomial time in routing games as well, which is in contrast to a previous conjecture made.

2. We show that, while there exist no polynomial algorithm for computing the Shapley value of the non-prioritized ride-sharing problem (unless P=NP), the Shapley value of the prioritized ride-sharing problem can be used as an excellent proxy for the Shapley value of the non-prioritized ride-sharing problem (under the assumption that the provided travel path is the shortest path).

3.2 Related Work

The ride-sharing cost allocation problems that we study are closely related to traveling salesman games [63]. Specifically, the prioritized ride-sharing problem is similar to the fixed-route traveling salesman game [26, 63, 10], also known as routing game [83].

One variant of routing game is the fixed-route traveling salesman problems with appointments. In this variant the service provider is assumed to travel back home (to the origin) when she skips a sponsor. This variant was introduced by Yengin [83], who also showed how to efficiently compute the Shapley value for this problem but stated that his technique does not carry over to routing games.

The prioritized ride-sharing problem can also be interpreted as a generalization of the airport problem [45] to a two dimensional plane. In the airport problem we need to decide how to distribute the cost of an airport runway among different airlines who need runways of different lengths. In our case we distribute the cost among passengers who need rides of different lengths and destinations. It was shown that the Shapley value can be efficiently computed for the airport problem, however achieving efficient computation of the Shapley value in our setting requires a different technique.

The Shapley value for the traveling salesman game, which is related to our non-prioritized ride-sharing problem, has rarely received serious attention in the literature, due to its computational complexity. Notably, Aziz et al. [9] suggested a number of direct and sampling-based procedures for calculating the Shapley value for the traveling salesman game. They further surveyed several proxies for the Shapley value that are relatively easy to compute, and experimentally evaluate their performance. We develop a proxy for the Shapley value for the non-prioritized ride-sharing problem which is based on the Shapley-value for the prioritized ride-sharing problem, and compare its performance with proxies that are based on the work of Aziz et al.

The problem of fair cost allocation was also studied in the context of logistic operation. In this domain, shippers collaborate and bundle their shipment requests together to achieve better rates from a carrier [32]. The Shapley value was also investigated in this domain of collaborative transportation [28, 76]. In particular, Özener and Ergun [55] stated that "we do not know of an efficient technique for calculating the Shapley value for the shippers' collaboration game". Indeed, Fiestras-Janeiro et al. [25] developed the line rule, which is inspired by the Shapley value, but requires less computational effort and relates better with the core. However, the line rule is suitable for a specific inventory transportation problem. Özener [54] describes an approximation of the Shapley Value when trying to simultaneously allocate both the transportation costs and the emissions among the customers. Overall, we note that the main requirements from a cost allocation in the context of logistic operation is stability, and an equal distribution of the profit, since the collaboration is assumed to be long-termed. The type of interaction is our setting is inherently different, as it is an ad-hoc short term collaboration.

In another domain, Bistaffa et al. [11] introduce a fair payment scheme, which is based on the game theoretic concept of the kernel, for the social ride-sharing problem (where the set of commuters are connected through a social network).

3.3 Preliminaries

We are given a weighted graph $G(V, E)$ that represents a road network; V is the set of possible locations, and E is a set of weighted edges that represents the set of roads. We are given a set $U = \{u_1, u_2, ..., u_n\}$ of passengers (users) that depart from the same origin, $d_0 \in V$. Without loss of generality, we assume that passenger u_1 will be dropped-off first, passenger u_2 will be dropped off second, etc. Each passenger u_i has a corresponding destination $d_i \in V$. Let $D \subset V$ be the set of destinations, $D = \{d_1, d_2, ..., d_n\}$. We denote by $\delta(d_i, d_j)$ the shortest travel distance between d_i and d_j in G and $\delta(d_i, d_i) = 0$. To simplify the notation we define a dummy destination, d_{n+1}, such that for every $i \in \{0, 1, ..., n\}$, $\delta(d_i, d_{n+1}) = 0$. Given a set $S \subseteq D$, let $c(S)$ be the cost associated with the subset S. That is, $c(D)$ is the total travel cost of the shared ride. We note that $c(S)$, where $S \subsetneq D$, depends on the order in which the passengers are dropped off, and therefore $c(S)$ is defined differently in the prioritized ride-sharing problem and in the non-prioritized ride-sharing problem. The Shapley value for a passenger u_i is formally defined as:

$$\phi(u_i) = \sum_{S \subseteq U \setminus \{i\}} \frac{|S|!\,(|U|-|S|-1)!}{|U|!}(c(S \cup \{i\}) - c(S)).$$

That is, the Shapley value is an average over the marginal costs of each passenger.

3.4 The Prioritized Ride-sharing Problem

In this section we assume that the passengers are ordered according to some predetermined priority order, and efficiently compute the payment for every passenger using the Shapley value. Unlike other related work [63], we do not require that the priority order will be the optimal order that minimizes the total cost.

3.4.1 Notation

Given the set of passengers U, without loss of generality, we assume that passenger u_1 has the highest priority, passenger u_2 has the second highest priority, etc. Given a set $S \subseteq D$, let $\tilde{S}$ be the set S ordered in an ascending order (according to the priority order), and let $S[i]$ be the destination that is in the i-th position in $\tilde{S}$. For ease of notation we use $S[0]$ to denote d_0 and $S[|S|+1]$ to denote d_{n+1}.

Given a set $S \subseteq D$, let $v(S)$ be the shortest travel distance of the path that starts at the origin d_0 and traverses all destinations $d_i \in S$ according to an ascending order. That is, $v(S) = \sum_{i=0}^{|S|-1} \delta(S[i], S[i+1])$. This value ($v(S)$) serves as the cost associated with a subset of passengers, $c(S)$, in the computation of the Shapley value.

Let R be a permutation on D and let P_i^R be the set of the previous destinations to d_i in permutation R.

3.4.2 Efficient Computation of the Shapley Value

We are interested in determining the payment for each passenger, u_i, according to the Shapely value, $\phi(u_i)$. The Shapley value has several equivalent formulas, and we use the following formula to derive an efficient computation in the prioritized ride-sharing problem:

$$\phi(u_i) = \frac{1}{n!} \sum_R \left(v(P_i^R \cup \{d_i\}) - v(P_i^R) \right).$$

Given a permutation R and a passenger u_i, let $d_l \in P_i^R$ be a destination such that $l < i$ and $\forall d_j \in P_i^R, j \leq l$ or $i < j$. If no such destination exists, then d_l is defined as d_0. Similarly, let $d_r \in P_i^R$ be a destination such that $i < r$ and $\forall d_j \in P_i^R, j < i$ or $r \leq j$. If no such destination exists, then d_r is defined as d_{n+1}. We use ℓ (and r) to denote the position of d_l (and d_r) in the ordered $\tilde{P_i^R}$, respectively. If $d_l = d_0$ then $\ell = 0$, and if $d_r = d_{n+1}$ then $r = |P_i^R|+1$. We note that $P_i^R[\ell] = d_l$, $P_i^R[r] = d_r$ and $r = \ell + 1$.

For example, assume $D = \{d_1, d_2, d_3, d_4, d_5, d_6\}$ and $R = \{d_6, d_2, d_5, d_4, d_3, d_1\}$, we get $P_4^R = \{d_6, d_2, d_5\}$ and thus $\tilde{P_4^R} = \{d_2, d_5, d_6\}$, $d_l = d_2$ (i.e., $\ell = 1$), $d_r = d_5$ (i.e., $r = 2$), and $P_4^R[\ell] = d_2$.

Our first observation is that the computation of the Shapley value in our setting, $\phi(u_i)$, may be written as the sum over the distances between pairs of destinations.

Observation 3.4.1. $\phi(u_i) = \frac{1}{n!} \sum_{p=0}^{n-1} \sum_{q=p+1}^{n} \alpha_{p,q}^i \delta(d_p, d_q)$, *for some* $\alpha_{p,q}^i \in \mathbb{Z}$.

