Shared Mobility – Operations and Economics

by

Amirmahdi Tafreshian

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Civil Engineering)
in the University of Michigan
2021

Doctoral Committee:

Assistant Professor Neda Masoud, Chair
Professor Henry Liu
Associate Professor Siqian Shen
Professor Yafeng Yin

“*I don’t have any particular recipe . . . Doing research is challenging as well as attractive. It is like being lost in a jungle and trying to use all the knowledge that you can gather to come up with some new tricks, and with some luck you might find a way out.*”

Stanford’s Maryam Mirzakhani Wins Field’s Medal – August 2014

Amirmahdi Tafreshian

atafresh@umich.edu

ORCID iD: 0000-0003-1175-0707

© Amirmahdi Tafreshian 2021

DEDICATION

To *maman-o-baba, dadash*, and *my love*!

ACKNOWLEDGMENTS

First and foremost, I would like to express my sincere gratitude to my PhD advisor, Dr. Neda Masoud, for her continued mentorship, support, and encouragement during these four years. I am highly indebted to her for guiding me in every step of this journey and going out of her way to help me build my future career. Aside from her academic support, she was a unique *source of calmness* during these four years, as I was more confident and determined about my research after every meeting or phone call with her. I truly admire her passion for research, her devotion to teaching, her exemplar personality, and above all, her genuine care for her students. Dr. Masoud, thank you for being such a great role model and the best mentor that I have always wished for; I am and will always be proud to be your first PhD student.

I am utterly grateful to my doctoral committee members Dr. Yin, Dr. Liu, and Dr. Shen for their helpful insight and guidance throughout my PhD studies. Particularly, I would like to thank Dr. Yin for all his support and encouragement from the first day I started this program.

The last four years at the University of Michigan were undoubtedly among the best years of my academic life. I must thank all the faculty and staff in the Civil and Environmental Engineering Department at the University of Michigan for providing students with a golden opportunity to prosper both academically and professionally. A special thanks goes to Dr. Lynch for his great leadership as well as his continued mentorship and support during my service at the Michigan Transportation Student Organization. I further owe a big thanks to all my previous mentors, advisors, and supervisors, including, but not limited to, Dr. Majid Jaridi, Dr. Stephen Valentine, Dr. Avinash Unnikrishnan, Dr. Feng Yang, and Dr. Mark Culp at West Virginia University, and Dr. Hashem Mahlooji, Dr. Kourosh Eshghi, Dr. Hessam Niaki, and Dr. Farhad Ghassemi Tari at Sharif University of Technology.

I am deeply indebted to my dear friends, Dr. Mehrdad Shahabi and Dr. Mahdiar Khakinejad, for all their help and guidance before and during my PhD studies. I am forever thankful to them for fueling my passion and giving me invaluable pieces of advice in every step of my academic endeavors. Also, I would like to thank all my friends at the University of Michigan, including, but not limited to, Ethan Zhang, Yiyang Wang, Zhengtian Xu, Daniel Vignon, Xiaotong Sun, Omid Bahrami, Sina Bahrami, Yiheng Feng, Zheng Yang, Yan Zhao, Alex Sundt, Qi Lou, Ali Shirazi, Edmond Huang, Xiangguo Liu, Yuexi Tu, and Roger Lloret: your friendship made these four years so memorable and sweet.

I greatly appreciate the collaboration with Crystal Wang, Mohammad Abouali, and Mojtaba

Abdolmaleki and their partial contributions to the work presented in Chapter 3.

A special thanks goes to my parents, Kazem and Zhoreh, and my brother, Amirhossein, the one and only *real* doctor in our family! It has been many years since I saw you in person, but you have been always in my heart and on my mind. Thank you for teaching me it is always okay to fall, but how you rise determines how far you go. It would be impossible to stand where I stand, without your unconditional love and endless support.

Last but not least, I would like to thank my dearest better half, Sahar, for her incredible help and endless patience throughout this journey. She is not only my love, but my best friend too. None of this would have been possible without her.

The work in this thesis was financially supported by the gracious research grants from the U.S. Department of Transportation Center for Connected and Automated Transportation (CCAT), the Ford Motor Company, the Michigan Institute for Data Science (MIDAS), the National Science Foundation (NSF awards 1831347 and 2046372), and the Rackham Predoctoral Fellowship.

TABLE OF CONTENTS

Chapter

LIST OF FIGURES

Figure

LIST OF TABLES

Table

LIST OF APPENDICES

Appendix

ABSTRACT

In the last decade, ubiquity of the internet and proliferation of smart personal devices have given rise to businesses that are built on the foundation of the sharing economy. The mobility market has implemented the sharing economy model in many forms, including but not limited to, carsharing, ride-sourcing, carpooling, taxi-sharing, ridesharing, bikesharing, and scooter sharing. Among these shared-use mobility services, ridesharing services, such as peer-to-peer (P2P) ridesharing and ride-pooling systems, are based on sharing both the vehicle and the ride between users, offering several individual and societal benefits. Despite these benefits, there are a number of operational and economic challenges that hinder the adoption of various forms of ridesharing services in practice. This dissertation attempts to address these challenges by investigating these systems from two different, but related, perspectives.

The successful operation of ridesharing services in practice requires solving large-scale ride-matching problems in short periods of time. However, the high computational complexity and inherent supply and demand uncertainty present in these problems immensely undermines their real-time application. In the first part of this dissertation, we develop techniques that provide high-quality, although not necessarily optimal, system-level solutions that can be applied in real time. More precisely, we propose a distributed optimization technique based on graph partitioning to facilitate the implementation of dynamic P2P ridesharing systems in densely populated metropolitan areas. Additionally, we combine the proposed partitioning algorithm with a new local search algorithm to design a proactive framework that exploits historical demand data to optimize dynamic dispatching of a fleet of vehicles that serve on-demand ride requests. The main purpose of these methods is to maximize the social welfare of the corresponding ridesharing services.

Despite the necessity of developing real-time algorithmic tools for operation of ridesharing services, solely maximizing the system-level social welfare cannot result in increasing the penetration of shared mobility services. This fact motivated the second stream of research in this dissertation, which revolves around proposing models that take economic aspects of ridesharing systems into account. To this end, the second part of this dissertation studies the impact of subsidy allocation on achieving and maintaining a critical mass of users in P2P ridesharing systems under different assumptions. First, we consider a community-based ridesharing system with ride-back guarantee, and propose a traveler incentive program that allocates subsidies to a carefully selected set of commuters to change their travel behavior, and thereby, increase the likelihood of finding more compatible and profitable matches. We further introduce an approximate algorithm to solve large-scale instances of this problem efficiently. In a subsequent study for a cooperative ridesharing market with role flexibility, we show that there may be no stable outcome (a collusion-free pricing and allocation scheme). Hence, we introduced a mathematical formulation that yields a stable outcome by allocating the minimum amount of external subsidy. Finally, we propose a truthful subsidy scheme to determine matching, scheduling, and subsidy allocation in a P2P ridesharing market with incomplete information and a budget constraint on payment deficit. The proposed mechanism is shown to guarantee important economic properties such as dominant-strategy incentive compatibility, individual rationality, budget-balance, and computational efficiency.

Although the majority of the work in this dissertation focuses on ridesharing services, the presented methodologies can be easily generalized to tackle related issues in other types of shared-use mobility services.

CHAPTER 1

Introduction

1.1 Motivation

The sharing economy is a business model that leverages peer-to-peer (P2P) transactions to generate value. The past decade has witnessed widespread access to the internet and proliferation of smartphones, leading to an unprecedented growth in businesses that are built on the foundation of the sharing economy. The mobility market has implemented the sharing economy model in many forms, including but not limited to, carsharing, ride-sourcing, carpooling, taxi-sharing, ridesharing, bikesharing, and scooter sharing (Shaheen et al., 2010; Shaheen and Cohen, 2007; Shaheen and Chan, 2016; Benjaafar et al., 2017; Benjaafar and Hu, 2020). Among these shared-use mobility services, carsharing, ride-sourcing, carpooling, taxi-sharing, and P2P ridesharing are based on sharing a vehicle. These services introduce a number of benefits, of which the most notable is enhancing mobility due to reducing the cost of transportation. However, they may leave different footprints on the transportation system and the environment.

Carsharing is a shared-use mobility service that allows its registered users to rent a car from a nearby location for a short period of time (usually a few hours) (Shaheen et al., 2018). It differs from the conventional car rental services in that the entire process of reservation, pick-up, and drop-off is self-served, and pick-up/drop-off locations are more widely spread across the network, thereby making the service more accessible to customers. Carsharing services typically operate under two different business models: one-way (or free-floating) carsharing (Weikl and Bogenberger, 2013; Correia et al., 2014; Boyacı et al., 2015; He et al., 2017), and two-way (or round-trip) carsharing (Nourinejad and Roorda, 2015; Chang et al., 2017; Lu et al., 2018; Ströhle et al., 2019; Yu and Shen, 2020). In the former case, the pick-up and drop-off location of the vehicle may be different, while in the latter case, customers are required to return the vehicle to the same location from which it is picked up. Carsharing is fundamentally different from other shared-use mobility services that are centered around vehicle sharing, as its users take on the task of driving and do not share rides. Therefore, increasing vehicle occupancy and utilization are not at the center of the expected benefits from this form of service.

ride-sourcing is an emerging shared mobility service in which scheduled or on-demand trip requests are served by private vehicle owners (Zha et al., 2016; Clewlow and Mishra, 2017; Zha et al., 2018; Braverman et al., 2019; Xu et al., 2020). Drivers in ride-sourcing services serve as independent contractors with flexible, self-determined schedules. Vehicles in ride-sourcing systems, such as Uber and Lyft, may add empty miles to the transportation network due to cruising and deadheading. Cruising accounts for empty miles driven by a ride-sourcing vehicle when it roams the network, waiting for its next assignment, and deadheading captures the empty miles that a ride-sourcing vehicle has to travel to its next pick-up location. There are variations of ride-sourcing services that are specifically designed to increase vehicle occupancy by incorporating pooling. Such variations are referred to as ride-pooling, ride-splitting, or ridesharing, and may curb the empty miles of ride-sourcing vehicles to some extent; however, unless the pooling strategy is paired with an efficient matching algorithm and implemented widely, its positive impacts may remain limited (Jacob and Roet-Green, 2018).

Ridesharing is fundamentally different from other shared-use mobility systems. Carsharing and ride-sourcing services, and the overwhelming majority of their variants, focus on sharing a vehicle among users. Ridesharing takes sharing to the next level, where both the vehicle and the ride are shared between two or more users of the system. Carpooling is one of the first realizations of ridesharing that facilitates sharing rides for recurrent trips among groups of peers with the same origin and/or destination or other commonalities. In traditional carpooling, also referred to as work-to-home or home-to-work carpooling, typically the origin or the destination of all trips resides in a single location, and users with similar itineraries use a single vehicle to complete their trips (Baldacci et al., 2004; Xia et al., 2015; Chou et al., 2016). Traditional carpooling often requires long-term commitment from participants, and accommodates pre-arranged, recurrent trips. An informal version of carpooling has been practiced largely by companies to incentivize their employees to share their commutes. In order to share the common costs of transportation, users in each group often take turns serving their peers using their personal vehicles. Casual carpooling (or slugging) is another variant of this service that requires no commitment and usually occurs in an ad-hoc manner at meeting points along the drivers' routes (Kelly, 2007; Mote and Whitestone, 2011; Shaheen et al., 2016; Cui et al., 2019). These meeting points are usually determined to allow users to take the most advantage of gaining access to high-occupancy-vehicle (HOV) lanes. The inherent uncertainty in the availability of riders at meeting points and the typical unwillingness of drivers to take detours cast a shadow of doubt on the self-sustainability of such services in practice.

Taxi-sharing is another variant of ridesharing systems that allows for pooling rides. In such systems, a pre-determined number of taxis are dispatched from a single or multiple depots to satisfy on-demand requests (Ma et al., 2013; Santi et al., 2014; Hosni et al., 2014; Jung et al., 2016; Alonso-Mora et al., 2017). The decision making in such systems is composed of an assignment problem, where taxis are assigned to batches of riders with similar itineraries, and a re-balancing problem that routes idle taxis toward regions with a high rate of un-served riders. The main purpose of designing shared-taxi services is to alleviate a shortcoming commonly observed in traditional taxi systems, i.e., low utilization of available seats. However, similar to ride-sourcing services, this form of service may suffer from increasing vehicles miles traveled (VMT) due to deadheading.

A P2P ridesharing system is a shared-use mobility service that provides matching, routing, and scheduling for a group of peer travelers with compatible itineraries (Agatz et al., 2011). A P2P ridesharing system consists of an online platform in which participants register their trips ahead of time (a few hours to a few seconds prior to their trips), and a system operator that matches riders and drivers and devises their itineraries. The P2P ride-matching problem is a generalization of the dial-a-ride problem (DARP). DARP was originally designed to model para-transit systems (Madsen et al., 1995; Healy and Moll, 1995; Fu, 2002); however, it has evolved through several years to model the operation of ride-sourcing systems (Jaw et al., 1986; Cordeau, 2006; Wang and Yang, 2019). The P2P ride-matching problem has a fundamental difference from ride-sourcing and shared-taxi services, in that in a P2P ridesharing system drivers are also customers who are willing to share their rides while completing their own trips. As a result, they typically have tight time windows, and are not available during the entire time horizon. Also, P2P ridesharing differs from traditional or casual carpooling, as it does not require a long-term arrangement and recurrence.

Ridesharing services, in general, and P2P ridesharing, shared-taxi and ride-pooling systems, in particular, have been shown to offer several societal and individual benefits. First, they substantially reduce the total VMT in the transportation network, which may ameliorate the perennial traffic congestion, especially in large metropolitan areas or during the peak hours (Stiglic et al., 2016; Agatz et al., 2010). Secondly, they can potentially reduce the cost of transportation as the incurred costs are shared among the participants sharing a ride (Chan and Shaheen, 2012). Finally, in the case of P2P ridesharing, it could also meet the increasing demand for mobility without adding to, or even curbing, the environmental footprint of the transportation sector by utilizing the empty spaces in traveling vehicles, rather than recruiting drivers whose sole purpose is to transport passengers. Despite all these benefits, ridesharing services have not gained as much traction as ride-sourcing services provided by Uber and Lyft have by American travelers in the past two decades. As an instance, (Iqbal, 2021) reports that only 20% of Lyft rides were shared through Lyft Line, the ride-pool service of Lyft. Therefore, it is crucial to investigate the failure of ridesharing services, and devise new techniques that help these services become a viable and reliable mode of transportation. In the next section, we introduce a number of the main challenges faced by ridesharing services.

1.2 Challenge

The successful implementation of a ridesharing service in practice requires addressing the following operational and economic challenges:

- **Complex Real-Time Ride-Matching:** When operating over a geographically large region and/or during the traffic peak hours, riders/drivers may have several options for sharing their rides, and the platform has to consider all these options with respect to a global view of the system. As such, the core of a ridesharing system is solving a ride-matching problem to assign riders to drivers and determine a plausible scheduling and vehicle routing that optimizes the pre-determined objective of the system. However, the inherent uncertainty of customer (rider and driver) arrivals, the large size of the decision space, and the necessity of having short response times by the platform turn this problem into a highly complex one, which immensely undermine the real-time application of ridesharing services in practice. Hence, it is crucial to devise techniques that provide high-quality, although not necessarily optimal, system-level solutions that can be applied in real time.
- **Absence of a Critical Mass:** In early stages of implementation, ridesharing services, especially P2P ridesharing, fail to achieve a critical mass that is necessary for obtaining a satisfactory matching rate, and thereby, ensuring a high quality-of-service for customers. There are at least two factors that contribute to this failure. First, the customers of these services often provide a tight time window that causes scarcity of spatial and/or temporal compatibility between trips. Secondly, even in the case of finding compatible trips, sharing rides may not attribute to a profitable match between travelers. Taking these issues into consideration, it is imperative for the platform to design promotion/reward programs that change the travel behavior of customers and/or surges their participation into the system.
- **Selfish And Strategic Users:** In solving the ride-matching problems for ridesharing services, the operator often chooses to maximize either the social welfare of the entire system or the total revenue of the platform. However, maximizing these objectives cannot solely result in increasing or even maintaining the penetration rate of a shared mobility service in which selfish customers act strategically to maximizes their own benefit. More precisely, similar to every other public market, ridesharing services cannot succeed without ensuring a proper allocation of profit shares among their users. This is true for any form of the ridesharing system regardless of assuming a cooperative or non-cooperative behavior for the participants in the market. In the case of a cooperative market, users, which often have different preferences over their assigned partners, might be better off by deviating from the system-optimal matching and forming coalitions with other user(s) to increase their collective benefits. Additionally, in a non-cooperative environment with incomplete information, customers may find incentives

to manipulate the outcome of the system by revealing their private information untruthfully to maximize their own utility. Therefore, it is of upmost importance that these platforms conform their policies and actions with respect to selfish behavior of their customers.

1.3 Contribution and Dissertation Outline

This dissertation aims to bring ridesharing services one step closer to becoming an attainable and sustainable mode of transportation in the near future. To this end, this dissertation focuses on the operational and economic challenges faced by ridesharing services from the perspective of both the platform and its users, and introduces new ideas and techniques to resolve these challenges or alleviate their adverse impacts. In what follows, we provide a brief overview of the following chapters, which collectively reveal the contributions of this dissertation.