Proof. We note that $\phi(u_i) \cdot n!$ is a sum over $v(S)$ for multiple $S \subseteq D$. By definition, $v(S) = \sum_{j=0}^{|S|-1} \delta(S[j], S[j+1])$, such that $S[j] = d_p$ and $S[j+1] = d_q$ where $p < q$. □

We now show that we can rewrite the computation of the Shapely value in our setting as follows.

Lemma 3.4.2.

$$\phi(u_i) = \frac{1}{n!} \sum_{R} \Big(\delta(d_l, d_i) + \delta(d_i, d_r) - \delta(d_l, d_r) \Big)$$

Proof.

$$v(P_i^R) = \sum_{j=0}^{|P_i^R|-1} \delta(P_i^R[j], P_i^R[j+1]) =$$

$$\sum_{j=0}^{\ell-1} \delta(P_i^R[j], P_i^R[j+1]) + \delta(d_l, d_r) + \sum_{j=r}^{|P_i^R|-1} \delta(P_i^R[j], P_i^R[j+1])$$

In addition,

$$v(P_i^R \cup \{d_i\}) = \sum_{j=0}^{\ell-1} \delta(P_i^R[j], P_i^R[j+1]) +$$

$$\delta(d_l, d_i) + \delta(d_i, d_r) + \sum_{j=r}^{|P_i^R|-1} \delta(P_i^R[j], P_i^R[j+1]).$$

By definition,

$$\phi(u_i) = \frac{1}{n!} \sum_{R} \big[v(P_i^R \cup \{d_i\}) - v(P_i^R) \big] =$$

$$\frac{1}{n!} \sum_{R} \Bigg(\sum_{j=0}^{\ell-1} \delta(P_i^R[j], P_i^R[j+1]) + \delta(d_l, d_i) + \delta(d_i, d_r) + \sum_{j=r}^{|P_i^R|-1} \delta(P_i^R[j], P_i^R[j+1]) - \Big(\sum_{j=0}^{\ell-1} \delta(P_i^R[j], P_i^R[j+1]) + \delta(d_l, d_r) + \sum_{j=r}^{|P_i^R|-1} \delta(P_i^R[j], P_i^R[j+1]) \Big) \Bigg) =$$

$$\frac{1}{n!} \sum_{R} \Big(\delta(d_l, d_i) + \delta(d_i, d_r) - \delta(d_l, d_r) \Big)$$

□

Following Observation 3.4.1 and Lemma 3.4.2 we now show that we can rewrite the computation of the Shapely value as a sum over distances, that can be computed in polynomial time.

Theorem 3.4.3. *For each i, $\phi(u_i) = \sum_{p=0}^{i} \sum_{q=i}^{n} \beta_{p,q}^{i} \delta(d_p, d_q)$, where $q \neq p$, and $\beta_{p,q}^{i} \in \mathbb{Q}$ are computed in polynomial time.*

Proof. By definition, $l < i < r$. According to Lemma 3.4.2 $\phi(u_i) \cdot n!$ is a sum over $\delta(d_p, d_q)$, where $p \leq i \leq q$. There are several terms in this sum:

- $\beta_{0,i}^{i}$ multiplies $\delta(d_0, d_i)$. Now, $\delta(d_0, d_i)$ appears in $\phi(u_i)$ in every permutation R when $d_l = d_0$. That is, in all of the permutations where destination d_i appears before any other destination d_x such that $x < i$. We now count the number of such permutations. There are $\binom{n}{i}$ options to place the destinations $d_1, d_2, ..., d_i$ among the n available destinations. For each such option there are $(i-1)!$ options to order the destinations $d_1, d_2, ..., d_i$ such that d_i is the first destination among them. Finally, there are $(n-i)!$ options to order the destinations

$d_{i+1}, d_{i+2}, ..., d_n$. Therefore, $\delta(d_0, d_i)$ appears in $\binom{n}{i} \cdot (i-1)! \cdot (n-i)! = \frac{n!}{i}$ permutations, and by inserting $\frac{1}{n!}$ into the sum we get that $\beta^i_{0,i} = \frac{1}{i}$.

- For each $q > i$, $\beta^i_{0,q}$ multiplies $\delta(d_0, d_q)$. Now, $\delta(d_0, d_q)$ appears negatively in $\phi(u_i)$ in every permutation R when $d_l = d_0$ and $d_r = d_q$. That is, in all of the permutations where destination d_q appears before d_i (i.e., $d_q \in P^R_i$), but any other destination d_x such that $x < q$, appears after d_i. We now count the number of such permutations. There are $\binom{n}{q}$ options to place the destinations $d_1, d_2, ..., d_i, ..., d_q$ among the n available destinations. For each such option there are $(q-2)!$ options to order the destinations $d_1, d_2, ..., d_i$ such that d_q is the first destination and d_i is the second destination among them. Finally, there are $(n-q)!$ options to order the destinations $d_{q+1}, d_{q+2}, ..., d_n$. Therefore, $\delta(d_0, d_q)$ appears negatively in $\binom{n}{q} \cdot (q-2)! \cdot (n-q)! = \frac{n!}{q \cdot (q-1)}$ permutations, and by inserting $\frac{1}{n!}$ into the sum we get that $\beta^i_{0,q} = -\frac{1}{q \cdot (q-1)}$.

- For each $0 < p < i$, $\beta^i_{p,i}$ multiplies $\delta(d_p, d_i)$. Now, $\delta(d_p, d_i)$ appears in $\phi(u_i)$ in every permutation R when $d_l = d_p$. That is, in all of the permutations where destination d_p appears before d_i (i.e., $d_p \in P^R_i$), but any other destination d_x such that $p < x < i$, appears after d_i. We now count the number of such permutations. There are $\binom{n}{i-p+1}$ options to place the destinations $d_p, d_{p+1}, ..., d_i$ among the n available destinations. For each such option there are $(i-p+1-2)!$ options to order the destinations $d_p, d_{p+1}, ..., d_i$ such that d_p is the first destination and d_i is the second destination among them. Finally, there are $(n-(i-p+1))!$ options to order the destinations $d_1, d_2, ..., d_{p-1}, d_{i+1}, d_{i+2}, ..., d_n$. Therefore, $\delta(d_p, d_i)$ appears in $\binom{n}{i-p+1} \cdot (i-p-1)! \cdot (n-(i-p+1))! = \frac{n!}{(i-p) \cdot (i-p+1)}$ permutations, and by inserting $\frac{1}{n!}$ into the sum we get that $\beta^i_{p,i} = \frac{1}{(i-p) \cdot (i-p+1)}$.

- For each $q > i$, $\beta^i_{i,q}$ multiplies $\delta(d_i, d_q)$. Now, $\delta(d_i, d_q)$ appears in $\phi(u_i)$ in every permutation R when $d_r = d_q$. That is, in all of the permutations where destination d_q appears before d_i (i.e., $d_q \in P^R_i$), but any other destination d_x such that $i < x < q$, appears after d_i. We now count the number of such permutations. There are $\binom{n}{q-i+1}$ options to place the destinations $d_i, d_{i+1}, ..., d_q$ among the n available destinations. For each such option there are $(q-i+1-2)!$ options to order the destinations $d_i, d_{i+1}, ..., d_q$ such that d_q is the first destination and d_i is the second destination among them. Finally, there are $(n-(q-i+1))!$ options to order the destinations $d_1, d_2, ..., d_{i-1}, d_{q+1}, d_{q+2}, ..., d_n$. Therefore, $\delta(d_p, d_i)$ appears in $\binom{n}{q-i+1} \cdot (q-i-1)! \cdot (n-(q-i+1))! = \frac{n!}{(q-i) \cdot (q-i+1)}$ permutations, and by inserting $\frac{1}{n!}$ into the sum we get that $\beta^i_{i,q} = \frac{1}{(q-i) \cdot (q-i+1)}$.

- For each p, q such that $p < i < q$, $\beta^i_{p,q}$ multiplies $\delta(d_p, d_q)$. Now, $\delta(d_p, d_q)$ appears negatively in $\phi(u_i)$ in every permutation R when $d_l = d_p$ and $d_r = d_q$. That is, in all of the permutations where destinations d_p, d_q appear before d_i (i.e., $d_p, d_q \in P^R_i$), but any other destination d_x such that $p < x < q, x \neq i$, appears after d_i. We now count the number of such permutations. There are $\binom{n}{q-p+1}$ options to place the destinations $d_p, d_{p+1}, ..., d_i, ..., d_q$ among the n available destinations. For each such option there are $(q-p+1-3)!$ options to order the destinations $d_p, d_{p+1}, ..., d_i, ..., d_q$ such that d_p is the first destination, d_q is the second and d_i is the third destination among them. Similarly, there are $(q-p+1-3)!$ options to order these destinations such that d_q is the first destination, d_p is the second and d_i is the third. Finally, there are $(n-(q-p+1))!$ options to order the destinations $d_1, d_2, ..., d_{p-1}, d_{q+1}, d_{q+2}, ..., d_n$. Therefore, $\delta(d_p, d_q)$

appears in $\binom{n}{q-p+1}\cdot 2\cdot(q-p-2)!\cdot(n-(q-p+1))! = \frac{2\cdot n!}{(q-p-1)\cdot(q-p)\cdot(q-p+1)}$ permutations, and by inserting $\frac{1}{n!}$ into the sum we get that $\beta^i_{p,q} = -\frac{2}{(q-p-1)\cdot(q-p)\cdot(q-p+1)}$.

□

We note that the prioritized ride-sharing problem is very similar to the setting of routing games [63]. The model of routing games is of one service provider that makes a round-trip along the locations of several sponsors in a fixed order, where the total cost of the trip should be distributed among the sponsors. Clearly, our problem is almost identical: the service provider corresponds to the vehicle and the sponsors correspond to the passengers. The only difference is that in a routing game the sponsors also pay the cost of the trip back to the origin. Indeed, the results presented in this section carry over to routing games.

Theorem 3.4.4. *The Shapley value in routing games can be computed in polynomial time.*

Proof (sketch). We use our previous definitions and results with the following slight modifications. The dummy destination d_{n+1} becomes d_0. Thus, $\delta(d_i, d_{n+1}) = \delta(d_i, d_0)$. In Observation 3.4.1 we need to modify the bound in the outer sum (with the index p) to n and the bound in the inner sum (with the index q) to $n+1$. In addition, we use the proof of Theorem 3.4.3, but we add $\sum_{p=0}^{i} \beta^i_{p,n+1}\delta(d_p, d_{n+1})$ to the calculation of $\phi(u_i)$, where for $p < i$, $\beta^i_{p,n+1} = -\frac{1}{(n-p)\cdot(n-p+1)}$ and $\beta^i_{i,n+1} = \frac{1}{n-i+1}$. □

Note that this is an unexpected result, since it refutes the conjecture in [83] that there is no efficient way for computing the Shapley value in routing games.

3.5 Non-prioritized Ride-sharing Problem

Similar to the prioritized ride-sharing problem we are given an initial priority order, which determines the drop-off order of the passengers. However, in the non-prioritized variant we do not enforce the fixed order for every subset of passengers. Instead, given a strict subset of passengers S, the cost associated with it, $c(S)$, is the length of the shortest path that traverses all of the destinations of the passengers in S.

3.5.1 The Hardness of the Non-prioritized Ride-sharing Problem

In Section 3.4 we showed that we can efficiently compute the Shapley value for the prioritized ride-sharing problem. In essence, the computation could be done efficiently since most of the travel distances cancel out, and only a polynomial number of terms remain in the computation. Unfortunately, this is not the case with the non-prioritized ride-sharing problem, where the Shapley value cannot be computed efficiently unless $P = NP$.

Clearly, finding the length of the shortest path (not necessarily a simple path) that starts at a specific node, v_0, and traverses all nodes in a graph (without returning to the origin) cannot be performed in polynomial time, unless $P = NP$. We denote this problem as *path-TSP*. We use the path-TSP to show that computing the Shapley value for the prioritized ride-sharing cannot be done efficiently, unless $P = NP$.

Theorem 3.5.1. *There is no polynomial time algorithm that computes the Shapley value for a given passenger in the prioritized ride-sharing problem unless $P = NP$.*

Proof. Given an instance of the path-TSP problem on a graph $G(V, E)$ we denote the solution by x. We construct an instance of the non-prioritized ride-sharing problem as follows. We build a graph $G'(V', E')$, where we add a node v', i.e., $V' = V \cup \{v'\}$. If $e \in E$ then $e \in E'$, and for all $v \in V$, $(v, v') \in E'$ with a weight of M, where M is the sum of weights of all the edges in E. Finally, we set $|U| = |V|$, $D = V' \setminus \{v_0\}$, $d_0 = v_0$, and the drop-off order is arbitrarily chosen. Recall that $c(D)$ is the total travel cost associated with the chosen drop-off order. We ask to compute the Shapley value of the user u' that is associated with the destination v'.

Clearly, the marginal contribution of u' to any strict subset of V is exactly M. However, the marginal contribution of u' to the complete set V is exactly $c(D)$ minus x (the length of the shortest path starting at v_0 and traversing all nodes in V). That is,

$$\phi(u') = \frac{(|U|-1)!}{|U|!}(c(D) - x) + \frac{|U|! - (|U|-1)!}{|U|!}M$$

After some simple mathematical manipulations we get that $x = |U|\phi(u') - (|U|-1)M + c(D)$. Therefore, if we can compute $\phi(u')$ in polynomial time then we can solve the path-TSP problem in polynomial time, which is not possible unless $P = NP$. □

3.5.2 Shapley Approximation based on a Prioritized Order

In Section 3.4 we presented a method for efficiently computing the Shapley value when a prioritization exists. In this section we show that our solution may be also applicable to the non-prioritized ride-sharing problem as an efficient proxy for the Shapley value. We term our proxy SHAPO: SHapley Approximation based on a Prioritized Order.

We compare SHAPO with the following three proxies for computing the Shapley value in traveling salesman games, that are in use in real-world applications [9].

Depot Distance This method divides the total ride cost proportionally to the distance from the depot, i.e. $Depot(u_i) = \frac{\delta(d_0,d_i)}{\sum_{j=1}^{n}\delta(d_0,d_j)}c(D)$. For example, a passenger traveling to a destination that is twice as distant from the origin as another passenger has to pay twice the cost, regardless of the actual travel path. We note that this method has outperformed all other methods in [9] on real data.

Shortcut Distance This method divides the total cost proportional to the change realized by skipping a destination when traversing the given path. Formally, let $Cut_i = \delta(d_{i-1}, d_i) + \delta(d_i, d_{i+1}) - \delta(d_{i-1}, d_{i+1})$. Then, $Shortcut(u_i) = \frac{Cut_i}{\sum_{j=1}^{n} Cut_j}c(D)$.

Re-routed Margin This method is a more sophisticated realization of the shortcut distance method. That is, instead of using the given path when skipping a destination, we compute the optimal path. Formally, $Reroute(u_i) = \frac{c(D)-c(D\setminus\{d_i\})}{\sum_{j=1}^{n} c(D)-c(D\setminus\{d_j\})}c(D)$. Note that when evaluating this proxy we need to solve n TSPs, one for leaving out each destination. This is the only proxy we consider that requires a non-negligible time to compute.