In Chapters 2 and 3, I develop customized algorithms that address the computational challenges that arise when implementing two different forms of shared mobility services, namely P2P ridesharing and ride-pooling systems, in large-scale dynamic transportation networks. More specifically, in Chapter 2, we develop a solution methodology based on graph partitioning that facilitates solving ride-matching problems. The proposed method decomposes the original problem into multiple sub-problems that can be solved in parallel, with the goal of reducing the overall computational complexity of the problem as well as providing high quality solutions. This decomposition algorithm is shown to have a number of desirable properties, including converging in a finite number of steps, improving solutions with iterations, and more importantly, being applicable to different forms of ride-matching problems in the literature, e.g., one-to-one, one-to-many, and many-to-many matching.

In Chapter 3, we study the problem of dispatching a fleet of shuttles to serve trip requests in a dynamic environment with stochastic demand, which can be mathematically formulated as a dial-a-ride problem (DARP). This chapter introduces a general framework to shift much of the computational burden of the optimization problems that need to be solved into an offline setting, thereby addressing the on-demand requests with fast and high quality solutions. This framework includes: *i*) a new local search algorithm for generating a pool of useful routes in the offline mode; *ii*) an efficient algorithm to find the best subset of customers served by a fixed route; and *iii*) an online method that exploits the information from historical data to deploy shuttles proactively and change their routes if necessary in real time.

In Chapter 4, we study community-based ridesharing, as an instance of P2P ridesharing with ride-back guarantee. Community-based ridesharing provides a platform for commuters to transport their fellow community members from home to work in the morning and from work to home in the evening. This chapter addresses the second challenge discussed in the previous section by introducing a traveler incentive program that allocates monetary incentives to *i*) subsidize a select set of ridesharing rides, and, *ii*) encourage a few, carefully selected set of travelers to change their travel behavior (i.e., departure or arrival time). We present the mathematical formulation of this problem and develop an approximate method to solve its large-scale instances. We further propose a budget-balanced version of the problem that can be solved very efficiently.

In the next two chapters, we consider two variants of a P2P ridesharing market, once as a cooperative game with complete information and once as a non-cooperative game with incomplete information. In each case, we devise a pricing scheme that prevents the users from manipulating the outcome of the corresponding platform. More precisely, Chapter 5 introduces a new market game for a P2P ridesharing system in which users are flexible to play the role of either a rider or a driver. For this market, we theoretically show that there might exist no stable outcome, i.e., a pricing and allocation scheme from which no coalition of users have incentives to deviate. As such, we further present a mathematical formulation that yields the stable outcome if it ever exists or finds a near-stable outcome otherwise, and propose a decomposition algorithm to solve large instances of this problem efficiently.

So far in Chapters 2-5, we assume that the personal information of users are common knowledge. In Chapter 6, we relax this assumption and consider a P2P ridesharing market in which users hold their personal information private. For this market, we propose a subsidy scheme that provides monetary incentives to drivers to extend their time windows while ensuring that the payment deficit does not violate a budget constraint. We further prove that this pricing scheme satisfies important economic properties such as truthfulness, voluntary participation, and computational efficiency.

It is worth noting that although the majority of the work in this dissertation revolves around ridesharing services, the presented methodologies can be adjusted with minor effort to tackle related issues in other types of shared-use mobility services. Also, in all the chapters above, we provide the results of numerical experiments using the taxi trips in New York City to showcase the scalability and various aspects of the proposed methods. Finally, Chapter 7 concludes this dissertation and further discusses possible directions for future work.

The main chapters of this dissertation, Chapters 2 through 6, have appeared in the following five papers, respectively:

- Tafreshian, A. and Masoud, N. (2020a). Trip-based graph partitioning in dynamic ridesharing. *Transportation Research Part C: Emerging Technologies*, 114:532–553
- Tafreshian, A., Abdolmaleki, M., Masoud, N., and Wang, H. (2021). Proactive shuttle dispatching in large-scale dynamic dial-a-ride systems. *Transportation Research Part B: Methodological*, page to be appeared
- Tafreshian, A. and Masoud, N. (N.D.a). A traveler incentive program for promoting community-based ridesharing. *Under second round of review*
- Tafreshian, A. and Masoud, N. (2020b). Using subsidies to stabilize peer-to-peer ridesharing markets with role assignment. *Transportation Research Part C: Emerging Technologies*, 120:102770
- Tafreshian, A. and Masoud, N. (N.D.b). A truthful subsidy scheme for a peer-to-peer ridesharingmarket with incomplete information. *To be submitted*

Also, the content of Chapters 1 and 7 have partially appeared in the following paper:

- Tafreshian, A., Masoud, N., and Yin, Y. (2020). Frontiers in service science: Ride matching for peer-to-peer ride sharing: A review and future directions. *Service Science*, 12(2-3):44–60

CHAPTER 2

Trip-based Graph Partitioning in Dynamic Ridesharing

2.1 Introduction

General growth in the world's population is projected to add 2 billion to the world's population by the year 2050 (UN Department of Economics and Social Affairs, 2019). The rise in transportation demand that could arise from this growth has started movements toward devising more sustainable forms of transportation. One solution, which has been attracting more attention recently, is to increase the utilization rate of current transportation supply by providing a platform for travelers to share their trips, thereby reducing the number of single-occupancy vehicles on the roadway network.

With the prospect of driver-less vehicles becoming a technologically viable option in the near future, the number of start-up companies emerging in the areas of ridesharing, ride-sourcing, and carsharing has seen a sudden spike. More interestingly, the travelers' positive response to ride-sourcing services (e.g., Uber, Lyft, Didi) and the effect of carsharing services on vehicle ownership (Martin and Shaheen, 2011; Martin et al., 2010; Masoud and Jayakrishnan, 2017a) have convinced a number of well-established automotive manufacturers to start a movement to re-brand themselves as mobility companies. These recent trends point to a slow but steady shift from private vehicle ownership to the mobility-as-a-service (MaaS) business model.

A dynamic ridesharing system is a shared-use mobility service that provides matching, routeing, and scheduling for a group of peer travelers with compatible itineraries on a short notice (Agatz et al., 2011). The operator of a ridesharing system matches riders with drivers based on the proximity of their trips by solving a ride-matching problem. Since each rider/driver may have several potential driver/rider matches, the ridesharing operator needs to make the matching decision based on a global view of the system. In such a view, the operator considers all riders and drivers that are active in the system during the period of time for which a matching solution should be obtained. When operating over a geographically large region and/or during the traffic peak hours, the matching problem can grow large and challenging to solve.

This chapter introduces a methodology to decompose the graph associated with the ride-matching problem into smaller sub-graphs whose corresponding matching problems are easier to solve, while providing near-optimal solutions. Toward this end, we develop a graph partitioning-based method based on a one-to-one ride-matching problem, in which each driver shares a trip with at most one rider. Without loss of generality, we assume that the objective of the matching problem is to maximize the savings in vehicle miles obtained from ridesharing. Using numerical experiments with various parameter settings, the effectiveness of our proposed methodology on different metrics is compared with other partitioning techniques in the literature. We further evaluate the performance of our methodology in two more complex systems, namely a one-to-many ridesharing system, in which each driver may provide rides to multiple riders, and a ridesharing system with flexible roles, in which users are willing to perform either as a rider or as a driver.

In the rest of the chapter, we first review the related work in graph partitioning and ridesharing literature in Section 2.2. In Section 2.3, we set out the foundation of our work by introducing the dynamic ridesharing setting and briefly describing how the bipartite graph on which the one-to-one matching problem is solved can be constructed based on travelers' trip information. Then we discuss the characteristics of a desirable partitioning, and formulate what we call the "ε-uniform graph partitioning" problem. Further in this section, we discuss the computational complexity of this problem and introduce a high-quality heuristic to efficiently solve this problem. We finalize this section by extending our methodology to one-to-many ride-matching problem and ride-matching with role flexibility. In Section 2.4, we conduct a numerical study using the New York City taxi data and compare the performance of our proposed algorithm with the optimal solution as well as a benchmark algorithm. Finally, Section 2.5 concludes our findings and provides some directions for future research.

2.2 Literature Review

In this section we present a review of the academic literature in two fields, namely graph partitioning and dynamic ridesharing, and clearly state our contributions.

2.2.1 Graph Partitioning

Graph partitioning is a well-known problem in the graph theory literature. A graph G can be simply represented by a set of vertices V and edges E. Graphs are the cores of various network optimization problems. In recent years, with the advent of parallel machines on the one hand and the capability of handling big data on the other, researchers are incessantly seeking new techniques to provide fast and high quality solutions for operations over large graphs. One promising, yet sometimes challenging, method is to perform graph partitioning prior to conducting the operations. In general, graph partitioning aims at dividing a graph into smaller components such that the computational burden can be evenly distributed on every component.

In the literature, there is a wide spectrum of approaches to partition graphs based on the objectives they serve and the constraints to which they are subjected. These approaches can be roughly divided into two categories, global and local. The global methods use the entire graph and provide a partitioning directly. Global approaches can be further categorized as either exact methods (Brunetta et al., 1997; Karisch et al., 2000) or heuristic algorithms (Simon, 1991; Donath and Hoffman, 1972; Fiedler, 1975). The local algorithms, however, start with an initial partition and iteratively improve it through local search or similar approaches (Kernighan and Lin, 1970; Fiduccia and Mattheyses, 1982). For more detail on various approaches of graph partitioning, we refer the interested reader to a comprehensive survey by Buluç et al. (2016).

Many systems in various fields, including the transportation field, can be modeled using graphs. As previously stated, graph partitioning methods seek to segment a graph into smaller ones, when the graph grows large. However, before implementing a graph partitioning method, one needs to first create the graph. In many fields, creating the graph itself could be a time-sensitive task with substantial computational cost, specifically for systems that operate in real-time (one may need to spend computational resources to assess the feasibility of each edge/node in a graph.) Hence, many real-world problems may benefit from methods that can circumvent this step, and partition the graph nodes/edges before assessing the feasibility of edges/nodes. This type of partitioning, however, is problem specific. In this chapter, we introduce a graph partitioning method for the ridesharing problem without constructing the graph first. Specifically, we group the to-be graph nodes into partitions, where a sub-graph for each partition can be efficiently obtained. The problem-specific research question is how to perform such partitioning so as to ensure that valuable edges are preserved. An edge is considered valuable if removing it deteriorates the objective of the problem defined on the graph considerably.

2.2.2 Dynamic Ridesharing

Road congestion in metropolitan areas on one side, and ubiquity of smartphones and other means of communication on the other side have caused dynamic ridesharing services to attract increasing levels of interest in the past decade (Agatz et al., 2012). In a dynamic ridesharing system, travelers enter the system over time and provide some information regarding the location and time of their trips in the hope of sharing their trips with their peers who have similar routes and time windows. In such a system, the arrangement between travelers is non-recurring and can be on-the-fly or pre-arranged (Masoud and Jayakrishnan, 2017b). This is different from more traditional carpooling services (Baldacci et al., 2004; Calvo et al., 2004; Tamannaei and Irandoost, 2019; Li et al., 2018) that require a long-term commitment between travelers, and do not provide real-time arrangements.

Additionally, dynamic ridesharing differs from on-demand ride-hailing and taxi-sharing services (Ma et al., 2014; Zha et al., 2016; Lokhandwala and Cai, 2018; Ramezani and Nourinejad, 2017)) as in the latter case drivers are employed by the companies, and also drivers may not have narrow, self-determined time windows or pre-determined destinations. More details on the differences between dynamic ridesharing and other shared mobility services can be found in the survey by Furuhata et al. (2013).

Study of the related work reveals that there are different considerations for implementing a dynamic ridesharing system. For instance, a group of studies (Agatz et al., 2011; Najmi et al., 2017; Lloret-Batlle et al., 2017b) merely focus on one-to-one matching, in which each rider can be matched with exactly one driver, and each driver can transport at most one rider. While other studies (Herbawi and Weber, 2011; Di Febbraro et al., 2013; Stiglic et al., 2015; Masoud and Jayakrishnan, 2017b; Regue et al., 2016; Masoud et al., 2017b; Nam et al., 2018; Masoud et al., 2019) relax this assumption by allowing for multiple stops for drivers and/or multiple transfers between drivers for riders. Another aspect of ridesharing corresponds to the role of participants (i.e., rider or driver) in the system. Although it is usually assumed that users specify their roles during the trip registration, some studies (Agatz et al., 2011; Masoud and Jayakrishnan, 2017c; Amey et al., 2011; Lloret-Batlle et al., 2017b) consider cases where the system optimally assigns the role of participants.

Due to realizing the ride-share requests over time, the researchers usually focus on using a rolling horizon approach. In this approach, the system operator re-solves a static ride-matching problem at certain intervals by considering only current active users in the system. However, Agatz et al. (2012) emphasizes the fact that the ride-matching problem may not be solved in a timely manner when one practices the dynamic ridesharing system over a large metropolitan area with thousands of ride-share requests in a few minutes. To ameliorate this issue, they propose a decomposition approach based on the origin and destination of users' trips. They point out that this results in having possibly many users with trips in different sub-regions, and hence, a poor quality matching. As an alternative, Shen et al. (2016a) propose a filter-and-refine framework that partitions the original problem based on a grid of road network and solves the matching for each sub-region separately. Similarly, Pelzer et al. (2015) introduce a graph partitioning based on the topology of road networks to reduce the solution space. They further adopt an agent-based method that matches a rider and a driver if the rider's destination lies on the driver's route. Nourinejad and Roorda (2016) introduce a decentralized optimization technique combined with a first-price Vickrey auction to match riders and drivers. In their proposed algorithm, each driver places a bid on a rider only if the origin or the destination of the rider is in the proximity of the driver's route. More recently, Najmi et al. (2017) suggest a k-means clustering that groups users in k regions using the origin and destination of their trips. In their algorithm, the users whose origin and destination lie in two different clusters will be considered in both clusters. After solving the matching in each cluster separately, they need to solve another matching involving those users that have been matched in more than one cluster. Hereafter, we refer to this algorithm as "point-based partitioning" as opposed to our proposed algorithm which we refer to as "trip-based partitioning". In Section 2.4, we compare the performance of the trip-based partitioning method with that of the point-based partitioning method as a benchmark. In their seminal work, Alonso-Mora et al. (2017) propose a multi-step procedure to assign batches of riders to a set of drivers. They present an efficient technique to find shareable rides, and optimally assign them to a driver by solving an intger linear program. Finally, Simonetto et al. (2019) propose a two-level federated optimization architecture for the dynamic ridesharing problem. At the top level of this architecture, they efficiently find vehicles whose current position is geographically close to the pick-up location of customers that results in narrowing down the pairwise options between vehicles and customers considerably. It is also worth noting that some studies (Chen et al., 2017; Dong et al., 2018) in recent years have been devoted to analyzing the ridesharing data to better understand and predict the travel patterns of participants. It goes without saying that ridesharing systems can largely benefit from these analyses when solving ride-matching problems in real time.

2.2.3 Our Contributions

Similar to the aforementioned techniques, we propose a heuristic graph partitioning method to partition the ride-matching problem into smaller sub-problems with the objective of (*i*), reducing the overall complexity of the problem, and (*ii*), providing high quality solutions. Unlike the other methods that use the spatial proximity of origins or destinations, our method uses a dissimilarity measure based on the shareability of trips. Also, our method ensures that the matching problems of different clusters have similar running times, a consideration that cannot be guaranteed by previous methods. This feature enables us to reduce the maximum solution time of sub-problems when solving them in parallel. Our proposed partitioning method clusters the node set of the graph based on an approximation of the density of the graph edge set obtained from raw trip data and without generating the edges. After partitioning is completed based on this approximation, edges will be generated selectively, reducing the computational complexity of the problem. Finally, although our method is developed based on a one-to-one ride-matching problem, we later investigate the performance of the proposed method on more complex matching scenarios.

In sum, the contributions of this chapter to the literature are three-fold; we propose a graph partitioning approach to decompose the ride-matching problem and formulate it as an integer programming model, introduce a clustering problem as a proxy for the graph partitioning problem, and devise a heuristic algorithm to solve the clustering problem efficiently. The proposed graph partitioning technique enables us to divide a large ride-matching problem into smaller sub-problems that can be solved in parallel, facilitating real-time implementation of dynamic ridesharing over large-scale transportation networks.

2.3 Solution Methodology

We start this section by introducing the ridesharing system and its settings.

2.3.1 The Dynamic Ridesharing System

Consider a dynamic ridesharing system for a metropolitan area that provides a platform for individuals to share their personal trips with their peers over time. Without loss of generality, we assume that the study region consists of m stations, in which participants start or finish their trips, denoted by $s_1, s_2, \dots, s_m$. (Note that to model door-to-door transportation, it is sufficient to consider a large m.) These stations are selected at locations with high levels of demand for ridesharing, such as high density residential/recreational areas, business districts, and central transit stations (Masoud and Jayakrishnan, 2017c). Moreover, the study time horizon is discretized into a set of short time intervals of length, say, one minute. Discretizing time and introducing stations enables us to use time-dependent travel time matrices in our study. As a result, let $\tau_t(s_i, s_j)$ denote a function that generates the shortest-path travel times between stations s_i and s_j realized at time t from these matrices.