Experimental Settings

In order to evaluate the performance of SHAPO, we evaluated each of the methods for $3, 4, 5, 6, 7, 8$ and 9 passengers. For the road network we used the graph of the

city of Toulouse, France[1] as presented in Figure 2.1. This graph includes the actual distances between the different vertices. To convert the distances to travel costs we assumed a cost of $\$1$ per kilometer. The graph also includes the Toulouse-Blagnac airport, which was set as the origin (d_0). We cropped the graph to $40,000$ vertices, by running Dijkstra algorithm [23] starting at the airport, sorting all vertices by their distance from the airport, and removing all farther away vertices (including those that are unreachable). The destination vertices were randomly sampled for every passenger using a uniform distribution over all vertices, and each of the methods was evaluated 100 times against the true Shapley value of all passengers.

For running the simulations we assume that the given order of the passengers is according to the shortest path. This is a reasonable assumption, since if there is no prioritization, it is very likely that, in order to reduce the overall cost, the vehicle would travel using the shortest path (computed once). We conjecture that the results presented in this chapter will carry-out also to situations in which the given passenger order is very close to being optimal (but not necessarily the exact optimal order), but we leave it for future investigation.

Results

Figure 3.1 presents the running time, in seconds, required to compute the Shapley value and its proxies for all passengers on a single instance (in logarithmic scale). As expected, we can compute the proxies, except for the Re-routed margin proxy, almost instantaneously. However, due to the extensive time required to compute the Shapley value, and since we evaluate each method 100 times, we only evaluate the performance of all methods with up-to 9 passengers.

We evaluate the performance of SHAPO against the three other proxies using 5 different statistical measures (averaged on all 100 iterations). We use $X(u_i)$ to denote the estimated Shapley value by the evaluated proxy.

1. **Percent**: The average percentage of the deviation from the Shapley value. Formally, $Percent = \frac{1}{n}\sum_{i=1}^{n}\frac{|X(u_i)-\phi(u_i)|}{\phi(u_i)}$.
2. **MAE**: The mean absolute error, $MAE = \frac{1}{n}\sum_{i=1}^{n}|X(u_i)-\phi(u_i)|$.
3. **MSE**: The mean squared error, $MSE = \frac{1}{n}\sum_{i=1}^{n}(X(u_i)-\phi(u_i))^2$. This measure gives higher weight to larger deviations.
4. **RMSE**: The root mean squared error, $RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(X(u_i)-\phi(u_i))^2}$.
5. **Max-Error**: The maximum deviation among all passengers between the real and estimated Shapley value, $Max = \max_{i=1}^{n}(|X(u_i)-\phi(u_i)|)$.

The results are depicted in Tables 3.1, 3.2, 3.3, 3.4 and 3.5. SHAPO significantly outperforms the other proxies in all measures, with any number of passengers evaluated. Despite the depot distance method outperforming the other two methods, SHAPO is between 5.5 to 42.3 times better than the depot distance in all measures. Note that the units of MAE and Max-Error are dollars. That is, as depicted in Table 3.2, SHAPO deviated by only 19 cents, on average, from the actual Shapley value. The depot distance deviated by $\$1.33$, while the averaged shared-ride cost per passenger was approximately $\$5$. Similarly, the maximal deviation of SHAPO was less than 44 cents (on average), while the maximal deviation of the depot distance was more than $\$2.9$.

[1] obtained from https://www.geofabrik.de/data/shapefiles_toulouse.zip

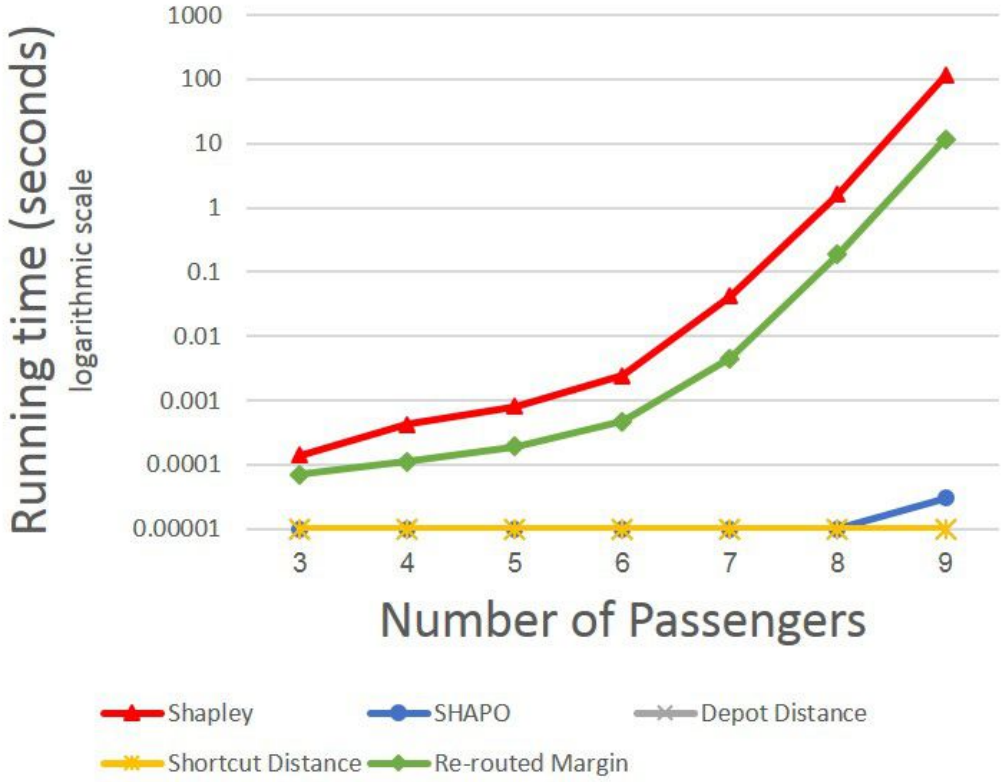

FIGURE 3.1: Running time, in seconds, required to compute a single instance of the Shapley value (in logarithmic-scale).

	3	4	5	6	7	8	9	AVG
SHAPO	**1.84%**	**3.44%**	**4.49%**	**4.68%**	**5.27%**	**6.53%**	**5.94%**	**4.60%**
Depot Distance	20.02%	26.36%	29.52%	35.28%	35.78%	35.92%	35.91%	31.26%
Shortcut Distance	34.79%	41.55%	45.44%	53.70%	54.23%	53.35%	55.89%	48.42%
Re-routed Margin	84.59%	76.13%	79.65%	119.90%	80.57%	85.97%	72.83%	85.66%

TABLE 3.1: Average percentage of the deviation from the Shapley value (Percent). Averaged over 100 iterations. Lower is better.

	3	4	5	6	7	8	9	AVG
SHAPO	**$0.11**	**$0.18**	**$0.20**	**$0.20**	**$0.20**	**$0.22**	**$0.21**	**$0.19**
Depot Distance	$1.11	$1.29	$1.34	$1.53	$1.38	$1.35	$1.28	$1.33
Shortcut Distance	$1.87	$2.19	$2.23	$2.47	$2.27	$2.07	$2.09	$2.17
Re-routed Margin	$3.21	$2.81	$2.45	$2.43	$2.10	$2.03	$1.85	$2.41

TABLE 3.2: The mean absolute error of the deviation from the Shapley value (MAE). Averaged over 100 iterations. Lower is better.

	3	4	5	6	7	8	9	AVG
SHAPO	**0.068**	**0.115**	**0.124**	**0.098**	**0.102**	**0.159**	**0.117**	**0.112**
Depot Distance	1.988	2.634	3.054	4.130	3.327	3.366	3.066	3.081
Shortcut Distance	5.805	8.431	8.937	10.840	9.749	8.097	7.731	8.513
Re-routed Margin	15.951	12.404	9.587	10.174	7.399	7.168	5.986	9.810

TABLE 3.3: The mean squared error of the deviation from the Shapley value (MSE). Averaged over 100 iterations. Lower is better.