In order to participate in the ridesharing system, every user needs to register their trip in the system ahead of their desired departure time. Let us characterize the trip of a participant p by its origin and destination stations, denoted by s_p^{O} and s_p^{D}, respectively, and the earliest departure time from the trip origin, denoted by t_p^{ED}. Further, let t_p^{A} represent the announcement time of this trip that can be a few minutes to a few hours prior to its earliest departure time. The latest arrival time of the participant at the trip's destination station, denoted by t_p^{LA}, can be computed as the sum of the earliest departure time, travel time on the shortest path from the origin station to the destination station, and a maximum detour time. The maximum detour time for participant p, denoted by t_p^{det}, can either be specified by the participant or decided by the system operator, and correlates inversely with level of service. Finally, in this study we start by assuming that every user registers their trip as either a rider trip, if they are seeking a ride-share service, or a driver trip if they are willing to give a ride to their peers. We later introduce a version of the algorithm that can be applied to the more general ridesharing problem that includes role flexibility.

In response to the inherent uncertainty embedded in dynamic ridesharing systems, we adopt the rolling horizon strategy suggested by Agatz et al. (2011). In this strategy, we assume that the system operator solves the ride-matching problem periodically at pre-specified, equally-spaced points in time referred to as “re-optimization times”. The time window between every two consecutive re-optimization times is called “re-optimization period”. At each re-optimization time, the ride-matching problem includes all announced trips that have not yet expired or finalized (see Figure 2.1).

An announced trip is considered expired if its latest departure time occurs before the end of the current re-optimization period. In Figure 2.1, for example, all trips whose announcement times are before time t and latest departure times are after time $t + 1$ will be considered in the ride-matching problem in the specified re-optimization period. Also, an announced trip is considered finalized if it gets matched in the ride-matching problem, and the implied latest departure time of the driver trip associated with this match occurs before the end of the next re-optimization period. Agatz et al. (2011) defines the implied latest departure time as the latest possible time that a driver can leave its origin station while serving their matched rider(s) on time. For example, suppose driver d is matched to rider r in the specified re-optimization period of Figure 2.1. These trips will be finalized if the implied latest departure time of driver d (computed as $\min\{t_d^{\mathsf{LA}} - \tau(s_r^{\mathsf{O}}, s_r^{\mathsf{D}}) - \tau(s_d^{\mathsf{O}}, s_r^{\mathsf{O}}) - \tau(s_r^{\mathsf{D}}, s_d^{\mathsf{D}}), t_r^{\mathsf{LA}} - \tau(s_r^{\mathsf{O}}, s_r^{\mathsf{D}}) - \tau(s_d^{\mathsf{O}}, s_r^{\mathsf{O}})\}$) is before $t + 2$.

We set the objective function of the ride-matching as maximizing the vehicle miles traveled savings (VMTS). As mentioned in the previous section, the ride-matching problem can have different forms based on the preference of riders and drivers. In the next subsection, we consider the simplest form of the ride-matching problem, namely one-to-one matching problem. Please note that since the dynamic ridesharing system in a rolling time horizon framework can be thought as a process of solving different instances of the same problem over time, we draw our attention to only one re-optimization period for the rest of this section.

2.3.2 Maximum Weighted Bipartite Matching

The one-to-one matching problem can be represented by a bipartite graph $G = (P, L)$, where P is the set of participants and L is the link set. Let us denote the set of riders by $R = \{r\}$ and the set of drivers by $D = \{d\}$. In this section, we consider the rider and driver sets to be mutually exclusive,

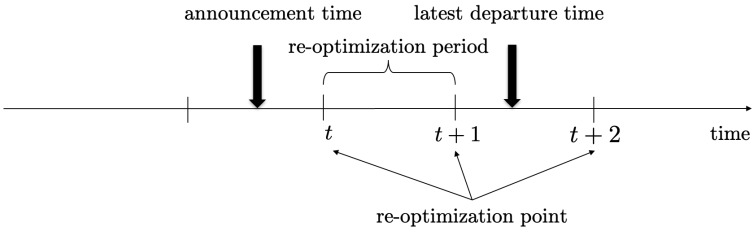


Figure 2.1: Rolling horizon framework

and collectively form the set of participants, $P = \{p\} = R \cup D$, which constitute the set of nodes in graph G.

A link $\ell = (r, d) \in L$ between rider r and driver d exists in graph G if the driver is capable of serving the rider before heading to his/her own destination. This condition can be mathematically expressed by having inequalities (2.1a) and (2.1b) hold concurrently, where t is the most recent realization time of travel time matrices. Inequality (2.1a) ensures that a driver who arrives at a rider's origin station as early as possible would have enough time to carry the rider to his/her destination. Inequality (2.1b) states that after dropping off the rider, the driver should have enough time to accomplish his/her own trip.

$$\max\{t_d^{\mathsf{ED}} + \tau_t(s_d^{\mathsf{O}}, s_r^{\mathsf{O}})\; t_r^{\mathsf{ED}}\} + \tau_t(s_r^{\mathsf{O}}, s_r^{\mathsf{D}}) \leq t_r^{\mathsf{LA}}\,, \tag{2.1a}$$

$$\max\{t_d^{\mathsf{ED}} + \tau_t(s_d^{\mathsf{O}}, s_r^{\mathsf{O}})\; t_r^{\mathsf{ED}}\} + \tau_t(s_r^{\mathsf{O}}, s_r^{\mathsf{D}}) + \tau_t(s_r^{\mathsf{D}}, s_d^{\mathsf{D}}) \leq t_d^{\mathsf{LA}}\,. \tag{2.1b}$$

For every link $(r, d) \in L$, let w_{rd} denote the saving in the vehicle miles traveled which can be calculated as:

$$w_{rd} = \max\{\phi(s_d^{\mathsf{O}}, s_d^{\mathsf{D}}) - \phi(s_d^{\mathsf{O}}, s_r^{\mathsf{O}}) - \phi(s_r^{\mathsf{D}}, s_d^{\mathsf{D}})\,, 0\}\,, \tag{2.2}$$

where $\phi(s_i, s_j)$ represents the shortest driving distance between stations s_i and s_j. We note that Equation (2.2) is based on the assumption that riders who are not served use their private vehicles. Equation (2.2) simply indicates that the saving equates the difference between the shortest driving time of the driver, and the incurred detours due to the rider pick-up and drop-off. As a result, the goal of the ride-matching problem is to find the solution that yields the highest total VMTS. This problem can be mathematically formulated as an integer programming model in (2.3). The decision variable u_{rd} is a binary variable that holds value 1 if rider r is matched with driver d, and value 0 otherwise. The objective function in Statement (2.3a) maximizes the total savings in vehicle miles travelled (VMT). Constraint (2.3b) ensures that each driver is assigned to at most one rider, and constraint (2.3c) ensures that each rider is served at most once. Note that although the decision variable u is a binary variable by definition, the total-unimodularity structure of the constraint set allows for relaxing the binary constraint and replacing it with constraint (2.3d).

$$\max \quad \sum_{r \in R} \sum_{\substack{d \in D: \\ (r,d) \in L}} w_{rd} u_{rd} \tag{2.3a}$$

$$\text{s.t.} \quad \sum_{\substack{r \in R: \\ (r,d) \in L}} u_{rd} \leq 1\,, \qquad \forall\, d \in D\,, \tag{2.3b}$$

$$\sum_{\substack{d \in D: \\ (r,d) \in L}} u_{rd} \leq 1\,, \qquad \forall\, r \in R\,, \tag{2.3c}$$

$$0 \leq u_{rd} \leq 1\,. \tag{2.3d}$$

The matching problem in (2.3) can be solved reasonably fast using commercial optimization engines or custom-designed algorithms (Edmonds and Karp, 1972; Fredman and Tarjan, 1987)) for small- to medium-sized problems. However, in ridesharing systems that operate over a large region and/or under high levels of demand, the large scale of the problem can prohibit it from being solved in real-time, or at all. In this chapter, we propose a graph partitioning method to divide the bipartite graph G into disjoint sub-graphs, creating smaller-size problems that are computationally less expensive to solve.

2.3.3 The Connectivity Matrix

Let us form the binary matrix F based on graph G, such that each element $(r, d) \in R \times D$ of this matrix holds value 1 if r and d are connected in the bipartite matching graph, and value 0 otherwise. We call matrix F the "connectivity matrix" of graph G, denoted by F_G (hereafter referred to as F for simplicity).

Decomposing the connectivity matrix F into sub-matrices corresponds to partitioning graph G into sub-graphs. Graph partitioning can be done all at once or sequentially. In the former case, we divide the original graph into K sub-graphs. In the latter case, we successively partition each sub-graphs into two sub-graphs at each level. In this case, $K = 2^{\text{number of levels}}$. Figure 2.2 displays an example of sequential partitioning. In this figure, the matrix has been divided into four sub-matrices at splitting points b_v and b_h. Level 1 partitioning as depicted in this figure will provide sub-matrices $F_i,\ i = \{1, 2, 3, 4\}$, while level 2 partitioning corresponds to decomposing level 1 sub-matrices, and will result in sub-matrices $F_{i,j},\ (i, j) \in \{1, 4\} \times \{1, 2, 3, 4\}$. In general, level i of partitioning is equivalent to dividing all partitions in level $i - 1$ into four partitions.

In Figure 2.2, the sub-graph corresponding to F_1 includes drivers in set D_1 and riders in set R_1. Similarly, the sub-graph of F_4 includes driver in D_2 and riders in R_2. Hence, since $R_1 \cup R_2 = R$, $R_1 \cap R_2 = \emptyset$, $D_1 \cup D_2 = D$, and $D_1 \cap D_2 = \emptyset$, the sub-graphs associated with F_1 and F_4 are exhaustive and disjoint in set P. Similarly, F_2 and F_3 correspond to exhaustive disjoint sub-graphs. Hence, in order to approximate the matching problem associated with G with two smaller matching problems, we solve the matching problems associated with either F_1 and F_4, or F_2 and F_3. Without loss of generality, we assume that F_1 and F_4 are always selected. We note that this procedure provides solutions to two sub-problems the union of which provides only an approximation of the solution to the original problem. The reason is that the unit elements of sub-matrices F_2 and F_3 correspond to links in G (i.e., potential matches) that have been removed in order to partition G into two disjoint sub-graphs.

2.3.4 Properties of a Desirable Partitioning

In the previous section, we noted that the splitting points b_v and b_h are required to decompose matrix F. However, there are two sets of decisions that affect the quality of the sub-graphs resulting from this partitioning, namely the location of the splitting points, and the row and column orderings in matrix F.

Choice of the splitting points, b_v and b_h, affects the quality of the solutions in two ways. Firstly, by moving the location of the splitting points, we are in fact changing the riders and drivers in the resultant sub-graphs. The proper selection of b_v and b_h would result in a partitioning that (*i*) removes as few links as possible, and (*ii*), results in a near uniform split of the graph. The reason for the former condition is that the number of links removed as a result of partitioning of G provides a bound on the performance of the partitioning procedure, since each removed link is a potential matching opportunity that is being ignored. The latter condition stems from the objective of partitioning, which is reducing the computational complexity of the problem. One can use the Fredman-Tarjan algorithm to solve the maximum weighted bipartite matching problem, which is known to have a running time complexity of $O(|P||L| + |P|^2 \log |P|)$. Hence, a partitioning of the graph that is uniform in terms of number of nodes and links would provide the highest computational benefit.

Secondly, for a given set of splitting points, the ordering of riders and drivers in F leads to different sub-graphs. It is easy to see that we can re-order the identification numbers of riders and drivers in F before decomposing it, without making any changes to the original problem. Figure 2.3 showcases an example of how re-ordering riders and drivers in F can lead to more desirable partitions. In Figure 2.3(a), a good split, subject to uniformity in the size of the sub-graphs, requires omitting 7 links. Re-ordering the rider and driver sets, as presented in Figure 2.3(b), could lead to a much more desirable partitioning that removes only 2 links. Note that choice of the splitting points

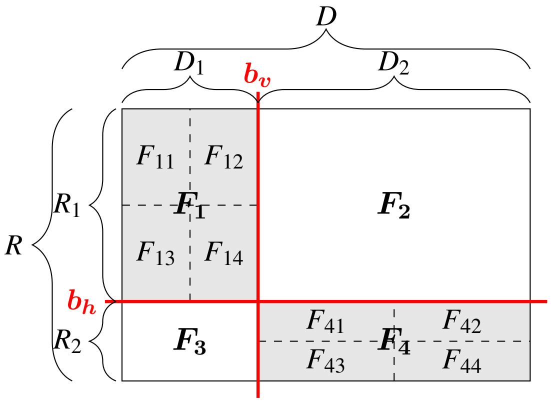
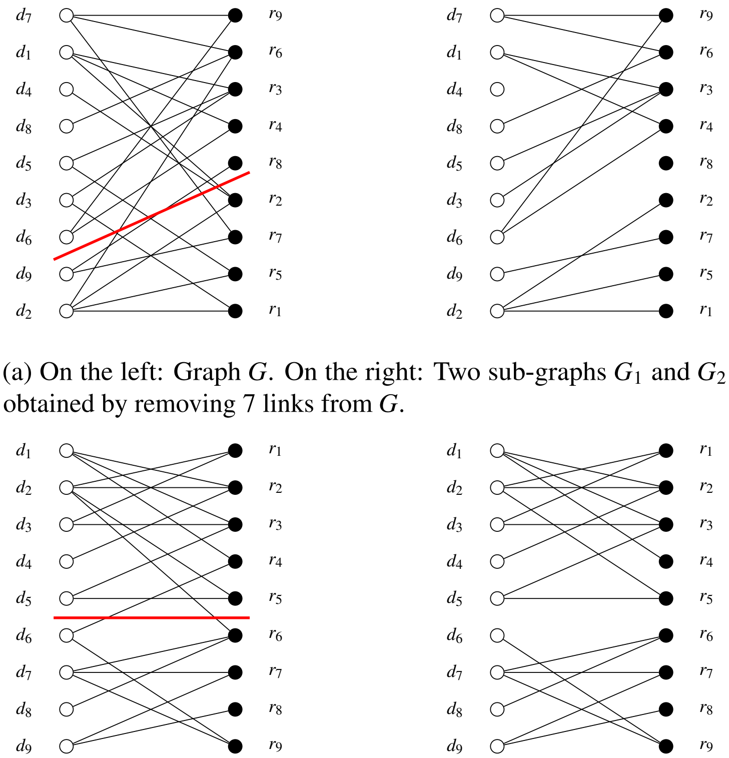


Figure 2.2: Partitioning the connectivity matrix F at splitting points b_v and b_h. Level 1 partitioning will provide sub-matrices $F_i,\ i = \{1, 2, 3, 4\}$, while level 2 partitioning will provide sub-matrices $F_{i,j},\ (i, j) \in \{1, 4\} \times \{1, 2, 3, 4\}$.

and row and column re-orderings should take place concurrently. In the next section, we present the mathematical formulation of a partitioning that satisfies both conditions presented in this section.



(a) On the left: Graph G. On the right: Two sub-graphs G_1 and G_2 obtained by removing 7 links from G.

(b) On the left: Graph G after row and column re-ordering. On the right: Two subgraphs G_1 and G_2 obtained by removing 2 links from G.

Figure 2.3: Bipartite matching graph: Two re-orderings of same set of participants

2.3.5 The ε-uniform Graph Partitioning Problem

The goal of the ε-uniform graph partitioning problem is to partition graph G into K disjoint components by eliminating links with the smallest weights, while guaranteeing that the difference between the number of nodes and links in partitions are within the thresholds of $\varepsilon_p|P|$ and $\varepsilon_\ell|L|$, respectively. Note that ε_p and ε_ℓ are uniformity factors ($\in [0, 1]$) whose values inversely affects the uniformity in the number of nodes and links between partitions. This problem is mathematically formulated in model (2.4). Let us define the binary decision variable f_{pk} to take value 1 if node $p \in P$ is assigned to sub-graph k, and value 0 otherwise. Furthermore, we define the binary variable $h_{\ell k}$ to take value 1 if link ℓ is assigned to sub-graph k, and value 0 otherwise. Also, let us denote the non-negative weight of each link by ω_ℓ.

The objective function in Statement (2.4a) maximizes the total sum of link weights within the partitions, hence minimizing the weighted sum of omitted links with end points in different partitions. Constraint (2.4b) ensures that each node is assigned to a single sub-graph. Constraint (2.4c) ensures that only links connecting two nodes within the same sub-graph are selected; if both end nodes of link ℓ, p and p', belong to sub-graph k, the left-hand side of the constraint becomes 1, and the max objective function forces $h_{\ell,k}$ to take its upper bound of 1. Any other arrangement of 0 and 1 values assigned to variables f_{pk} and $f_{p'k}$ would make the left hand side of inequality (2.4c) smaller than 1, forcing the binary variable $h_{\ell k}$ to take value 0.

Although Constraints (2.4b) and (2.4c) partition G into K sub-graphs, they do not impose any restrictions on the size of the resulting sub-graphs. Constraint (2.4d) ensures that links are, within a threshold $\varepsilon_\ell|L|$, distributed uniformly among sub-graphs. Setting ε_ℓ close to zero would distribute the links more uniformly among sub-graphs, reducing the computational complexity of the otherwise larger sub-problem. However, this reduced computational complexity may come at the cost of a higher number of eliminated links, and hence lower accuracy. Analogous to constraint (2.4d), constraint (2.4e) ensures a uniform distribution of nodes among sub-graphs, within the threshold $\varepsilon_p|P|$. Constraint (2.4f) enforces decision variables to take only binary values.

$$
\begin{aligned}
\max \quad & \sum_{\ell \in L} \omega_\ell \sum_{k=1}^{K} h_{\ell k} && && \text{(2.4a)}\\
\text{s.t.} \quad & \sum_{k=1}^{K} f_{pk} = 1\,, && \forall p \in P\,, && \text{(2.4b)}\\
& \frac{f_{pk} + f_{p'k}}{2} \geq h_{\ell k}\,, && \forall k \in \{1,\dots,K\}, \ell = (p,p') \in L\,, && \text{(2.4c)}\\
& \sum_{\ell \in L} h_{\ell k} \leq (1+\varepsilon_\ell)\frac{|L|}{K}\,, && \forall k \in \{1,\dots,K\}\,, && \text{(2.4d)}\\
& \sum_{p \in P} f_{pk} \leq (1+\varepsilon_p)\frac{|P|}{K}\,, && \forall k \in \{1,\dots,K\}\,, && \text{(2.4e)}\\
& f, h \in \{0,1\}\,. && && \text{(2.4f)}
\end{aligned}
$$

We note that problem (2.4) formulates the partitioning into K sub-graphs all at once. In order to use this formulation in a successive bisection method, at each level, we let $K = 2$, and set P and L as the sets of nodes and links from the graph in the previous level, respectively. Also, we substitute $(1 + \varepsilon_p)$ and $(1 + \varepsilon_\ell)$ with $(1 + \varepsilon_p)^{\frac{1}{\text{number of levels}}}$ and $(1 + \varepsilon_\ell)^{\frac{1}{\text{number of levels}}}$. It has been shown that the mathematical problem in model (2.4) is NP-hard (Hyafil and Rivest, 1973). Additionally, creating graph G, which is an input to this problem, can create a computational burden in real-time applications, since it requires $|R| \times |D|$ trip comparisons to generate the set L, each trip comparison requiring computations in inequalities (2.1a) and (2.1b). In the next section, we present an algorithm to approximate this problem fast and with a high level of accuracy, without the necessity of generating graph G a priori.

2.3.6 The Trip-based ε-uniform Partitioning Algorithm

As an alternative for the problem in (2.4), we introduce a "proxy" problem in which generating the entire graph G is not required a priori; rather, we strategically decide the parts of graph G that need to be generated. In this proxy problem, we assume that every user's trip is an object that has a dissimilarity with any other object. This helps us formulate the problem as a clustering problem that aims at assigning $N = |P|$ objects to K clusters such that the total dissimilarity between objects assigned to a cluster is minimized, and the clusters are of about the same size. The uniformity between the number of riders and drivers in each cluster ensures that the solution time of solving the maximum weighted bipartite matching problem in all clusters are somewhat similar. As a result, the solution time of solving the problem on different clusters in parallel will be minimized. Finally, we assume that a cluster can be represented by one of the objects in that cluster. Before presenting the formulation, we need to elaborate on how the dissimilarity between every two objects can be measured.

For simplicity, suppose that N objects are ordered such that the first $|R|$ objects are rider trips and the remaining $|D|$ objects are driver trips, i.e. $n \in \{1, \ldots, |R|, |R| + 1, \ldots, |R| + |D| = N\}$. As a result, one can define matrix $C = [c_{nk}],\ \forall\ (n, k) \in \{1, \ldots, N\} \times \{1, \ldots, N\}$ to hold the dissimilarity between objects n and k. Every element c_{nk} in this matrix captures how well trip n can be represented by trip k. As noted in Section 2.3.1, every announced trip contains information on the origin and destination stations, time window, and role specification (rider or driver). Next, we explain how the elements of matrix C can be computed based on the information provided for the announced trips. Given the origin and destination stations of trips n and k, there are six possible scenarios for the order of station visits as demonstrated in Figure 2.4. The incurred detours for scenarios $i = 1, 2, \ldots, 6$ in this figure can be obtained as follows:

$$c^1_{nk} = \tau_t(s^{\mathsf{O}}_n, s^{\mathsf{O}}_k) + \tau_t(s^{\mathsf{D}}_k, s^{\mathsf{D}}_n) + \tau_t(s^{\mathsf{O}}_k, s^{\mathsf{D}}_k) - \tau_t(s^{\mathsf{O}}_n, s^{\mathsf{D}}_n)\ , \tag{2.5a}$$

$$c^2_{nk} = \tau_t(s^{\mathsf{O}}_n, s^{\mathsf{O}}_k) + 2\ \tau_t(s^{\mathsf{O}}_k, s^{\mathsf{D}}_n) + \tau_t(s^{\mathsf{D}}_n, s^{\mathsf{D}}_k) - \tau_t(s^{\mathsf{O}}_n, s^{\mathsf{D}}_n) - \tau_t(s^{\mathsf{O}}_k, s^{\mathsf{D}}_k)\ , \tag{2.5b}$$

$$c^3_{nk} = 0\ , \tag{2.5c}$$

$$c^4_{nk} = \tau_t(s^{\mathsf{O}}_k, s^{\mathsf{O}}_n) + \tau_t(s^{\mathsf{D}}_n, s^{\mathsf{D}}_k) + \tau_t(s^{\mathsf{O}}_n, s^{\mathsf{D}}_n) - \tau_t(s^{\mathsf{O}}_k, s^{\mathsf{D}}_k)\ , \tag{2.5d}$$

$$c^5_{nk} = \tau_t(s^{\mathsf{O}}_k, s^{\mathsf{O}}_n) + 2\ \tau_t(s^{\mathsf{O}}_n, s^{\mathsf{D}}_k) + \tau_t(s^{\mathsf{D}}_k, s^{\mathsf{D}}_n) - \tau_t(s^{\mathsf{O}}_n, s^{\mathsf{D}}_n) - \tau_t(s^{\mathsf{O}}_k, s^{\mathsf{D}}_k)\ , \tag{2.5e}$$

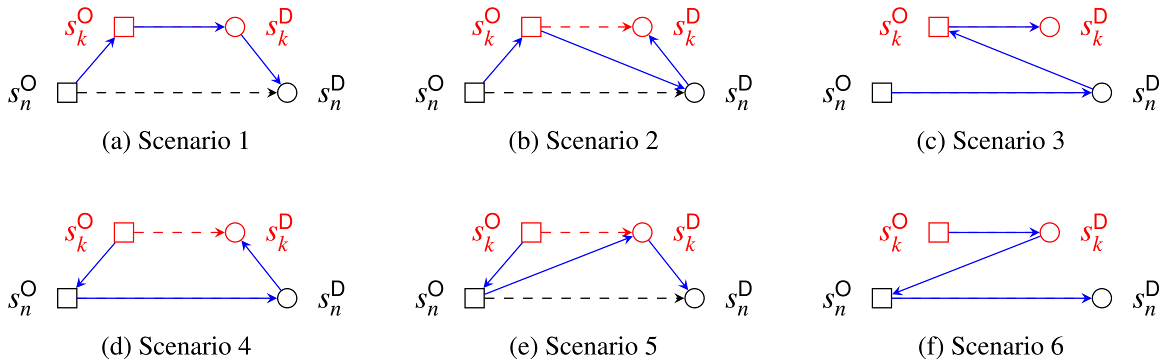
$$c^6_{nk} = 0\ . \tag{2.5f}$$

As a result, the dissimilarity between trips n and k can be calculated as:

$$c_{nk} = \left|(t_n^{\mathsf{ED}} + t_n^{\mathsf{LA}}) - (t_k^{\mathsf{ED}} + t_k^{\mathsf{LA}})\right| + \frac{1}{4}\sum_{i=1}^{6} c_{nk}^i \,. \tag{2.6}$$

The first component in Equation (2.6) measures the temporal proximity of the two trips by finding twice the absolute difference between the centers of their time windows. The second component in Equation (2.6) measures the spatial proximity of the two trips, which can be computed as the average detours incurred by visiting the origin and destination of both trips over scenarios in which the trips are shareable (i.e. scenarios 1, 2, 4 and 5). It is worth mentioning that both components in Equation (2.6) are expressed in the same unit, say minutes.

Having computed the dissimilarity matrix, the proxy problem can be formulated as an integer programming model presented in (2.7). Let us define the binary decision variable q_{nk} to take value 1 if trip $n \in \{1, \dots, N\}$ is represented by trip $k \in \{1, \dots, N\}$, and value 0 otherwise. We further define another binary decision variable, denoted by ζ_k, that takes value 1 if trip $k \in \{1, \dots, N\}$ is a cluster representative, and value 0 otherwise. The objective function in Statement (2.7a) minimizes the total dissimilarity between trips and their associated cluster representatives. Constraint (2.7b) restricts the number of clusters to K. Constraint (2.7c) ensures that each trip is assigned to a single cluster. Constraint (2.7d) serves two purposes. First, it ensures that the difference between the number of rider trips in any two clusters is at most $\varepsilon|R|$. Second, it ensures that rider trips can be represented by trip k only if trip k is chosen as a cluster representative. Similarly, constraint (2.7e) ensures that the difference between the number of drivers in any two clusters is at most $\varepsilon|D|$, and that driver trips



(a) Scenario 1
(b) Scenario 2
(c) Scenario 3
(d) Scenario 4
(e) Scenario 5
(f) Scenario 6

Figure 2.4: Different scenarios for the order of visiting stations. Dashed arrows represent the trips and the blue solid arrows demonstrate the path of visiting stations

can be assigned to trip k only if trip k is chosen as a cluster representative.

$$\min \quad \sum_{n=1}^{N}\sum_{k=1}^{N} c_{nk}\, q_{nk} \tag{2.7a}$$

$$\text{s.t.} \quad \sum_{k=1}^{N} \zeta_k = K\,, \tag{2.7b}$$

$$\sum_{k=1}^{N} q_{nk} = 1\,, \qquad \forall\, n \in \{1,\ldots,N\}\,, \tag{2.7c}$$

$$\sum_{n=1}^{|R|} q_{nk} \leq (1+\varepsilon)\frac{|R|}{K}\zeta_k\,, \qquad \forall\, k \in \{1,\ldots,N\}\,, \tag{2.7d}$$

$$\sum_{n=|R|+1}^{N} q_{nk} \leq (1+\varepsilon)\frac{|D|}{K}\zeta_k\,, \qquad \forall\, k \in \{1,\ldots,N\}\,, \tag{2.7e}$$

$$q_{nk} \in \{0,1\}\,, \tag{2.7f}$$

$$\zeta_k \in \{0,1\}\,. \tag{2.7g}$$

The problem in model (2.7) is a generalization of the capacitated k-median problem, as we further divide the objects into two types of rider trip and driver trip. Since this problem is NP-hard (Byrka et al., 2016), we devise a polynomial-time heuristic based on Lloyd's algorithm. Lloyd's algorithm is an iterative method that has been implemented in several well-known clustering algorithms, such as k-means and expectation-maximization (EM). It is a 2-step heuristic that iteratively assigns objects to their nearest cluster centers, and adjusts cluster centers by choosing the object that is closest to all the objects in that cluster. Our proposed algorithm closely resembles the application of Lloyd's algorithm in k-medoids (Friedman et al., 2001), but modifies it by adding an optimization problem to step 2 (where objects are assigned to given cluster representatives to ensure the uniformity of the clusters within a certain threshold). In the absence of the uniformity constraint, given cluster representatives, the total system-wide within-cluster dissimilarity is minimized when each point is assigned to the cluster with the nearest representative (Friedman et al., 2001). In our specific problem of interest, however, the ultimate goal is to reduce the complexity of the matching problems associated with clusters. Consequently, we need to adjust this allocation rule by requiring uniformity in cluster sizes.

Our proposed solution methodology is described in Algorithm 2.1. The algorithm is initialized by calculating the dissimilarity matrix C, and randomly selecting K rider trips as cluster representatives. In step 1, objects are assigned to clusters by solving the optimization problem in model (2.8). Constraint (2.8b) ensures that each trip is assigned to a single cluster, and constraints (2.8c) and (2.8d) guarantee an ε-uniform distribution of riders and drivers between the K sub-graphs, respectively. We are taking the integer part of the expressions on the left-hand sides of constraints (2.8c)-(2.8d) to obtain an integer right-hand side vector, which combined with the totally-unimodular structure of the constraint set (see Proposition A.1), enables us to relax the binary constraints on the decision variables. Constraint set (2.8e) relaxes the binary constraint on variable q. Note that model (2.8) can be transformed into an unbalanced Hitchcock Transportation problem (see Proposition A.2). For such problems with n source nodes and k sink nodes (where $k \ll n$), Brenner (2008) developed an efficient algorithm with a worst-case running time of $O(nk^2(\log n + k \log k))$. Hence, the problem can be solved in polynomial time.

In step 2, cluster representatives are updated by finding the objects that has the least total dissimilarity with all the objects within their clusters. The algorithm terminates when the change in the objective function in Statement (2.8a) is bound by a pre-determined threshold (0 in our

Algorithm 2.1: The ε-uniform partitioning algorithm

```
Input: $N$ trips including $|R|$ rider trips and $|D|$ driver trips
Output: $K$ uniform clusters of trips
Calculate the dissimilarity matrix $C = [c_{nk}], \forall\ n \in \{1, \dots, N\},\ k \in \{1, \dots, N\}$;
for $\nu = 1, \dots, 10$ do
    Randomly select $K$ trips denoted by $\mathcal{K} = \{n_1, \dots, n_K\}$ as cluster representatives;
    repeat
        Step1 (Assignment)
        Assign $N$ trips to $K$ clusters by solving the optimization problem in model (2.8):

            $\min \sum_{k \in \mathcal{K}} \sum_{n=1}^{N} c_{nk}\, q_{nk}$                                                    (2.8a)
            $\text{s.t.} \sum_{k \in \mathcal{K}} q_{nk} = 1\,, \qquad \forall\ n \in \{1, \dots, N\}\,,$                  (2.8b)
            $\sum_{n=1}^{|R|} q_{nk} \leq \lceil (1+\varepsilon) \frac{|R|}{K} \rceil\,, \qquad \forall\ k \in \mathcal{K}\,,$   (2.8c)
            $\sum_{n=|R|+1}^{N} q_{nk} \leq \lceil (1+\varepsilon) \frac{|D|}{K} \rceil\,, \qquad \forall\ k \in \mathcal{K}\,,$  (2.8d)
            $q_{nk} \geq 0\,.$                                                                 (2.8e)

        Step 2 (Update Clusters)
        Update the cluster representatives in $\mathcal{K}$:

            $n_i = \operatorname*{argmin}_{k:q_{k,n_i}=1} \sum_{n=1}^{N} c_{nk}\, q^*_{n,n_i}\,, \qquad \forall\ i = 1, \dots, K\,.$   (2.9)

    until Termination condition;
Take the best result out of 10 trials;
```

experiments), or when a max number of iterations is reached. Note that the structure of graph G is not an input to Algorithm 2.1, and the spatio-temporal compatibility of trips will be captured through the dissimilarity matrix C. Using Algorithm 2.1 each sub-graph can be recursively partitioned into smaller sub-graphs using the same procedure, as needed. After the graph partitioning is done, the optimization problem in model (2.3) needs to be solved for each sub-graph. In the next section, we show how one can benefit from using the proposed partitioning method.

2.3.7 Properties of the ε-uniform Partitioning Algorithm

In the following two subsections, we first prove certain properties of the ε-uniform partitioning algorithm, and next, we compare the computational complexity of the problem before and after partitioning to quantify the effectiveness of the proposed algorithm.

Proposition 2.1. *The objective function of the proxy problem in (2.7) decreases with iterations of Algorithm in 2.1.*

Proof. Since ζ_k is a binary variable, one can easily show that replacing Statements (2.7a) and (2.7c) with Statements (2.10a) and (2.10b), respectively, does not change its optimal solution.

$$\min \quad \sum_{k=1}^{N} \zeta_k \sum_{n=1}^{N} c_{nk} q_{nk} \,, \tag{2.10a}$$

$$\sum_{k=1}^{N} \zeta_k q_{nk} = 1 \,, \qquad \forall\, n \in \{1, \dots, N\} \,. \tag{2.10b}$$

Let us call the new objective function Δ which is a function of both binary variables q_{nk} and ζ_k. In Step 1, the cluster representatives are fixed, and the problem defined by Constraints (2.7d) to (2.7f), (2.10a) and (2.10b) will be reduced to model (2.8), where Δ is a linear function of only q_{nk}. It is obvious that Δ will decrease from the previous iteration by solving the model in (2.8), since it assigns the objects to the clusters so as to minimize Δ.