	3	4	5	6	7	8	9	AVG
SHAPO	**0.121**	**0.204**	**0.239**	**0.246**	**0.259**	**0.299**	**0.277**	**0.235**
Depot Distance	1.205	1.474	1.594	1.859	1.699	1.703	1.617	1.593
Shortcut Distance	2.112	2.556	2.698	3.068	2.904	2.675	2.662	2.668
Re-routed Margin	3.540	3.200	2.866	2.951	2.517	2.508	2.292	2.839

TABLE 3.4: The root mean squared error of the deviation from the Shapley value (RMSE). Averaged over 100 iterations. Lower is better.

	3	4	5	6	7	8	9	AVG
SHAPO	**$0.17**	**$0.30**	**$0.39**	**$0.44**	**$0.49**	**$0.62**	**$0.58**	**$0.43**
Depot Distance	$1.66	$2.24	$2.73	$3.44	$3.32	$3.51	$3.47	$2.91
Shortcut Distance	$2.81	$3.92	$4.69	$5.66	$5.77	$5.49	$5.56	$4.84
Re-routed Margin	$4.81	$4.74	$4.60	$5.32	$4.67	$4.95	$4.71	$4.83

TABLE 3.5: The maximum deviation among all passengers between the real and estimated Shapley value (Max-Error). Averaged over 100 iterations. Lower is better.

3.6 Conclusions and Future Work

The Shapley value is considered one of the most important division scheme of revenues or costs, but its direct computation is often not practical for a reasonable size game. Therefore, Mann and Shapely [48] suggest to consider restrictions and constraints in order to find games where the Shapley value can be efficiently computed. The airport problem [45] is one example of these games, where the Shapley value can be efficiently computed. Our prioritized ride-sharing problem is a generalization of the airport problem to the 2D plane, and we showed that the Shapley can be efficiently computed in this generalization as well. However, we show that the non-prioritized ride-sharing problem, which is possibly the next level of generalization, cannot be efficiently computed (unless $P = NP$). Interestingly, the prioritized ride-sharing can still serve as an efficient proxy for the Shapley value of the non-prioritized ride-sharing problem where the provided travel path is the shortest path.

There are several interesting directions for future work. One possible direction is to compare our proxy for computing the Shapley value in the non-prioritized ride-sharing problem to a sampling based approach [16]. It is expected that a sampling based approach will be more accurate if there is a sufficient number of samples, but it will certainly require a lot more computation time. It is thus interesting to analyze when our proxy is still better than a sampling-based approach, and when it is the point in which a sampling-based approach becomes better than our proxy. From a theoretical perspective, we showed that computing the Shapley value for the non-prioritized ride-sharing problem is a hard problem. However, the hardness may be derived also from the hardness of path-TSP. There are several polynomial time approximation and heuristics for TSP that can be adjusted for path-TSP. It is thus interesting to analyze the computational complexity of finding the Shapley value, where $c(S)$ is computed using one of these approximations or heuristics.

Chapter 4

Setting the Drop-off Order using an Auction

4.1 Introduction

In many ride-sharing scenarios the different passengers may have different preferences regarding the possible drop-off order. For example, young people may not mind being dropped-off last if they save a few dollars, but people in their 40's may be more concerned about their time and thus would like to pay more to be dropped-off first. However, the usual approach of ride-sharing services is to determine the drop-off order according to a specific criterion (e.g., the order that minimizes the total travel time), disregarding the actual preferences of the passengers in a given ride.

In this chapter we thus present a mechanism for determining the drop-off order, which is based on the Vickrey–Clarke–Groves (VCG) mechanism [78, 19, 31]. Our mechanism allows every passenger, who was assigned to a specific vehicle, to provide her "value of time", and the mechanism then picks the drop-off order which maximizes the social value. The mechanism is truthful, that is, it is a dominant strategy for the passengers to provide their true "value of time". We run simulations to estimate the average commission charged by our mechanism, which resulted in only 8.5% of the ride cost. This value is significantly lower than the commission charged by ride-sourcing services such as Uber and Lyft which was reported to be at least 25% [62]. Clearly, our mechanism may charge an additional fee on the ride cost, without losing its beneficial properties.

4.2 Related Work

In this chapter we use an auction based mechanism for determining the drop-off order. Moulin and Shenker [52] also investigate an auction mechanism when sharing submodular costs. In their setting each agent may either receive a service or not, and the marginal cost of serving additional agents decreases as the group of agents who are already served increases. Moulin and Shenker investigate an auction mechanism to decide which agents are served, and then how to share the cost among them. In their work there is no drop-off order and each agent has a fixed value for being served. In our work the passengers are assumed to submit their value for each drop-off order, and are assumed to have a different value depending on the drop-off order. The service provider, on the other hand, only selects the drop-off order and must serve all the passengers.

Several works have considered the use of auctions in the domain of car-pooling. Kamar and Horvitz [39] use VCG-based payment in the car-pooling domain, which

depends on the assignments of passengers to vehicles and the chosen routes. We use the VCG-based payment in the domain of ride-sharing, and it depends only on the drop-off order of the passenger in a given vehicle. Kleiner, Nebel, and Ziparo [41] provide a solution for the assignment problem in the domain of car-pooling, based on auctions. Passengers are bidding for increasing their ranking, and thus visibility to drivers, whereas drivers can select passengers according to their preferences. Zhao et al. [84] analyze mechanisms for designating commuters to be either drivers or riders and calculating the payments. They show that the VCG mechanism results in a very high deficit, and they thus propose other mechanism with deficit control.

4.3 Mechanism Design for Determining the Drop-off order

In this chapter we assume that the group of passenger should decide on the drop-off order. Clearly, each order has a different total and personal ride cost, and the drop-off order also affects the travel time of each passenger (and possibly additional factors). Since each passenger may have a different value of time, we build a mechanism that elicits the preferences from the passengers and outputs a drop-off order as well as the mechanism fee for each passenger (in addition to the ride cost). We would like the mechanism to maximize the social welfare and to be strategy-proof. That is, the mechanism chooses an order that maximizes the sum of passengers' values, and truth-telling is a dominant strategy for each passenger.

We begin with some definitions. As before, let U be the set of passengers, and let R be a permutation of U. Passenger u_i has a true value v_i^R, for the shared-ride that drops-off all passengers according to the order in the permutation R. Given a set of passengers U and a drop-off order, R, a passenger u_i is associated with some ride cost c_i^R for this specific ride. This cost may depend on the entire set of destinations, as well as the order in which all passengers are dropped-off, and is known in advance to all passengers. Each passenger u_i reports her value for all possible drop-off orders; we use $\bar{v}_i^R$ to denote each of these reported values. The mechanism selects a drop-off order, $\hat{R}$, and determines the fee, $f_i^{\hat{R}}$, for passenger u_i. We note that, in addition to the fee paid by the passenger, the passenger must pay the ride cost, $c_i^{\hat{R}}$. That is, unlike the standard setting when VCG is applied, a user may have different values for the different options as well as different costs that are associated with each of the options. We use $g_i^{\hat{R}}$ to denote the utility (gain) of passenger u_i, $g_i^{\hat{R}} = v_i^{\hat{R}} - c_i^{\hat{R}} - f_i^{\hat{R}}$.

Before we present a truthful mechanism for this problem, we show that, in our case, the VCG mechanism that maximizes the reported values and ignores the pre-determined ride cost is not strategy-proof. Recall that the drop-off order ultimately selected by the VCG mechanism is $\hat{R} = \operatorname{argmax}_R \sum_i (\bar{v}_i^R)$, and the fee of VCG is $f_i^{\hat{R}} = \max_R \sum_{j \neq i} (\bar{v}_j^R) - \sum_{j \neq i} (\bar{v}_j^{\hat{R}})$. Consider the example described in Table **??**, where $U = u_1, u_2$.