In Step 2, the assignment variables are fixed, and we find the rider trips that minimize Δ with respect to ζ_k. Let us denote the assignment solution obtained from Step 1 by q^*_{nk}. Thus, we have:

$$\begin{aligned}\Delta &= \sum_{k \in \mathcal{K}} \sum_{n=1}^{N} c_{nk} q^*_{nk} \zeta_k \\ &= \sum_{i=1}^{K} \sum_{k \in \mathcal{K}} \sum_{n=1}^{N} c_{nk} q^*_{n,n_i} \zeta_k + \sum_{i=1}^{K} \sum_{k \notin \mathcal{K}} \sum_{n=1}^{N} c_{nk} q^*_{n,n_i} \zeta_k - \sum_{i=1}^{K} \sum_{k \notin \mathcal{K}} \sum_{n=1}^{N} c_{nk} q^*_{n,n_i} \zeta_k\end{aligned}$$

$$= \sum_{i=1}^{K}\sum_{k=1}^{N}\zeta_k \sum_{n=1}^{N} c_{nk} q^*_{n,n_i} - \sum_{i=1}^{K}\sum_{k\notin\mathcal{K}}\zeta_k\sum_{n=1}^{N} c_{nk}q^*_{n,n_i}\,. \tag{2.11}$$

In Equation (2.11), the first component is the sum of total within-cluster dissimilarity over all K cluster and the second component is a non-positive value. Equation (2.10b) enforces each cluster to have one representative trip and Equation (2.7b) ensures that exactly K rider trips be chosen as cluster centers. As a result, Step 2 clearly minimizes the first component in Equation (2.11). Hence, Δ will decrease with iterations. □

Proposition 2.2. *Algorithm 2.1 converges in a finite number of iterations.*

Proof. Let us denote the objective function in (2.10a) by Δ which is a function of the assignment variable q_{nk} and the cluster representative variable ζ_k. The proposed algorithm consists of two steps. In the assignment step, it solves an assignment problem by minimizing Δ given the cluster representative, subject to a set of uniformity constraints. In the update step, it minimizes Δ by letting each cluster's representative be the object that provides the lowest within-cluster dissimilarity. Thus, we can infer that:

1. There are possibly many but finite number of ways to assign N objects to K clusters.
2. From one iteration to the next, Δ will not increase (see Proposition 2.1). Since we stop the algorithm when Δ does not change significantly between two consecutive iterations (i.e., when representative trips do not change), we can further state that Δ will strictly decrease as iterations proceed.

From 1, we know that the solution set is finite. For each solution in this finite set, there is a unique minimum Δ based on the update step. From 2, we know that the solution should improve, and therefore, we will visit any solution at most once. As such, we conclude that the algorithm in (2.1) converges in a finite number of iterations. □

Proposition 2.3. *Algorithm 2.1 has polynomial time complexity.*

Proof. Computing the dissimilarity matrix, C, requires a running time of $O(N^2)$. As noted earlier, the cluster assignment step of one iteration has a worst-case running time of $O(NK^2(\log N + K \log K))$. Also, the update step requires a running time of $O(\frac{N^2}{K})$, at every iteration. Given the maximum number of iterations, denoted by I, the running time complexity of the ε-uniform partitioning is $O(\max\{N^2, \frac{IN^2}{K}, INK^2(\log N + K \log K)\})$. □

In spite of being effective in lowering the computational complexity of the ride-matching problems, the clustering problem itself may need a considerable amount of time for large N and K. In order to provide faster solutions, Najmi et al. (2017) suggest that the system operator does not need to solve the whole clustering problem every time prior to solving a ride-matching problem at every re-optimization time. Instead, they can only assign current trips to their closest clusters at every re-optimization period, and update clusters less frequently using the trip information of past users. However, in a dynamic ridesharing system with thousands of users entering and leaving the system in a short period of time, the clusters obtained from the trip information of the participants from a few minutes prior may not sufficiently represent the current trips.

As an alternative, we suggest that one can still solve the clustering problem at each re-optimization period, but instead of using all available trips, the operator can only take a sample of participants of size n (where $n \ll N$) to determine the cluster representatives. Next, they can find the dissimilarity of all trips with the cluster representatives and assign them to the clusters by solving step 1 of Algorithm 2.1. Doing so, the running time complexity of the partitioning problem reduces to $\mathcal{O}(NK^2(\log N + K \log K))$. Note that this small sample should have the same ratio of riders and drivers as the original set of participants. We further suggest that for efficiently solving the assignment step with large K, one can replace it with recursively implementing a bisection method that reduces the running time complexity of Step 1 to $\mathcal{O}(N \log N)$. Following these two propositions, the running time complexity of the clustering problem at each period reduces to $\mathcal{O}(\min\{NK, N \log N\})$.

After partitioning, sub-problems are independent of each other and can be solved in parallel. Therefore, the highest computational complexity after partitioning can be attributed to the largest sub-problem. This complexity has two sources: generating the bipartite graph within the cluster, and solving the maximum weighted bipartite matching problem on this graph. In the following, we present upper bounds on the computational requirements of both sources.

Remark 2.1. *After partitioning the matching problem based on Algorithm 2.1, the worst-case computational complexity of the original problem is reduced as follows: Graph generation by a factor of* $\frac{(1+\varepsilon)^2}{K}$*, and solving the matching problem by a factor of at least* $\frac{(1+\epsilon)^2}{K^2}$ *and at most* $\frac{(1+\epsilon)^3}{K^3}$*.*

Proof. The most complex sub-problem generated by Algorithm 2.1 could have $(1+\varepsilon)\frac{|R|}{K}$ riders and $(1+\varepsilon)\frac{|D|}{K}$ drivers, requiring $(1+\varepsilon)^2\frac{|R|\,|D|}{K^2}$ trip comparisons in the worst case to generate the bipartite graph, which is by a factor of $\frac{(1+\varepsilon)^2}{K}$ smaller than the original number of trip comparisons given K processors. Once graph G_k for partition k is generated, we solve a matching optimization problem whose complexity depends on the size of G_k. More specifically, in the case of using the Fredman-Tarjan algorithm to solve the problem in model (2.3), the computational complexity of the original problem is a function of the number of nodes and links, i.e., $(|R| + |D|)L + (|R| + |D|)^2 \log(|R| + |D|)$, which is by a factor of at least $\frac{(1+\epsilon)^2}{K^2}$, if $\mathcal{O}(L) < \mathcal{O}((|R| + |D|)\log(|R| + |D|))$, and by a factor of at most $\frac{(1+\epsilon)^3}{K^3}$, if $\mathcal{O}((|R| + |D|)\log(|R| + |D|)) < \mathcal{O}(L) < \mathcal{O}(|R||D|)$, larger than the complexity of the same method when solving the matching problem in G_k. □

It is obvious that as the number of clusters increases, or when a more uniform distribution of riders and drivers in sub-graphs is required (i.e., lower ε values), the upper bound on the number of computations to generate the graph in the largest sub-problem decreases. However, it is very likely that being less flexible in terms of rider and driver uniformity in sub-problems and/or considering more clusters lead to eliminating more links from the original graph, hence reducing the accuracy of the solution obtained from Algorithm 2.1. Finally, in addition to the fact that partitioning the graph would lead to smaller sub-problems that can be solved faster, a more important contribution of partitioning is solving problems that cannot be solved at all due to the large number of variables and constraints. As we will demonstrate in the next subsection, this type of problem arises more frequently in more complex ride-matching problems, e.g., one-to-many problems.

2.3.8 Extension to More Complex Ride-matching Problems

As mentioned earlier in Section 2.3.1, we assume that participants do not have role flexibility, and can perform a single pre-specified role (i.e., a rider or a driver). Further, we assume that every driver can serve only one rider on their way to their destination. In the next two subsections, we relax one of these assumptions at a time which results in having more complex forms of the ride-matching problem. We also elaborate on how one can extend the proposed partitioning method at each case.

2.3.8.1 One-to-one Ride-matching Problem with Role Flexibility

In a dynamic ridesharing system where all participants have access to a personal vehicle, one can assume that participants need not specify their roles at the time of registering their trips in the system. Instead, they can rely on the ride-share system to find the optimal decision regarding their roles in the system. As such, the set of participants, $P = \{p\}$, is no longer the union of two mutually exclusive sets of riders and drivers, which implies that the ride-matching problem with flexible roles can be modeled as a maximum weighted matching problem in a general graph $G = (P, \Lambda)$. In this graph, edge $\lambda = (p, p')$ exists if and only if participant p' can give a ride to participant p and the associated vehicle miles traveled saving, denoted by $w_{pp'}$, is greater than that of participant p giving ride to participant p'. This problem can be mathematically formulated as an integer programming model in (A.2).

Similar to the one-to-one matching case, there are polynomial algorithms (Galil et al., 1986; Gabow, 1990) to solve this matching problem efficiently for small- to medium-sized problems. In practice, however, the ride-matching problems involve thousands of users for which the ride-share operator needs to first find the set of all possible matches between participants, Λ, and then solve the problem in (A.2) in real time. In order to obtain a near-optimal solution for this problem in real time, one can use the proposed partitioning algorithm with one little adjustment. Inequalities (2.8c) and (2.8d) must be replaced by the following inequality which makes sure that the number of trips in clusters are almost equal:

$$\sum_{n=1}^{N} q_{nk} \leq \lceil (1+\varepsilon)\frac{N}{K} \rceil, \qquad \forall\, k \in \mathcal{K}\,. \tag{2.12}$$

2.3.8.2 One-to-many Ride-matching Problem

As we will demonstrate in the numerical study section, one-to-one matching makes an appropriate modeling framework when the number of riders and drivers in the system are about the same. In a ridesharing system with high discrepancy between the number of available riders and drivers, the system operator can use more complex forms of ride-matching that allow drivers to serve multiple riders while completing their personal trips. In a one-to-many ride-matching problem, for instance, every driver can serve multiple riders, while every rider can be served by a single driver. Therefore, not only is the matching between riders and drivers required, but also is the itinerary on which they share a ride. In what follows, we present a formulation for this problem that is closely adopted from Masoud and Jayakrishnan (2017b) with some minor modifications. This problem can be mathematically formulated as an integer programming model in (A.3). Note that all parameters introduced in Section 2.3.1, except the set of links Π, preserve their definitions in this formulation.

This problem is a special case of general pick-up and delivery problem introduced by Savelsbergh and Sol (1995), which is NP-hard. Therefore, it is not practical to use this formulation for optimizing a large-scale dynamic ridesharing system, where one needs to solve the problem for thousands of users in a short period of time. However, one can use the proposed trip-based partitioning method to solve multiple smaller instances of this problem in parallel, in real-time. In our proposed clustering problem, the defined dissimilarity measure between riders facilitate riders with similar routes being grouped in the same cluster. As a result, the losses due to restricting riders and drivers to one smaller cluster isolated from their peers in the other clusters are expected to be minimal.

2.4 Numerical Experiment

In this section, we use the New York City taxi dataset to showcase the performance of the trip-based ε-uniform partitioning algorithm under different parameter settings. This section starts with a brief overview of the dataset. Next, the parameter settings for the simulation of the dynamic ridesharing system are described. Then, the effect of our partitioning method along with other benchmark methods on different system-level performance metrics will be analyzed for various parameter settings. Finally, we evaluate the performance of the proposed technique on more complex ride-matching problems, namely the one-to-one ride-matching problem with flexible roles and the one-to-many ride-matching problem.

2.4.1 Dataset

In this study we use the New York City Taxi Dataset[1], which was released to the public as a result of a freedom of Information Law request from the New York City Taxi & Limousine Commission. This dataset covers several years of taxi operations in New York city and includes over 6 million trips. The information available for each trip includes the longitude and latitude of the trip origin and destination, and the departure time of the trip.

The data used in this study belongs to the evening peak hour (19:00-20:00) of February 19, 2016, which consists of more than 22000 trips. The selected trips are geographically concentrated in the Manhattan area, and start in the span of one hour. Using data from a peak period of a typical working day allows us to generate graphs that are dense and therefore finding the optimal ride-matching solution on them is challenging.

We assume that the study area consists of 248 pre-defined "stations", denoted by S, where trips start/end. Stations are strategically positioned in the network so as to ensure that there is at least one station within the walking distance (< 0.15 miles) of a typical trip origin/destination. Introducing stations will increase the spatial proximity of trip requests, hence increasing the number of served requests. Toward this end, we form a hexagonal grid of the Manhattan area (see Figure 2.5), and consider the centroids of hexagons as stations.

2.4.2 Experiment Setup and Parameter Settings

In this study, we assume that all trips will be completed on their shortest paths (unless a detour has to be incurred), where the shortest travel times and driving distances between every two stations are extracted from Google Maps API for every re-optimization period. We let re-optimization period have a length of 1 minute. As a result, the ride-matching problem will be solved every 1 minute starting from 18:59. In order to obtain various scenarios for our experiments, we introduce three parameters of "rider-to-driver ratio", "detour time budget", and "penetration rate", denoted by α, β, and γ, respectively. The rider-to-driver ratio represents the ratio between rider trips and driver trips in our experiments. For every trip in our experiments, we randomly generate its role based on this ratio. The detour time budget parameter is considered as the maximum allowable detour time in minutes for each participant. Having an earliest departure time and a time budget, one can compute the latest arrival time of participant p by $t_p^{\mathsf{LA}} = t_p^{\mathsf{ED}} + \beta$. The Penetration rate parameter can be defined as the percentage of all trips available in our dataset that participate in the ridesharing system. As mentioned earlier in Section 3.1, the announcement time of a trip can be a few minutes

[1] http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml

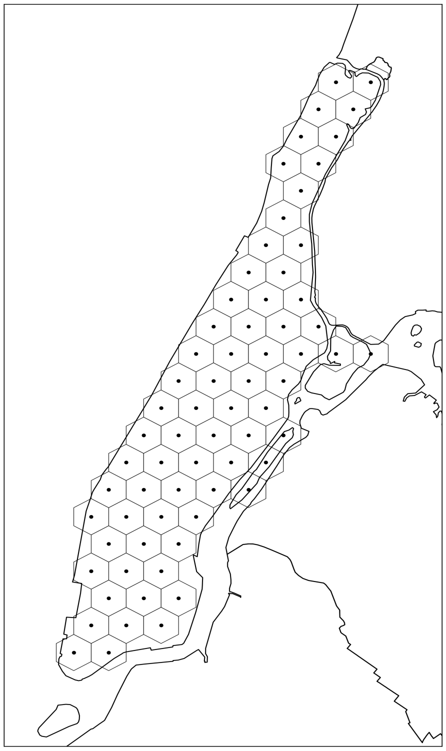


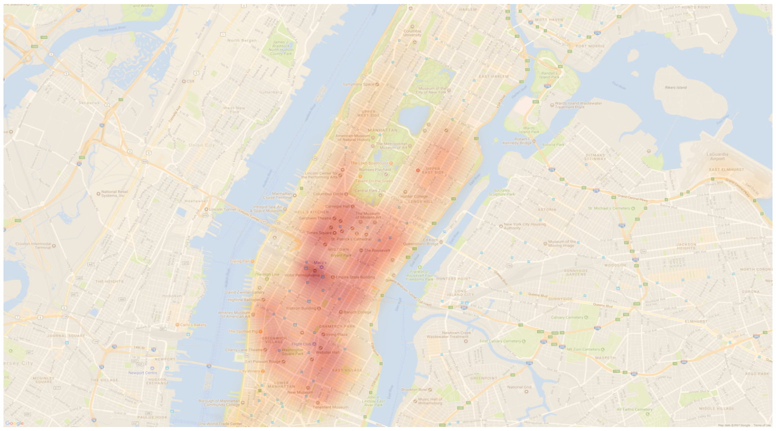
Figure 2.5: A hexagonal grid for the Manhattan area

to a few hours prior to its earliest departure time. Since the NYC taxi dataset does not contain the information regarding the announcement time/request time, we generate a uniform random number from 0 to 60 minutes for each trip and subtract it from its earliest departure time to obtain its announcement time. Doing so allows us to have a mixture of pre-arranged (when random number is close to 60 minutes) and on-the-fly (when random number is close to 0 minutes) trips.

We let the sample size of participants for running algorithm (2.1) at each re-optimization period be 500. Thus, given an arbitrary re-optimization period with a total number of N trips including $|R|$ rider trips and $|D|$ driver trips, we take a sample consisting of $500 \times |R|/N$ riders and $500 \times |D|/N$ drivers to find the K representative trips. Finally, the value of ε in the experiments is set to 0.1, and the maximum number of iterations in algorithm (2.1) is set to 10. In order to evaluate the performance of our trip-based method on one-to-one matching, we repeat the experiments for two other methods of optimal (without partitioning graph G), and point-based (with partitioning method suggested by Najmi et al. (2017) and compare their results with that of the trip-based method.

2.4.3 Results

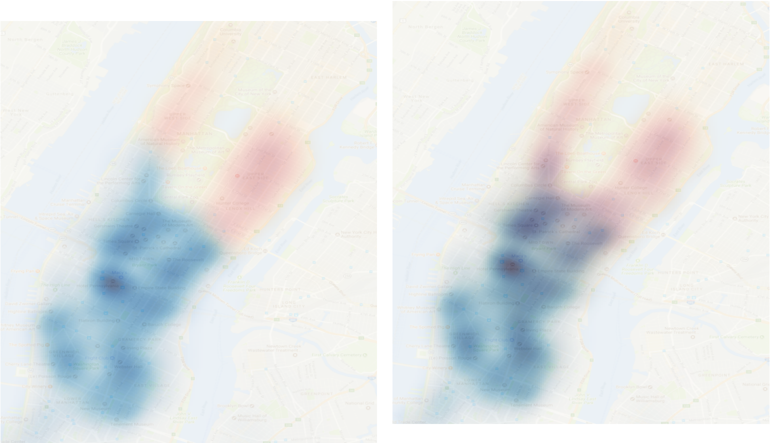
Figure 2.6(a) displays the heatmap of demand in the Manhattan area during the peak hour under study. When facing challenges in operating systems in large-scale areas, the typical approach



(a) Trip end distribution in the Manhattan area

is to split the area geographically into multiple regions, where each region is responsible for its own operations. Although intuitive in the first glance, this approach is not necessarily the best, since breaking the region geographically will come at the opportunity cost of losing trips with ends in different regions. The trip-based partitioning aims at recovering these lost opportunities by considering entire trips, rather than trip ends, as units of partitioning. The result of trip-based partitioning is therefore a network that is clustered into two intersecting regions. Figures 2.6(b) and 2.6(c) demonstrate the regions created by geographic and trip-based partitioning approaches, respectively. Not surprisingly, the two regions in Figure 2.6(c) have high levels of overlap at areas in the middle, and are almost non-intersecting on both ends. Table 2.1 summarizes the performance of the three methods of optimal, trip-based, and point-based by reporting their total VMTS values and computation times. The experiments have been conducted with three penetration rates of 50%, 75%, and 100% (with equal number of riders and drivers), and three cases of 5, 10, and 15 minutes detour time budget. The first scenario can be considered as the base from which all other scenarios are built by changing the value of a single parameter at a time. In addition, each scenario of the partitioning methods is repeated under three different values of 2, 3, and 4 clusters. Each experiment has been replicated with 20 different randomly sampled instances from the data. The matching problems were built, and solved using `GUROBI 9.0` with standard tuning. The rest of the computations were conducted in `Python 3.7`. Also, for parallel computing on partitions, we used the `joblib` package in `Python`. All experiments are implemented on a 3.50 GHz Intel Xeon machine with a 64-bit version of the Windows 10 operating system with 256.0 GB RAM.