If u_1 and u_2 bid truthfully (i.e. for all $R, \bar{v}_i^R = v_i^R$), the selected order is $u_1 \rightarrow u_2$ and the utilities are $g_1^{u_1 \rightarrow u_2} = 6 - 4 - f_1^{u_1 \rightarrow u_2} = 6 - 4 - (4 - 2) = 0$, and $g_2^{u_1 \rightarrow u_2} = 2 - 1 - f_1^{u_1 \rightarrow u_2} = 2 - 1 - (6 - 6) = 1$. Now, suppose that u_1 does not bid truthfully and $\bar{v}_1^{u_2 \rightarrow u_1} = 5$. The selected order in this case is $u_2 \rightarrow u_1$ and the utility of u_1 is $g_1^{u_2 \rightarrow u_1} = 3 - 2 - f_1^{u_2 \rightarrow u_1} = 3 - 2 - (4 - 4) = 1$. That is, truth-telling is not a dominant strategy for u_1 and VCG is not strategy-proof in this case.

Passengers \ Drop-off Order	$u_1 \to u_2$	$u_2 \to u_1$
	value (cost)	value (cost)
u_1	6 (4)	3 (2)
u_2	2 (1)	4 (4)

TABLE 4.1: An example in which the VCG mechanism that ignores the predetermined ride cost does not result in a truthful mechanism.

Our proposed VCG-based mechanism takes into account also the ride cost associated with every ride. It therefore selects the following drop-off order $\hat{R} = \underset{R}{\operatorname{argmax}} \sum_i (\bar{v}_i^R - c_i^R)$. In addition, given the selected order $\hat{R}$, we define $f_i^{\hat{R}} = \max_R \sum_{j \neq i} (\bar{v}_j^R - c_j^R) - \sum_{j \neq i} (\bar{v}_j^{\hat{R}} - c_j^{\hat{R}})$.

We now show that the proposed mechanism is truthful for every passenger u_i. Intuitively, given the selected order $\hat{R}$, $g_i^{\hat{R}} = v_i^{\hat{R}} - c_i^{\hat{R}} - f_i^{\hat{R}} = v_i^{\hat{R}} - c_i^{\hat{R}} - \left(\max_R \sum_{j \neq i} (\bar{v}_j^R - c_j^R) - \sum_{j \neq i} (\bar{v}_j^{\hat{R}} - c_j^{\hat{R}}) \right)$. Therefore, given the selected order, the utility of a passenger does not depend on her reported values.

Theorem 4.3.1. *For any passenger u_i, reporting the true values v_i^R is a dominant strategy.*

Proof. We need to show that for any reported values of the other passengers, the utility of u_i when she reports her true values, v_i^R, is greater than or equals her utility when she reports any other values. We fix $\bar{v}_j^R$ for all R and $j \neq i$. Let R^* be $\underset{R}{\operatorname{argmax}} \left(v_i^R - c_i^R + \sum_{j \neq i} (\bar{v}_j^R - c_j^R) \right)$. That is, the selected order when u_i bids truthfully. Let $\bar{R}$ be the selected order when u_i bids $\bar{v}_i^R$. We show that

$$g_i^{R^*} \geq g_i^{\bar{R}}$$

By definition,

$$g_i^{R^*} = v_i^{R^*} - c_i^{R^*} - \max_R \sum_{j \neq i} (\bar{v}_j^R - c_j^R) + \sum_{j \neq i} (\bar{v}_j^{R^*} - c_j^{R^*}).$$

In addition,

$$v_i^{R^*} - c_i^{R^*} + \sum_{j \neq i} (\bar{v}_j^{R^*} - c_j^{R^*}) = \max_R \left(v_i^R - c_i^R + \sum_{j \neq i} (\bar{v}_j^R - c_j^R) \right).$$

Therefore,

$$\begin{aligned}
g_i^{R^*} &= \max_R \left(v_i^R - c_i^R + \sum_{j \neq i} (\bar{v}_j^R - c_j^R) \right) - \max_R \sum_{j \neq i} (\bar{v}_j^R - c_j^R) \\
&\geq v_i^{\bar{R}} - c_i^{\bar{R}} + \sum_{j \neq i} (\bar{v}_j^{\bar{R}} - c_j^{\bar{R}}) - \max_R \sum_{j \neq i} (\bar{v}_j^R - c_j^R) \\
&= v_i^{\bar{R}} - c_i^{\bar{R}} - (\max_R \sum_{j \neq i} (\bar{v}_j^R - c_j^R) - \sum_{j \neq i} (\bar{v}_j^{\bar{R}} - c_j^{\bar{R}})) \\
&= g_i^{\bar{R}}.
\end{aligned}$$

□

A possible limitation of our approach is that each passenger needs to specify her value for each drop-off order. This can be a tedious task for a human to specify her exact value for each order. Furthermore, even specifying the value of a single drop-off order may be challenging for a human passenger, as this value requires an estimation of the monetary value of arriving at the destination. Instead, our mechanism can request a passenger to bid only on her value of time. That is, the amount (per minute) that a passenger would be willing to pay in order to save travel time. The mechanism can then automatically compute the values for each drop-off order based on the value of time, since the value of arriving to the destination is the same for each order. This results in a desirable property: a mechanism that encourages passengers to report their true value of time, and chooses the route accordingly.

4.3.1 Simulations

We now turn to evaluate the performance of our mechanism in a simulated environment. Our mechanism requires the definition of the ride cost (for each passenger and every drop-off order), as well as the valuation of each ride. Since our mechanism is truthful, we assume that $\bar{v}_i^R = v_i^R$. For every passenger, u_i, and drop-off order, R, we set the ride cost, c_i^R, to the Shapley value of that ride $\phi(u_i)$ (according to Section 3.4). Let t_i^R be the travel time from d_0 to d_i by the order R. Let c_i^p be the cost of a private and direct ride from d_0 to d_i, and t_i^p be the time of this ride. Each passenger is assumed to have a "value or time" Vot_i. In order to evaluate the value of a shared ride, we assume that the passenger's utility from a private ride equals 0. That is, the value of a passenger arriving at her destination, v_i^{dest}, is given by her "value of time" multiplied by the travel time of a private ride, added to the cost of a private ride, i.e., $v_i^{dest} = t_i^p \cdot Vot_i + c_i^p$. Therefore, given a passenger u_i, and a drop-off order R, the value of the shared ride $v_i^R = v_i^{dest} - t_i^R \cdot Vot_i$.

We use the graph of the city of Toulouse, France[1]. This graph includes the actual distances between the different vertices, and it also includes the Toulouse-Blagnac airport. We cropped the graph to $40,000$ vertices, by running Dijkstra algorithm [23] starting at the airport, sorting all vertices by their distance from the airport, and removing all farther away vertices (including those that are unreachable).

Being a last mile problem, we set the origin vertex (d_0) to be the same for all passengers, the Toulouse-Blagnac airport. To convert the distances to travel times we set the average speed to 30 kph. We also set the cost per minute of travel to $\$1$. The capacity of each vehicle was set to 4 passengers. We repeat the following process 1000 times. We sample 12 destination vertices using a uniform distribution over all vertices. We assign the passengers associated with the destinations to vehicles such that the total cost of the rides would be minimized. From this assignment we choose the vehicles which have 4 passengers assigned to them, but disregard the drop-off order determined by the assignment algorithm. The value of time was randomly sampled from an average income per minute, computed using US data of income and hours of work for each decile[2]. This process resulted in a total of 7424 passengers assigned to 1856 vehicles.

The main factor that we wanted to evaluate is the overhead of our mechanism. That is, we would like to ensure that the fees from the mechanism are not too high so

[1]The graph of Toulouse was obtained from https://www.geofabrik.de/data/shapefiles_toulouse.zip.

[2]Data was obtained from: https://dqydj.com/income-percentile-calculator.

that the total cost of the shared-ride would become inexpedient for the passengers. Indeed, the average fee in our simulations was only 8.48% from the overall cost ($c_i^{\hat{R}} + f_i^{\hat{R}}$). This overhead is significantly less than the commission charged by ride-sourcing services such as Uber and Lyft, which was reported to be at least 25% [62]. In addition, the average utility, $g_i^{\hat{R}}$ (which considers the travel time, the cost and the service fee) was 4.92. Recall that we assume that the utility of a private ride is 0, which entails that our proposed ride-sharing mechanism seems quite beneficial for the passengers.