For each scenario in Table 2.1, the reported VMTS is the total savings from 60 re-optimization periods averaged over all 20 instances, and the reported time is the maximum computation time over all instances and periods. This is due to the fact that the ride-matching problem is solved at the start of each re-optimization period and is expected to provide a solution before the next period starts. The computation times consist of 3 components of graph generation time, partitioning



(b) Geographic clustering using trip ends. Two colors correspond to two disjoint clusters.

(c) Trip-based ε-uniform graph partitioning. Trip ends are colored based on their assigned clusters. Two colors correspond to two intersecting clusters.

Figure 2.6: Heatmaps of trip ends obtained from the New York taxi data

algorithm time, and optimization solution time. The graph generation time indicates the time it takes to generate the connectivity matrix, i.e., assessing the compatibility of rider and driver trips in time and space. We note that generating the connectivity matrix can be easily parallelized for the optimal method using multiple parallel processing cores. Hence, in order to have a fair comparison between the computation time of the optimal method and the two partitioning methods, we also report the computation time of the optimal method under three scenarios of $K = 2, 3, 4$. The partitioning algorithm time (only for the trip-based and point-based methods) represents the time spent on partitioning the objects (i.e., trip ends or entire trips). Finally, the optimization solution time is the sum of the model construction time and the time spent by the solver to provide a solution.

Table 2.1 clearly shows that the original one-to-one ride-matching problem fails to provide a solution within the length of an optimization period (1 minute) in all scenarios except scenario 2 with a low penetration rate, which implies the need for a heuristic to solve the aforementioned problem faster. Based on the provided optimality gaps, the VMTS by the point-based and trip-based algorithms with 2 partitions closely follow the optimal solutions, with better performance by the trip-based method in all scenarios especially in the case of lower time budgets. Not surprisingly, however, this table indicates that as the number of partitions increases and more links are deleted from the original graph, both partitioning solutions start to deviate more from the optimal solution.

Although the point-based method seems to provide smaller deviations from the optimal solutions when compared with the trip-based method with 2 or 3 clusters, its computation times are at least twice larger than trip-based in all cases. There are at least two reasons to explain this difference.

Table 2.1: The comparison of the VMTS and computation time of the one-to-one ride-matching problem. The value K under Optimal represents the number of available cores and under Point-based and Trip-based represents the number of partitions.

Scenario	α	β	γ	**Optimal**			**Point-based**				**Trip-based**			
		(min)	(%)	K	Computation Time (sec)	VMTS (mile)	K	Computation Time (sec)	VMTS (mile)	Gap (%)	K	Computation Time (sec)	VMTS (mile)	Gap (%)
1	1	10	75	2	107.77	25860.75	2	80.21	25476.15	1.49	**2**	**36.32**	**25712.69**	**0.57**
				3	84.74		3	69.31	25370.93	1.89	3	24.40	24771.48	4.21
				4	74.98		4	70.93	25205.47	**2.53**	4	16.49	24052.63	6.99
2	1	10	50	2	48.22	15798.58	**2**	**48.97**	**15467.03**	**2.10**	**2**	**24.54**	**15675.34**	**0.78**
				3	40.89		3	42.31	15382.59	2.63	3	14.50	15031.27	4.86
				4	36.73		4	37.17	15270.42	3.34	4	8.18	14584.37	7.68
3	1	10	100	2	182.50	36484.08	2	138.70	35998.69	1.33	2	74.91	36366.83	0.32
				3	139.75		3	119.87	35906.21	1.58	**3**	**40.97**	**35003.92**	**4.06**
				4	124.54		4	99.98	35702.19	2.14	4	28.62	34063.42	6.63
4	1	5	75	2	104.99	19297.17	2	62.70	18906.60	2.02	**2**	**38.48**	**19273.85**	**0.12**
				3	79.83		**3**	**61.78**	**18755.62**	**2.81**	3	18.95	18565.03	3.79
				4	67.23		4	59.23	18587.14	3.68	4	12.71	17977.58	6.84
5	1	15	75	2	155.68	28580.18	2	118.62	28236.59	1.20	2	64.35	28212.19	1.29
				3	129.16		3	105.48	28134.69	1.56	**3**	**37.34**	**27173.33**	**4.92**
				4	111.49		4	97.32	28025.14	1.94	4	21.47	26447.68	7.46

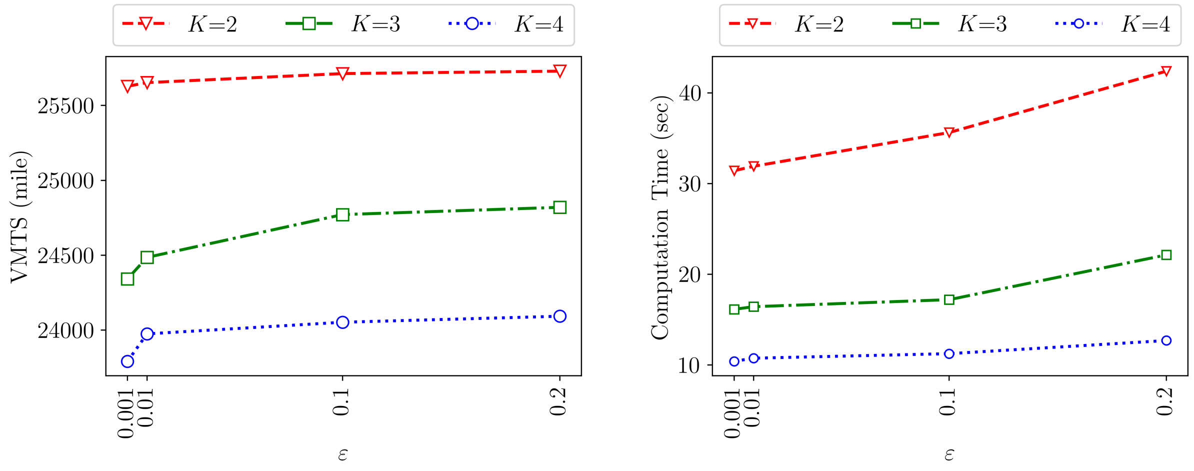
First, as the number of clusters goes up, the number of trips whose origin and destination fall into different clusters in the point-based method increases significantly. Since the point-based method will incorporate these trips in both clusters, the number of trips in every cluster increases, which directly translates into higher computation times for the clusters. Also, having many instances of such trips will result in larger second stage problems in the point-based method, and hence, an increase in the computation time. Second, unlike the trip-based method, the point-based method does not guarantee similar cluster sizes, which causes a large difference between the running time of different partitions, especially in scenarios with high penetration rates and high time budgets. Since the running time of the largest cluster is of the most importance when solving the ride-matching problems in parallel, the point-based method falls short of meeting the minimum computation time requirement.

In order to have a fair comparison between the performance of the partitioning methods, we highlighted the cases with the lowest gaps, where the computation times are less than one minute. one can clearly see the advantage of the trip-based method over the point-based method, as it leads to lower gaps in all scenarios for which the point-based method has a computation time lower than 1 minute. It is worth mentioning that another advantage of our proposed method is in that it accounts for the traffic congestion in the road networks by working based on the driving time distance of two stations instead of only geographical proximity of their locations. It is obvious that the latter case may result in forming poor clusters in regions with heterogeneous traffic conditions.

Figure 2.7 demonstrates the impact of ε on the VTMS and computation time of the trip-based algorithm with different number of clusters. In this experiment, we applied the trip-based partitioning algorithm with different values of ε to all instances of the base scenario. Figure 2.7(a) indicates that the savings increase with ε over all values of K, but the growth rate rapidly decreases as ε becomes larger. This is due to the fact that a larger number of edges will be removed when ε is too small, which results in lower VMTS. However, when ε is large enough, the uniformity constraints will be relaxed, and hence, the trips will be assigned to their closest representative trips. That is the reason why there is no improvement in VMTS when ε changes from 0.1 to 0.2 with 2 or 3 clusters.

Figure 2.7(b) indicates a similar trend between the computation time and the value of ε. By increasing ε, the cluster sizes become less uniform, resulting in the largest sub-graph taking longer to be generated and solved. We note that the effect of ε becomes clearer with smaller number of clusters, since a small change in ε potentially has a larger impact on the size of the partitions, and hence, the computation times.

Figure 2.8 displays the convergence of the objective function in (2.8) using the proposed trip-based algorithm for partitioning an arbitrary instance of the base sceanrio in an arbitrary period. This figure clearly shows that the objective function monotonically decreases until it converges to a local minimum. It also indicates that the algorithm requires only 5 iterations to converge verifying



(a) VMTS

(b) Total time

Figure 2.7: The VMTS and computation time of trip-based partitioning with different values of ε

the choice of 10 as the maximum number of iterations. It is worth mentioning that the partitioning method at every re-optimization period takes less than 1 second to converge in all scenarios. This is beacuse of choosing a constant sample size of 500 trips in all cases.

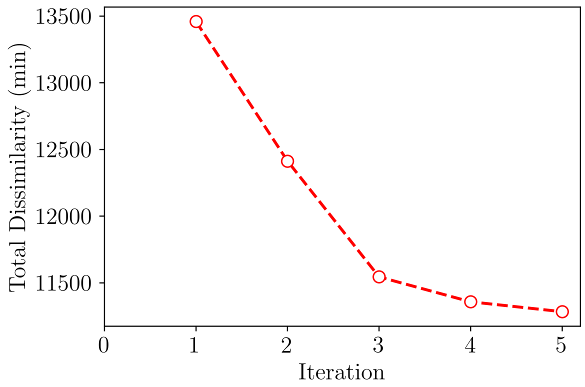


Figure 2.8: The convergence of the total dissimilarity in Algorithm 2.1

Next, we present the performance of the trip-based partitioning algorithm for more complex ride-matching problems summarized in Tables 2.2 and 2.3. More specifically, Table 2.2 compares the VMTS and computation time of the optimal one-to-one ride-matching and the trip-based method for the one-to-many ride-matching problem. We were not able to solve any instance of the original one-to-many ride-matching problem to optimality due to out-of-memory issues, and thus, the result of the one-to-one ride-matching problem is reported instead.

In addition, for our partitioning method, we adopt the bisection method with different levels of recursion. In the first level of recursion, the graph will be partitioned into 2 components. In the

Table 2.2: The VMTS and computation times of the one-to-many ride-matching problem. The value K under Optimal represents the number of available cores and under Trip-based represents the number of partitions.

Scenario	α	β	γ	**Optimal***			**Trip-based**		
		(min)	(%)	K	Computation Time(sec)	VMTS (mile)	K	Computation Time(sec)	VMTS (mile)
1	1	10	75	64	48.53	25860.75	64	261.21	28279.25
				128	41.19		**128**	**42.96**	**26415.01**
6	3	10	75	16	39.54	15553.52	16	456.48	31039.34
				32	33.06		32	73.52	29396.81
				64	32.20		**64**	**39.84**	**28113.42**

* Optimal solution for the one-to-one ride-matching

second level, each of the 2 components obtained in the previous level will be further partitioned into 2 new components, creating a total of 4 sub-graphs, and so on. As mentioned earlier in Section 2.3, this technique helps us find clusters for large values of K(e.g., > 5) more efficiently.

Table 2.2 presents the results for the one-to-many matching case. In these experiments, we assume capacity of 3 for vehicles. Due to the high computational complexity of the one-to-may problems, we applied our methodology only to scenario 1 (base), and a new scenario 6. In scenario 6, all parameters are set to those of the base except the rider-to-driver ratio which has been changed to 3, i.e. the total number of rider trips is 3 times of the number of driver trips in this scenario. The results from the first row of Table 2.2 suggest that solving a one-to-many ride-matching problem in a large-scale dynamic ride-sharing systems with almost equal number of riders and drivers may have no significant advantage over solving the one-to-one ride-matching problem. In our special case, solving the one-to-many problem with 128 partitions yields a solution whose VMTS is very close to that of the one-to-one problem with 2 partitions. There are two reasons for this poor performance. First, when the ratio of riders and drivers is close to 1, the improvement in the VMTS by allowing drivers serve multiple riders will be minimal. Second, the large size of this NP-hard problem enforces us to consider a high number of clusters, which adversely affect the optimality gap.

Nevertheless, as mentioned earlier in Section 2.3, one can take advantage of using the one-to-many ride-matching problem if the ridesharing system experiences a high rate of unbalancedness between the number of riders and drivers. The results from scenario 6 in Table 2.2 confirms this claim, as it shows that one can use 64 clusters to solve this large integer programming model in less than 1 minute while the VMTS is 81% higher than what we can achieve by solving the one-to-one problem in about the same time.

Finally, Table 2.3 summarizes the VMTS and the total time of solving the optimal and trip-based method for the one-to-one matching with flexible roles in the base scenario. In this scenario we assume that all participant are potential riders and drivers. This table indicate that the original problem takes more than 5 minutes to provide the optimal solution. However, our trip-based method reaches an optimality gap of 5.19% while the computation time is improved by a factor of almost 10. In summary, the observations from Tables 2.1, 2.2, and 2.3 suggest that in large-scale systems not only can we use the proposed trip-based graph partitioning algorithm to successfully solve large-scale problems, but these solutions will be of high quality (with optimality gap of < 10%) and can be obtained in a short period of time.

Table 2.3: The VMTS and computation times of the one-to-one ride-matching problem with role flexibility. The value K under Optimal represents the number of available cores and under Trip-based represents the number of partitions.

Scenario	β	γ	**Optimal**			**Trip-based**			
	(min)	(%)	K	Computation Time (min)	VMTS (mile)	K	Computation Time (min)	VMTS (mile)	Gap (%)
7	10	75	3	364.70	33284.23	3	68.46	32237.53	3.15
			4	316.34		**4**	**35.01**	**31558.25**	**5.19**
			5	296.34		5	23.34	31239.75	6.14

2.5 Conclusion

In this chapter, we introduced the trip-based ε-uniform graph partitioning algorithm as a fast and high-quality method to partition graphs that arises in modeling of ridesharing systems. Specifically, we construct this method to solve the maximum weighted bipartite matching problem. Since generating the bipartite graph can turn into a computational bottleneck in real-time implementations, we developed an algorithm to partition a "proxy" of the bipartite graph with the goal of (*i*) minimizing the weight of the deleted links, where the weight of a link is the savings in vehicle miles traveled due to sharing the ride by users who correspond to that link, and (*ii*), ensuring a certain level of uniformity in the size of the resulting sub-graphs, since the size of the graph corresponds to the complexity of the matching problem. We showed that this iterative algorithm terminates in a finite number of steps, and obtains strictly superior partitions with iterations. Furthermore, we demonstrated the worst-case time complexity of finding solutions using the proposed algorithm as a function of problem size and the algorithm's internal parameters. The performance of the proposed algorithm was compared with the optimal solution, as well as one other benchmark algorithm borrowed from the literature, using data from the New York City taxi dataset.

The experiments show that even for a moderate-size ride-matching problem, the large size of the problem may prohibit it from being solved in a timely manner. Furthermore, experiments demonstrated that using trips as units of partitioning as well as incorporating uniformity considerations in the size of the resulting sub-graphs both play important roles in reducing the computational complexity of the problem. Overall, it was shown that the vehicle miles traveled savings produced by the trip-based ε-uniform graph partitioning algorithm follow the optimal results very closely, and outperform the benchmark partitioning method with similar computation times for larger scale ride-matching problems.

Although the proposed algorithm is developed based on a one-to-one ride-matching problem, we show that our methodology can be extended to the more complex forms of ride-matching problem in the literature namely, one-to-many ride-matching problem and one-to-one ride-matching with flexible roles. Conducting experiments using the same dataset indicates that the trip-based ε-uniform graph partitioning shows promising results in terms of providing high quality solutions while using much less computational resources.

CHAPTER 3

Proactive Shuttle Dispatching in Large-Scale Dynamic Ride-Pooling Systems

3.1 Introduction

The problem of dispatching a fleet of shuttles to serve trip requests can be mathematically formulated as a dial-a-ride problem. Although well-studied, the dial-a-ride problem has been traditionally used for dispatching shuttles in para-transit systems, where trip requests are known well ahead of time. With on-demand shared-use mobility services gaining more popularity as a result of high cost of shuttle ownership and the prospect of upcoming driver-less fleet, the real-time application of the dial-a-ride problem is attracting more interest. The successful implementation of on-demand mobility services for large-scale systems requires the dial-a-ride problem to accommodate high-occupancy shuttles that enable pooling requests with similar itineraries. However, the fact that the computational complexity of this problem grows exponentially with number of requests and available seats, and the possibility of changing the itinerary of shuttles en-route in real-time render the current solution methodologies inadequate for online applications.

Typically, mobility services operate under one of two models: (1) having riders register their trip requests ahead of time. This would enable solving the routing and scheduling problems offline, and providing high quality solutions. However, such systems are not convenient for customers since they require in advance planning on their part; (2) serving on-demand trips by heuristically inserting online requests into existing shuttle routes. This approach is much more flexible and more appealing to the customers. However, the insertion heuristics are typically very simple (e.g., assigning a request to the closest shuttle), in the interest of reducing the wait time for the newly joined customer. As such, this approach fails to take into account system-wide quality-of-service measures. This research introduces a general framework to bridge this gap by providing near-optimal insertion methodologies using system-level information, in real time. As such, our framework shifts much of the computational burden of the optimization problems that need to be solved into an offline phase, thereby addressing the on-demand requests with fast and high quality solutions. Furthermore, in order to improve the utilization rate of shuttles, we seek to dispatch our shuttles proactively, and not wait for the demand to be realized first. The framework, therefore, introduces a data-driven approach that starts with a high quality forecast of future trips based on historical data. Using numerical examples, we demonstrate the effectiveness of our methodology applied to a fairly large-scale problem.

The rest of this chapter can be summarized as the following: In Section 3.2 we review the related work in static and dynamic dial-a-ride problem. The general structure and assumptions of the problem are introduced in Section 3.3. In Section 3.4, we first propose a data-driven approach for finding a pool of useful shuttle routes in an offline phase. Further in this section, an online framework for dispatching shuttles in real-time is proposed. The result of a numerical experiment and some interesting aspects of our proposed model are discussed in Section 3.5. We conclude our findings and provide some suggestions for future research in Section 3.6.

3.2 Literature Review

The Dial-a-ride problem (DARP) is a generalization of the pick-up and delivery vehicle routing problem (PDVRP), and vehicle routing problem with time windows (VRPTW). Compared to other types of routing problems (see e.g., Christofides et al. (1981); Laporte and Nobert (1987); Yu et al. (2021)), DARP is usually considered for transporting humans with tight time windows, and the capacity of vehicles are mostly binding (Cordeau and Laporte, 2007). The static version of this problem is a well-studied problem in combinatorial optimization (Healy and Moll, 1995). In static DARP, it is assumed that all requests have been realized prior to the routing and scheduling of the fleet. This problem was initially introduced for scheduling and transporting elderly and disable people.

In recent decades, with the advent of advanced communication technologies and computational tools, the dynamic form of this problem has been the main focus of research. Jaw et al. (1986) were among the first to consider a dynamic DARP with time constraints. They proposed a simple and efficient insertion heuristic that influenced the work of many others afterwards. Interestingly, they showed that considering a batch of customers to simultaneously insert in a route does not result in a large improvement, but it certainly comes at a much higher computational cost. Madsen et al. (1995) considered a DARP with multiple capacities and multiple objective functions for a project by the Copenhagen Fire-Fighting Service (CFFS). They proposed another insertion heuristic that proved to provide a solution in real time. Mitrović-Minić et al. (2004) devised double horizon-based insertion and improvement heuristics that account for both short-term and long-term decisions. Based on their experiments, considering the slack time in the distant future may result in decreasing the routing costs. Coslovich et al. (2006) developed a two-stage insertion heuristic based on route perturbations. In the offline phase, this method creates a feasible neighborhood for the current route of a vehicle that is moving between 2 stops. Then, in real-time, it applies a simple insertion rule to decide whether to insert the new request or not. Using numerical examples, they showed that the result is close to that of static problem. However, they relaxed the capacity constraint due to its low possibility of being violated. Xiang et al. (2008) proposed a flexible scheduling scheme that can handle different sources of uncertainty in a dynamic DARP. They devised a heuristic approach based on local search and introduced a secondary objective function to diversify their search. Häme (2011) proposed an adaptive insertion algorithm that can find the optimal solution for static and dynamic single-vehicle DARP with time windows in small instances. For larger instances, however, this approach uses a local search heuristic. They further analyze the complexity of their approach and test it by various numerical experiments. Ma et al. (2013), using the lazy shortest path calculation strategy, devised a schedule allocation algorithm to insert a new user's trip request into the schedule of the taxi that satisfies the query with minimum incurred travel distance for the ridesharing system.

Maalouf et al. (2014) developed an efficient dynamic fuzzy logic algorithm to insert a request in vehicles. They further considered two objectives in their algorithm to optimize the trade-off between level of service (customers' detour times) and company's performance (miles travelled by vehicles). Ritzinger et al. (2016) proposed a hybrid meta-heuristic that combines an exact dynamic programming with a meta-heuristic that is based on dynamic programming into a large neighborhood search. One advantage of their model is that it does not require a commercial MIP solver. Jafari et al. (2016) proposed a dial-a-ride system where each passenger places a bid for their request and specifies their pick-up and drop-off time windows. The operator selects the requests considering vehicle capacities, placed bids, and the operating cost to maximize the net profit.

In the earlier studies reviewed in this section, researchers devised reactive solutions to respond to the dynamic nature of requests in a DARP. In a more advanced work, Bent and Van Hentenryck (2004) proposed a multiple scenario approach that continuously generates routing plans based on realized and anticipated customers to maximize the number of served requests in a stochastic dynamic DARP. They further introduce a consensus function to rank these plans in real time. Using numerical experiments, they show the prominent advantages of using stochastic information in vehicle routing. In another seminal work, Ichoua et al. (2006) introduced a real-time setting that accounts for the future request arrivals by strategically inserting some dummy customers in vehicle routes. They also use a parallel Tabu Search algorithm to investigate the neighborhood of current routes in real time. Schilde et al. (2011) considered a dynamic and stochastic DARP for a partially dynamic para-transit system. In this system, the inbound trips are scheduled ahead of time, but the outbound trips are unknown and will be realized in real time. In this work, they suggest using historical data to predict the demand for return trips. Applying different meta-heuristic techniques in the literature, they showed that incorporating trip information based on historical data can lead to large improvements.

Another scenario-based heuristic was introduced by Ghilas et al. (2016) to solve a dynamic and stochastic DARP in the existence of some fixed scheduled lines. In this study, they developed an adaptive large neighborhood search algorithm inside a sample average approximation algorithm. Using numerical experiments, they showed the impact of integrating their pick-up and delivery vehicles with scheduled lines. Ulmer et al. (2017) introduced a route-based Markov decision process (MDP) similar to the conventional MDP for routing problems, except that the definition of a state also includes route plans that will be updated in different steps. This change in the definition of state also gives rise to a change in the definition of the reward function. Wei et al. (2017) introduced a look-ahead insertion policy for a shared-taxi system that inserts requests based on a trade-off between operating cost, wait time, and detour time. They further devised a reinforcement learning approach that exploits historical data to learn the parameter governing this trade-off. Alonso-Mora et al. (2017) presented a novel framework that assigns customers to vehicles using a reactive anytime optimal algorithm, and re-positions the idle vehicles using an LP formulation. They further claim that their approach can be easily adjusted to accommodate historical data. They conducted a comprehensive study on the New York City taxi dataset where they found that almost all taxi trips can be satisfied by much fewer vehicles when sharing rides is allowed. Lowalekar et al. (2018) proposed a multi-stage stochastic optimization formulation that incorporates future demand from historical data. They also presented a worst-case theoretical bound on the performance of their algorithm. Using numerical experiments on the New York Taxi dataset, they show the advantages of their proposed method over other algorithms in the literature. More recently, Li et al. (2019) suggested deploying vans proactively by using multiple scenarios of future requests and traffic conditions to generate and evaluate potential decisions for the position of vans. Yu and Shen (2019) devised a novel approach based on Approximate Dynamic Programming (ADP) to solve the dynamic ride-pooling problem with at most two passenger groups. Using numerical experiments on the New York City taxi dataset, they showed the advantages of their proposed ADP approach over two other benchmarks as a result of including the future demand uncertainty into ride-pooling decision making. Also, for a ridesharing-based evacuation program, Moug et al. (ND) proposed and optimized a two-stage stochastic program and a chance-constrained formulation to ensure high demand fulfillment rates under uncertain spatio-temporal demand. They further introduced a heuristic approach to provide fast, yet conservative, solutions for dynamic cases. For a more comprehensive study of the related works, we refer interested readers to a recent survey by Ho et al. (2018).

This chapter contributes to the literature in the following ways: *i*) We propose a new local search algorithm for generating a pool of useful routes in the offline phase; *ii*) we introduce an efficient min-cost flow problem to find the optimal subset of customers served by a fixed route; and *iii*) we propose an online framework that exploits the information from historical data to deploy shuttles proactively and change their routes if necessary in real time.

3.3 Problem Statement

In this chapter, we study a dynamic dial-a-ride problem for an on-demand shuttle service that provides ride-pooling services to a set of passengers on a (directed) road network. We assume that the study region can be represented by a connected, directed graph $G = (\mathcal{S}, \mathcal{E})$, where $\mathcal{S} = \{1, 2, \ldots, S\}$ denotes the set of stations, and $\mathcal{E} \subseteq \mathcal{S} \times \mathcal{S}$ denotes the set of transportation links. The stations are strategically positioned in the road network to ensure that there is at least one station in the walking distance of a typical trip origin/destination. We further discretize the study time horizon into an ordered set of indexed time intervals of small duration, say 1 minute. We represent this set by $\mathcal{T} = \{0, 1, 2, \ldots, T\}$. We assume that the shortest-path (SP) travel time between every two stations can be extracted from time-dependent travel-time matrices, denoted by $\tau(t, i, j)$. Entry (t, i, j) in this matrix holds the SP travel time between station i and station j if one departs from station i at time t.

Let $\mathcal{H} = \{1, 2, \ldots, H\}$ denote the set of passengers (users) that dynamically enter the system during the study time horizon. The trip of a passenger h can be characterized by its origin and destination stations, denoted by $o(h) \in \mathcal{S}$ and $d(h) \in \mathcal{S}$, respectively, and the earliest departure time (EDT) from the trip origin, denoted by $e(h) \in \mathcal{T}$. In this study, we introduce a constant parameter ω that accounts for the maximum pick-up waiting time of every passenger at their origin station. As a result, the pick-up time of passenger h, represented by $p(h)$, should fall within the interval $[e(h), e(h) + w]$. Furthermore, in order to maintain a certain level of service, we introduce another constant parameter Ω that limits the detour of every passenger from their shortest path. We denote the latest arrival time (LAT) of passenger h at their trip's destination station by $l(h)$, where $l(h) = p(h) + \tau(p(h), o, d) + \Omega$. We assume passengers announce their trips at short notice prior to the start of their pick-up time windows and expect a response from the operator no later than their earliest departure time. We also assume that from historical data and/or survey responses, the distribution of the number of requests for an origin-destination pair, (o, d), that arrive at the system at time interval e can be estimated with mean $\phi(e, o, d)$ (e.g., the time-dependent travel demand may follow a Poisson distribution with rate $\phi(e, o, d)$).

The ride-sourcing company is in charge of a fleet of maximum K homogeneous high-capacity shuttles (also referred to as carpooling vans) denoted by $\mathcal{K} = \{1, 2, \ldots, K\}$, with C available seats for each shuttle, that operate during the time horizon $\mathcal{T}$. Every shuttle starts/ends empty in the beginning/end of the horizon from/at any station. It can be easily shown that our methodology can be simply adapted to special cases where there exist a set of heterogeneous origins and destinations as well as capacities for shuttles. As a result of having discrete sets of time and space, we can represent the time-expanded network in our study with a directed acyclic graph (DAG) $\mathcal{G} = (\mathcal{N}, \mathcal{L})$. Every node $n = (t, i) \in \mathcal{N}$ in this graph is a tuple whose first entry is the time and the second entry is the station in the road network. Moreover, the edge set $\mathcal{L}$ consists of all edges $\ell = ((t, i), (q, j))$ such that $q = t + \tau(t, i, j)$. The route of every shuttle in the study time horizon can be represented by a path (sequence of nodes) in this graph that connects a node with $t = 0$ to a node with $q = T$. Finally we assume that shuttles can wait at any station for any number of time steps as long as they can serve their on-board and scheduled customers within their specified time windows. This assumption implies that all edges of type $\ell = \{((t, i), (t + 1, i)),\ \forall\ i \in \mathcal{S}, t \in \mathcal{T}\}$ also exist in graph $\mathcal{G}$.

Finally, without loss of generality, we assume that the system aims to maximize the total origin-destination shortest-path driving distances of customers, i.e. the summation of the origin-destination shortest-path distances of the trips served by the system. Since fare of customers are usually proportionate to the distance of their trips, this objective function is aligned with the purpose of ride-sourcing companies in maximizing profit. Thus hereafter, we refer to the objective function as "Revenue" with unit of mile. However, note that the proposed methodology is independent of choice of objective function. It is worth mentioning that the introduction of the pool of useful trajectories makes our methodology particularly suitable for a central fleet operator with full control of all shuttles' paths during the specified time horizon, e.g., an autonomous fleet operator.

We end this section by introducing a toy example that enables us to clearly illustrate the proposed routing algorithms in our method.

3.3.1 A Toy Example

Let us consider a road network that consists of 4 stations, as shown in Figure 3.1. In this figure, the shortest-path travel time (in minutes) and driving distance (in miles) of every link is presented by a tuple for that link.

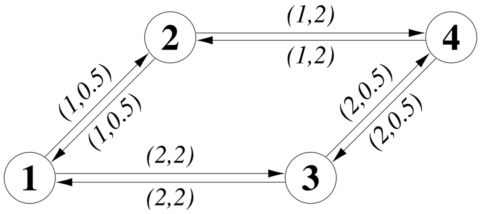


Figure 3.1: The road network of the toy example. The tuples on directed links denote the the shortest-path travel time (in minutes) and driving distance (in miles).

We further assume that the time horizon is 6 minutes with time steps of 1 minute. As a result, the time-expanded network of this example is demonstrated in Figure 3.2. Every node (t, i) in this network represents the spatio-temporal position of a shuttle, e.g., node (3,1) indicates that the shuttle is located in station 1 at time 3. In addition, there exists an edge between node (t, i) and (q, j) if the shuttle can depart from station i at time t and arrive at station j at time q. Hence, any path from the leftmost nodes to the rightmost ones of this network can be considered as a complete route of a shuttle on the road network during the specified time horizon.

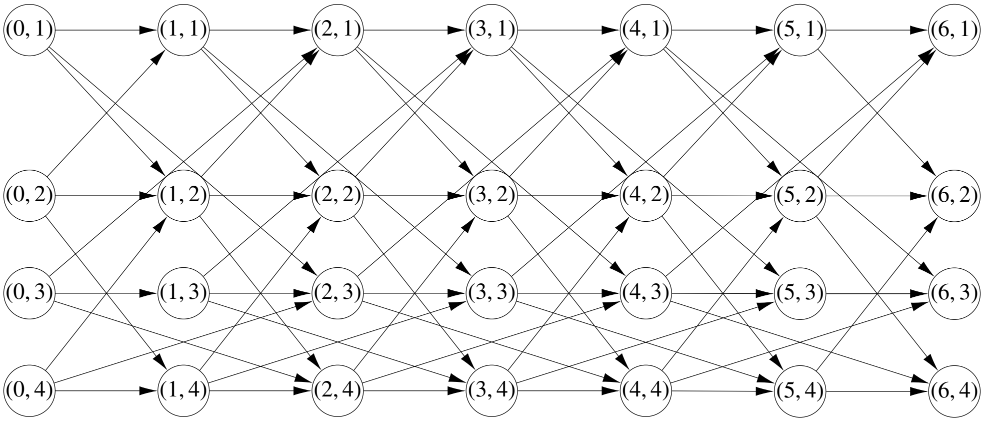


Figure 3.2: The time-expanded network of the toy example. A node n in this time-expanded network is annotated by the tuple (t, s), where t is a time step and s is a physical location on the road network.

The demand forecast for this toy example is provided in Table 1. Every entry (e, i, j) in ϕ represents the average number of requests from origin station i to destination station j that entered the system at time e, e.g. $\phi(2, 1, 4) = 8.0$ tells us that 8.0 requests for travel from station 1 to station 4 are expected to arrive at time 2. We further assume that customers are available for pick-up upon their arrival into the system for 2 time steps, i.e. $\omega = 2$. Thus, as an instance, the total expected number of requests that can be picked up by a shuttle from station 1 at time 2 and be dropped off at station 4 can be calculated as $\phi(1, 1, 4) + \phi(2, 1, 4) = 7.9 + 8.0 = 17.9$. Finally, we assume that only 1 shuttle with 10 seats is available for serving requests, and the maximum allowed detour time, Ω, is set to 3.