4.4 Conclusions and Future Work

We consider a VCG based mechanism for determining the drop-off order. The proposed mechanism obtains the value of time from each of the passengers and outputs a drop-off order. The mechanism is both efficient and truthful, and can easily be modified in order to take into account additional properties such as the travel distance, the number of additional passengers in the vehicle, and other properties that may affect the passengers' value from each drop-off order. Moreover, we provide simulations showing that the cost paid to the mechanism service is reasonable.

For future work, we would like to extend our mechanism so that it will also determine the assignment. That is, the mechanism should elicit the preferences of the passengers before the assignment, and assign the passengers to vehicles accordingly. A bigger challenge would be to develop such a mechanism that will determine both the assignment and the drop-off order, while still being truthful and efficient.

Bibliography

[1] Niels Agatz et al. "Dynamic ride-sharing: A simulation study in metro Atlanta". In: *Procedia-Social and Behavioral Sciences* 17 (2011), pp. 532–550.

[2] Niels Agatz et al. "Optimization for dynamic ride-sharing: A review". In: *European Journal of Operational Research* 223.2 (2012), pp. 295–303.

[3] Gerald Albaum. "The Likert scale revisited". In: *Market Research Society. Journal.* 39.2 (1997), pp. 1–21.

[4] Shani Alkoby and David Sarne. "The Benefit in Free Information Disclosure When Selling Information to People." In: *AAAI*. 2017, pp. 985–992.

[5] Javier Alonso-Mora et al. "On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment". In: *Proceedings of the National Academy of Sciences* 114.3 (2017), pp. 462–467.

[6] Dan Ariely, George Loewenstein, and Drazen Prelec. ""Coherent arbitrariness": Stable demand curves without stable preferences". In: *The Quarterly Journal of Economics* 118.1 (2003), pp. 73–106.

[7] Amos Azaria et al. "Giving Advice to People in Path Selection Problems". In: *AAMAS*. 2012, pp. 459–466.

[8] Amos Azaria et al. "Strategic Information Disclosure to People with Multiple Alternatives". In: *AAAI*. 2011, pp. 594–600.

[9] Haris Aziz et al. "A study of proxies for shapley allocations of transport costs". In: *Journal of Artificial Intelligence Research* 56 (2016), pp. 573–611.

[10] Nicola Besozzi et al. "The traveling salesman game for cost allocation: The case study of the bus service in castellanza". In: *Game theory* 2014 (2014).

[11] Filippo Bistaffa et al. "Recommending fair payments for large-scale social ridesharing". In: *Proceedings of the 9th ACM Conference on Recommender Systems*. 2015, pp. 139–146.

[12] C. F. Camerer. "Behavioral Game Theory. Experiments in Strategic Interaction". In: Princeton University Press, 2003. Chap. 2, pp. 43–118.

[13] Ann Melissa Campbell and Martin WP Savelsbergh. "Decision support for consumer direct grocery initiatives". In: *Transportation Science* 39.3 (2005), pp. 313–327.

[14] Harry Campbell. *RSG 2017 Survey Results: Driver Earnings, Satisfaction and Demographics*. 2017.

[15] Dale Carnegie. *How to win friends and influence people*. Simon and Schuster, 1936.

[16] Javier Castro, Daniel Gómez, and Juan Tejada. "Polynomial calculation of the Shapley value based on sampling". In: *Computers & Operations Research* 36.5 (2009), pp. 1726–1730.

[17] Shih-Fen Cheng, Duc Thien Nguyen, and Hoong Chuin Lau. "Mechanisms for arranging ride sharing and fare splitting for last-mile travel demands". In: *AAMAS*. 2014, pp. 1505–1506.

[18] Blerim Cici et al. "Quantifying the potential of ride-sharing using call description records". In: *Proceedings of the 14th Workshop on Mobile Computing Systems and Applications*. 2013, p. 17.

[19] Edward H Clarke. "Multipart pricing of public goods". In: *Public choice* 11.1 (1971), pp. 17–33.

[20] Jean-François Cordeau and Gilbert Laporte. "A tabu search heuristic for the static multi-vehicle dial-a-ride problem". In: *Transportation Research Part B: Methodological* 37.6 (2003), pp. 579–594.

[21] G. B. Dantzig and J. H. Ramser. "The Truck Dispatching Problem". In: *Management Science* 6.1 (1959), pp. 80–91.

[22] Marco Diana, Maged M Dessouky, and Nan Xia. "A model for the fleet sizing of demand responsive transportation services with time windows". In: *Transportation Research Part B: Methodological* 40.8 (2006), pp. 651–666.

[23] E. W. Dijkstra. "A note on two problems in connexion with graphs". In: *Numerische Mathematik* 1.1 (1959), pp. 269–271.

[24] Nasser A El-Sherbeny. "Vehicle routing with time windows: An overview of exact, heuristic and metaheuristic methods". In: *Journal of King Saud University-Science* 22.3 (2010), pp. 123–131.

[25] M. G. Fiestras-Janeiro et al. "Cost allocation in inventory transportation systems". In: *TOP* 20.2 (2012), pp. 397–410.

[26] PC Fishburn and HO Pollak. "Fixed-route cost allocation". In: *The American Mathematical Monthly* 90.6 (1983), pp. 366–378.

[27] Robert W. Floyd. "Algorithm 97: Shortest Path". In: *Communications of the ACM* 5.6 (1962), p. 345.

[28] Mikael Frisk et al. "Cost allocation in collaborative forest transportation". In: *European Journal of Operational Research* 205.2 (2010), pp. 448–458.

[29] Liping Fu and Stan Teply. "On-Line and Off-Line Routing and Scheduling of Dial-a-Ride Paratransit Vehicles". In: *Computer-Aided Civil and Infrastructure Engineering* 14.5 (1999), pp. 309–319.

[30] Ya'akov Gal and Avi Pfeffer. "Modeling reciprocal behavior in human bilateral negotiation". In: *AAAI*. 2007, pp. 815–820.

[31] Theodore Groves. "Incentives in teams". In: *Econometrica* 41.4 (1973), pp. 617–631.

[32] Mario Guajardo and Mikael Rönnqvist. "A review on cost allocation methods in collaborative transportation". In: *International transactions in operational research* 23.3 (2016), pp. 371–392.

[33] Erick Guerra. "Planning for Cars That Drive Themselves: Metropolitan Planning Organizations, Regional Transportation Plans, and Autonomous Vehicles". In: *Journal of Planning Education and Research* 36.2 (2016), pp. 210–224.

[34] Chen Hajaj, Noam Hazon, and David Sarne. "Enhancing comparison shopping agents through ordering and gradual information disclosure". In: *Autonomous Agents and Multi-Agent Systems* 31.3 (2017), pp. 696–714.

[35] John Michael Harris, Jeffry L Hirst, and Michael J Mossinghoff. *Combinatorics and graph theory*. Vol. 2. Springer, 2008.

[36] Koen Hindriks and Dmytro Tykhonov. "Opponent modelling in automated multi-issue negotiation using bayesian learning". In: *AAMAS*. 2008, pp. 331–338.

[37] Jang-Jei Jaw et al. "A heuristic algorithm for the multi-vehicle advance request dial-a-ride problem with time windows". In: *Transportation Research Part B: Methodological* 20.3 (1986), pp. 243–257.

[38] David S Johnson and Lyle A McGeoch. "The traveling salesman problem: A case study in local optimization". In: *Local search in combinatorial optimization* 1.1 (1997), pp. 215–310.

[39] Ece Kamar and Eric Horvitz. "Collaboration and shared plans in the open world: Studies of ridesharing". In: *Proceedings of the 21st International Joint Conference on Artificial Intelligence*. 2009, pp. 187–194.

[40] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[41] Alexander Kleiner, Bernhard Nebel, and Vittorio Amos Ziparo. "A mechanism for dynamic ride sharing based on parallel auctions". In: *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*. 2011, pp. 266–272.

[42] Jason Koebler. *Why Everyone Hates UberPOOL?* 2016.

[43] Fabien Lehuédé et al. "A multi-criteria large neighbourhood search for the transportation of disabled people". In: *Journal of the Operational Research Society* 65.7 (2014), pp. 983–1000.

[44] Yeqian Lin et al. "Research on optimization of vehicle routing problem for ride-sharing taxi". In: *Procedia-Social and Behavioral Sciences* 43 (2012), pp. 494–502.

[45] Stephen C Littlechild and Guillermo Owen. "A simple expression for the Shapley value in a special case". In: *Management Science* 20.3 (1973), pp. 370–372.

[46] George Loewenstein. "Willpower: A Decision-theorist's Perspective". In: *Law and Philosophy* 19 (1 2000), pp. 51–76. ISSN: 0167-5249.

[47] Yan Lyu et al. "R-Sharing: Rendezvous for Personalized Taxi Sharing". In: *IEEE Access* 6 (2017), pp. 5023–5036.

[48] Irwin Mann and Lloyd S Shapley. *Values of large games. 6: Evaluating the electoral college exactly*. Tech. rep. RAND CORP SANTA MONICA CA, 1962.

[49] Yves Molenbruch, Kris Braekers, and An Caris. "Typology and literature review for dial-a-ride problems". In: *Annals of Operations Research* (2017).

[50] Yves Molenbruch et al. "Multi-directional local search for a bi-objective dial-a-ride problem in patient transportation". In: *Computers & Operations Research* 77 (2017), pp. 58–71.

[51] Mojtaba Montazery and Nic Wilson. "Learning User Preferences in Matching for Ridesharing." In: *ICAART (2)*. 2016, pp. 63–73.

[52] Hervé Moulin and Scott Shenker. "Strategyproof sharing of submodular costs: budget balance versus efficiency". In: *Economic Theory* 18.3 (2001), pp. 511–533.

[53] John J Nay and Yevgeniy Vorobeychik. "Predicting human cooperation". In: *PloS one* 11.5 (2016), e0155656.

[54] Okan Örsan Özener. "Developing a collaborative planning framework for sustainable transportation". In: *Mathematical Problems in Engineering* (2014).

[55] Okan Örsan Özener and Özlem Ergun. "Allocating costs in a collaborative transportation procurement network". In: *Transportation Science* 42.2 (2008), pp. 146–165.

[56] Julie Paquette, Jean-François Cordeau, and Gilbert Laporte. "Quality of service in dial-a-ride operations". In: *Computers & Industrial Engineering* 56.4 (2009), pp. 1721–1734.

[57] Sophie N. Parragh, Karl F. Doerner, and Richard F. Hartl. "A survey on pickup and delivery problems. Part I: Transportation between customers and depot". In: *Journal für Betriebswirtschaft* 58.1 (2008), pp. 21–51.

[58] Sophie N. Parragh, Karl F. Doerner, and Richard F. Hartl. "A survey on pickup and delivery problems. Part II: Transportation between pickup and delivery locations". In: *Journal für Betriebswirtschaft* 58.1 (2008), pp. 81–117.

[59] Sophie N Parragh, Jorge Pinho de Sousa, and Bernardo Almada-Lobo. "The dial-a-ride problem with split requests and profits". In: *Transportation Science* 49.2 (2014), pp. 311–334.

[60] Sophie N Parragh et al. "A heuristic two-phase solution approach for the multi-objective dial-a-ride problem". In: *Networks* 54.4 (2009), pp. 227–242.

[61] Noam Peled, Ya'akov Kobi Gal, and Sarit Kraus. "A study of computational and human strategies in revelation games". In: *AAMAS*. 2011, pp. 345–352.

[62] Christian Perea. *What's The Real Commission That Uber Takes From Its Drivers?* https://therideshareguy.com/whats-the-real-commission -that-uber-takes-from-its-drivers-infographic/. 2016.

[63] Jos AM Potters, Imma J Curiel, and Stef H Tijs. "Traveling salesman games". In: *Mathematical Programming* 53.1-3 (1992), pp. 199–211.

[64] Lutz Prechelt. "Automatic early stopping using cross validation: quantifying the criteria". In: *Neural Networks* 11.4 (1998), pp. 761–767.

[65] Harilaos N Psaraftis. "A dynamic programming solution to the single vehicle many-to-many immediate request dial-a-ride problem". In: *Transportation Science* 14.2 (1980), pp. 130–154.

[66] Harilaos N Psaraftis, Min Wen, and Christos A Kontovas. "Dynamic vehicle routing problems: Three decades and counting". In: *Networks* 67.1 (2016), pp. 3–31.

[67] Avi Rosenfeld and Sarit Kraus. "Using aspiration adaptation theory to improve learning". In: *AAMAS*. 2011, pp. 423–430.

[68] Adella Santos et al. *Summary of travel trends: 2009 national household travel survey*. Tech. rep. U.S Departemnt of Transportation, Federal Highway Administration, 2011.

[69] Michael Schilde, Karl F Doerner, and Richard F Hartl. "Metaheuristics for the dynamic stochastic dial-a-ride problem with expected return transports". In: *Computers & operations research* 38.12 (2011), pp. 1719–1730.

[70] Nicola Secomandi. "Comparing neuro-dynamic programming algorithms for the vehicle routing problem with stochastic demands". In: *Computers & Operations Research* 27.11 (2000), pp. 1201–1225.

[71] Nicola Secomandi and François Margot. "Reoptation approaches for the vehicle-routing problem with stochastic demands". In: *Operations research* 57.1 (2009), pp. 214–230.

[72] Lloyd S Shapley. "A value for n-person games". In: *Contributions to the Theory of Games* 2.28 (1953), pp. 307–317.

[73] Bilong Shen, Yan Huang, and Ying Zhao. "Dynamic ridesharing". In: *SIGSPATIAL Special* 7.3 (2016), pp. 3–10.

[74] Harkiranpal Singh. "The importance of customer satisfaction in relation to customer loyalty and retention". In: *Academy of Marketing Science* 60.193-225 (2006), p. 46.

[75] Ventatramanan S Subrahmanian. *Heterogeneous agent systems*. MIT press, 2000.

[76] Lei Sun et al. "Transportation cost allocation on a fixed route". In: *Computers & Industrial Engineering* 83 (2015), pp. 61–73.

[77] Amos Tversky and Danliel Kahneman. "The Framing of Decisions and the Psychology of Choice". In: *Science* 211.4481 (1981), pp. 453–458.

[78] William Vickrey. "Counterspeculation, auctions, and competitive sealed tenders". In: *The Journal of finance* 16.1 (1961), pp. 8–37.

[79] Scott Wallsten. "The competitive effects of the sharing economy: how is Uber changing taxis?" In: *Technology Policy Institute* 22 (2015).

[80] Stephen Warshall. "A Theorem on Boolean Matrices". In: *Journal of the ACM* 9.1 (1962), pp. 11–12.

[81] Eyal Winter. "The shapley value". In: *Handbook of game theory with economic applications* 3 (2002), pp. 2025–2054.

[82] Jianjie Yang et al. "Multi-Objective Distribution Model and Algorithm for Online Shopping Express Logistics." In: *JCP* 8.10 (2013), pp. 2558–2564.

[83] Duygu Yengin. "Characterizing the Shapley value in fixed-route traveling salesman problems with appointments". In: *International Journal of Game Theory* 41.2 (2012), pp. 271–299.

[84] Dengji Zhao et al. "Incentives in ridesharing with deficit control". In: *Proceedings of the 13th International Conference on Autonomous Agents and Multi-agent systems*. 2014, pp. 1021–1028.