Table 3.1: The demand rate of the toy example

$$\phi(0,.,.) = \begin{bmatrix} 0.0 & 3.3 & 5.2 & 8.0 \\ 1.2 & 0.0 & 3.5 & 6.9 \\ 1.6 & 3.6 & 0.0 & 2.1 \\ 5.3 & 6.3 & 9.4 & 0.0 \end{bmatrix}, \phi(1,.,.) = \begin{bmatrix} 0.0 & 2.5 & 5.1 & 7.9 \\ 1.4 & 0.0 & 3.8 & 6.6 \\ 3.2 & 2.8 & 0.0 & 1.2 \\ 6.3 & 5.6 & 8.5 & 0.0 \end{bmatrix}, \phi(2,.,.) = \begin{bmatrix} 0.0 & 2.5 & 5.7 & 8.0 \\ 1.4 & 0.0 & 4.6 & 6.7 \\ 2.2 & 3.6 & 0.0 & 1.0 \\ 5.2 & 5.8 & 7.4 & 0.0 \end{bmatrix}$$

$$\phi(3,.,.) = \begin{bmatrix} 0.0 & 2.7 & 5.0 & 8.0 \\ 0.9 & 0.0 & 5.1 & 6.7 \\ 2.7 & 3.3 & 0.0 & 1.0 \\ 5.9 & 5.9 & 7.4 & 0.0 \end{bmatrix}, \phi(4,.,.) = \begin{bmatrix} 0.0 & 1.8 & 5.1 & 8.3 \\ 1.6 & 0.0 & 0.0 & 7.5 \\ 1.6 & 0.0 & 0.0 & 1.3 \\ 6.3 & 7.1 & 8.2 & 0.0 \end{bmatrix}, \phi(5,.,.) = \begin{bmatrix} 0.0 & 2.3 & 0.0 & 0.0 \\ 1.0 & 0.0 & 0.0 & 6.7 \\ 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 5.7 & 0.0 & 0.0 \end{bmatrix}$$

3.4 Solution Methodology

In this section, we present a data-driven methodology that exploits the useful patterns hidden in historical data to respond to the dynamic and stochastic nature of the shuttle dispatching problem. This methodology can be clearly divided into two distinct, yet related, parts, which are implemented in offline and online phases. The offline portion of the solution methodology aims to find a pool of near-optimal shuttle routes under different realizations of demand requests. In practice, many different trip patterns can be realized, each giving rise to a distinct set of optimal shuttle routes. These trip patterns are governed by a probability density function that can be learned from the historical data. In this research, for instance, we assume that the arrival of passengers in the system follows a Poisson distribution with a mean estimated from past requests. In the online phase, the shuttles will be proactively routed through one of the previously-devised routes. In case of having accurate demand forecasts, we expect that the shuttles readily serve a good portion of passengers en-route. However, the alternative routes in our pool can be used at any time to change the route of a shuttle toward more promising directions in real-time. In what follows, we first present a method to find this pool of routes in an offline phase. Next, we introduce a procedure to use this pool of routes in an online phase, where the realized demand could be stochastically different from the predicted one.

3.4.1 Offline Planning

As mentioned earlier, the offline phase of our method aims to construct a pool of useful (near-optimal) routes based on historical data. Toward this end, we propose a multi-step framework that takes the time-expanded network under study and the average rate of demand as its inputs, and provides a set of shuttle routes as its output. This general framework, presented in Algorithm 3.1, is built on Algorithms 3.2, 3.3, and 3.4, which will be described in further details later.

Suppose that we are interested in obtaining a set of M alternative routes per shuttle. Let us denote alternative route $m = 1, \ldots, M$ for shuttle k by $\mathcal{P}^k_m$. Also, we refer to the first route ($m = 1$) as the base route of each shuttle. In this framework, we first find the base routes using the average demand forecast ϕ in network graph $\mathcal{G}$. These routes help us determine a tentative direction for each shuttle in time-expanded network $\mathcal{G}$. We assume that each shuttle will be deployed on this primary route in the online phase of our problem.

In order to find the base routes, we first create K initial routes by partitioning the average demand, ϕ, into K approximately uniform clusters, ϕ^k, and then constructing a full route for each cluster ϕ^k using a routing algorithm (which will be described later) in network $\mathcal{G}$. For partitioning ϕ into clusters, we use the methodology described in Tafreshian and Masoud (2020a). After finding the initial base routes, we set the total revenue, denoted by z, to zero, and repeat a clustering

Algorithm 3.1: Offline framework

```
Input: Time-expanded network: $\mathcal{G}$
       Average demand forecast: $\phi$
       Threshold for reducing the graph around base: $\epsilon$
       Number of alternative routes per shuttle: $M$
Output: Pool of shuttle routes: $\mathcal{P}$
Partition the average demand in $\phi$ into $K$ uniform clusters, $\phi^k \quad \forall\, k \in \mathcal{K}$ ;
for $k \in \mathcal{K}$ do
    $\mathcal{P}_1^k \leftarrow$ Find a base route for shuttle $k$ using demand $\phi^k$ in network $\mathcal{G}$ (see Algorithm 3.4) ;
$z \leftarrow 0$ ;
repeat
    $\eta^k(e,o,d) \leftarrow$ Find the distance between every trip $(e,o,d)$ in $\phi$ and base route $\mathcal{P}_1^k$,
      $\forall\, k \in \mathcal{K}, \forall\, e \in \mathcal{T}, \forall\, o \in \mathcal{S}, \forall\, d \in \mathcal{S}$;
    Create the clustering problem:

        min   $\sum_{k=1}^K \sum_{(e,o,d)} \eta^k(e,o,d)\, \phi^k(e,o,d)$                                      (3.1a)
        s.t.  $\sum_{k=1}^K \phi^k(e,o,d) = \phi(e,o,d)\,,$     $\forall\, k \in \mathcal{K}, \forall\, e \in \mathcal{T}, \forall\, o \in \mathcal{S}, \forall\, d \in \mathcal{S}\,.$   (3.1b)

    for $r = 1, 2, \dots, R$ do
        $\phi^k \leftarrow$ Distribute demand between shuttles by solving the clustering problem in 3.1;
        for $k \in \mathcal{K}$ do
            $\mathcal{P}_1^k \leftarrow$ Find a base route for shuttle $k$ with demand $\phi^k$ in network $\mathcal{G}$ (see Algorithm 3.4) ;
            $\gamma_1^k, f_1^k \leftarrow$ Find optimal revenue and served requests given $\mathcal{P}_1^k$ and $\phi^k$ (see Algorithm 3.3);
            Add $\phi^k(e,o,d) \leq f_1^k(e,o,d)$ to the problem in 3.1,   $\forall\, e \in \mathcal{T}, \forall\, o \in \mathcal{S}, \forall\, d \in \mathcal{S}$;
        $z \leftarrow \sum_{k=1}^K \gamma_1^k$ ;
until $z$ converges;
for $m = 1, 2, \dots, M$ do
    $\Phi_m(e,o,d) \leftarrow$ Sample from $Poi(\phi(e,o,d)) \quad \forall\, e \in \mathcal{T}, \forall\, o \in \mathcal{S}, \forall\, d \in \mathcal{S}$ ;
    $\eta^k(e,o,d) \leftarrow$ Find the distance between every trip $(e,o,d)$ in $\Phi_m$ and base route $\mathcal{P}_1^k$,   $\forall\, k \in \mathcal{K}$,
      $\forall\, e \in \mathcal{T}, \forall\, o \in \mathcal{S}, \forall\, d \in \mathcal{S}$;
    Create the clustering problem:

        min   $\sum_{k=1}^K \sum_{(e,o,d)} \eta^k(e,o,d)\, \Phi_m^k(e,o,d)$                                     (3.2a)
        s.t.  $\sum_{k=1}^K \Phi_m^k(e,o,d) = \Phi_m(e,o,d)\,.$     $\forall\, k \in \mathcal{K}, \forall\, e \in \mathcal{T}, \forall\, o \in \mathcal{S}, \forall\, d \in \mathcal{S}\,.$   (3.2b)

    for $r = 1, \dots, R$ do
        $\Phi_m^k \leftarrow$ Distribute demand between shuttles by solving the clustering problem in 3.2;
        for $k \in \mathcal{K}$ do
            $\mathcal{G}_m^k \leftarrow$ Reduce $\mathcal{G}$ to a network around $\mathcal{P}_1^k$ using $\epsilon$ and $\Phi_m^k$ (see Algorithm 3.2);
            $\mathcal{P}_m^k \leftarrow$ Find an alternative route for shuttle $k$ with demand $\Phi_m^k$ in network $\mathcal{G}_m^k$ (see
              Algorithm 3.4) ;
            $f_m^k \leftarrow$ Find optimal served requests given $\mathcal{P}_m^k$ and $\Phi_m^k$ (see Algorithm 3.3);
            Add $\Phi_m^k(e,o,d) \leq f_m^k(e,o,d)$ to the problem in 3.2,   $\forall\, e \in \mathcal{T}, \forall\, o \in \mathcal{S}, \forall\, d \in \mathcal{S}$;
```

and routing procedure until the total revenue converges to a fixed value. At each iteration of this procedure, we first calculate the distance between every trip in ϕ and current base routes, and assign the demand to the closest shuttle by solving the problem in 3.1. For measuring the distance between trip (e, o, d) in ϕ and base route $\mathcal{P}_1^k = [(t_1, s_1), (t_2, s_2), \ldots, (t_x, s_x)]$, we use the equation:

$$\eta^k(e, o, d) = \min_{\substack{e \le p \le e+\omega \\ p+\tau(p,o,d) \le d \le p+\tau(p,o,d)+\Omega \\ (t_i,s_i),(t_j,s_j) \in \mathcal{P}_1^k}} \left\{ |t_i - p| + \tau(p, o, s_i) + |t_j - d| + \tau(l, d, s_j) \right\}, \tag{3.3}$$

which finds the spatio-temporal distance between the origin and destination of the trip with their closet nodes in the route. After assigning the demand to the closest shuttle, we use the routing algorithm in 3.4 to adjust the base routes for each shuttle. We also find the optimal revenue, γ_1^k, and served requests, f_1^k, for each shuttle using a min-cost flow algorithm (which will be described later in Algorithm 3.3). Since all requests for a given origin-destination pair may not be served by its closest shuttle, we repeat this process (for R times) by assigning the unserved demand to its next closest shuttle.

Note that using the average demand forecast enables us to take into account the average travel pattern. However, relying merely on average demand has two drawbacks. First, it does not necessarily represent a feasible demand profile due to continuity of demand rate for every (t, i, j). Second, it does not capture the variability in demand. In order to mitigate these issues, we realize $M - 1$ different demand profiles from the Poisson distribution with average rate of ϕ, and construct an alternative route per each demand profile. Let us denote these demand profiles with $\Phi_m, \ \forall \ m = \{2, 3, \ldots, M\}$. It is worth mentioning that there is no specific priority among different shuttles, and indices 1 through K do not represent any order among them. Also, there is no specific priority among the alternative routes of a single shuttle, and indices 2 through M for each shuttle do not represent any order among these routes.

Before finding an alternative route $m \in \{2, 3, \ldots, M\}$ for shuttle k, we introduce a 2-step procedure (see Algorithm 3.2) to reduce the time-expanded network to a graph in the neighbourhood of the base route. In the first step, we find a sub-graph of $\mathcal{G}$, denoted by $\mathcal{G}'$, in which every node has a spatio-temporal distance of maximum ϵ from its closest node in the base route. In the second step, we first find the optimal set of passengers (from the demand table Φ_m^k) that can be served by the base route $\mathcal{P}_1^k$ (using Algorithm 3.4 which will be discussed later). Let us denote the set of spatio-temporal nodes at which the served passengers are picked up by $\mathcal{N}_f$. Then, we obtain a sub-graph of $\mathcal{G}'$ in which every node is accessible by nodes in $\mathcal{N}_f$, i.e. there exists a path between every node $n' \in \mathcal{N}_f$ and every node n in $\mathcal{G}'$. Note that the accessibility is a two sided relationship, and the pair order just depends on the time dimension of the two nodes.

Using this procedure, one can ensure that (1) there are some common spatio-temporal nodes

Algorithm 3.2: Reduce time-expanded network around a route

```
Input: Time-expanded network: $\mathcal{G}$
       A realization of demand: $\Phi$
       Base route: $\mathcal{X}$
       Threshold for reducing the graph around base: $\epsilon$
Output: Reduced time-expanded network: $\mathcal{G}''$
Step 1:
$\mathcal{N}' \leftarrow \emptyset$ ;
for $n \in \mathcal{N}$ do
    for $n' \in \mathcal{X}$ do
        if length of Shortest Path (SP) between $n$ and $n'$ in $\mathcal{G} \leq \epsilon$ then
            $\mathcal{N}' \leftarrow \mathcal{N}' \cup \{n\}$
$\mathcal{G}' \leftarrow$ sub-graph of $\mathcal{G}$ induced by $\mathcal{N}'$ (West et al., 1996) ;
Step 2:
$f \leftarrow$ Find optimal served requests given $\mathcal{X}$ and $\Phi$;
$\mathcal{N}_f \leftarrow$ Find stations in $\mathcal{X}$ at which served requests in $f$ are picked up ;
$\mathcal{N}'' \leftarrow \emptyset$ ;
for $n \in \mathcal{N}'$ do
    for $n' \in \mathcal{N}_f$ do
        if node $n$ is accessible by $n'$ in $\mathcal{G}'$ then
            $\mathcal{N}'' \leftarrow \mathcal{N}'' \cup \{n\}$ ;
$\mathcal{G}'' \leftarrow$ sub-graph of $\mathcal{G}'$ induced by $\mathcal{N}''$ ;
```

between different alternative routes in case the shuttle is required to change its itinerary en-route in real-time, and (2) the set of passengers that can be served by different routes are highly overlapping, and thus, changing the route is not likely to violate the destination of on-board passengers.

Suppose, as an instance in our toy example, the base route is $\mathcal{X} = \{(0,1),(1,2),(2,1),(3,2),(4,1),(5,2),(6,1)\}$ and $\epsilon = 1$. By taking the first step, the time-expanded network can be reduced as shown in Figure 3.3. In this network, every node is either on the path (black solid lines), or 1 minute away from it (black dashed lines). Now, suppose we take a Poisson sample from the average demand in Table 3.1 as shown in Table 3.2. Based on the realized demand table and number of available seats, a shuttle can optimally pick up 1, 4, and 2 customers at nodes (2, 1), (3, 2), and (4, 1), respectively, on the specified path. Hence, we set $\mathcal{N}_f = \{(2,1),(3,2),(4,1)\}$. Figure 3.4 shows the result of applying the second step using this information. This figure clearly shows that every complete route from the leftmost nodes to the rightmost ones must pass through all nodes in $\mathcal{N}_f$.

In order to find alternative route m for each shuttle, we follow the same process of clustering and routing as in finding the base routes, with random demand forecast Φ_m^k and sub-graph $\mathcal{G}_m^k$. This procedure will be repeated $M - 1$ times to yield a pool of $K \times M$ routes. In the next two parts, we elaborate on two of the sub-procedures required to implement Algorithm 3.1 – we first propose

Table 3.2: A demand profile for the toy example

$$\Phi = \left[\begin{bmatrix} 0 & 0 & 7 & 4 \\ 2 & 0 & 5 & 7 \\ 3 & 3 & 0 & 0 \\ 5 & 5 & 9 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 4 & 9 \\ 2 & 0 & 5 & 6 \\ 4 & 5 & 0 & 1 \\ 7 & 7 & 8 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 2 & 8 \\ 3 & 0 & 3 & 4 \\ 5 & 4 & 0 & 3 \\ 8 & 6 & 5 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 4 & 8 & 3 \\ 1 & 0 & 2 & 3 \\ 1 & 2 & 0 & 1 \\ 1 & 6 & 9 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 2 & 1 & 6 \\ 1 & 0 & 0 & 8 \\ 0 & 0 & 0 & 0 \\ 3 & 4 & 9 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 \\ 0 & 0 & 0 & 0 \\ 0 & 7 & 0 & 0 \end{bmatrix} \right]$$

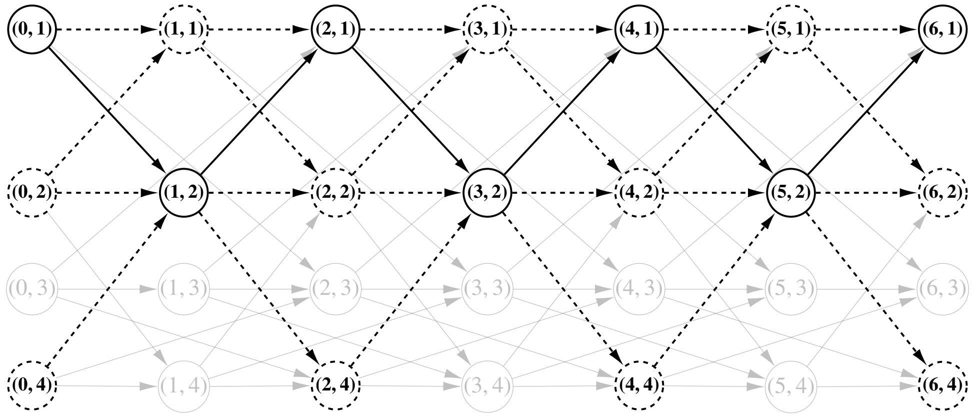


Figure 3.3: The reduced time-expanded network of the toy example in step 1

an algorithm to efficiently find the optimal subset of passengers served by a fixed route. Next, we introduce a local search algorithm to find the route of a single shuttle based on a demand table.

3.4.1.1 Finding the Optimal Set of Passengers Served by A Fixed Route

Suppose that we have a capacitated shuttle with a fixed route $\mathcal{X} = [n_1, n_2, \ldots, n_x]$, and that there is a set of passengers whose pick-up and drop-off nodes belong to the sequence of nodes in $\mathcal{X}$. We are looking for the best subset of passengers that can be served by this shuttle. In this section,

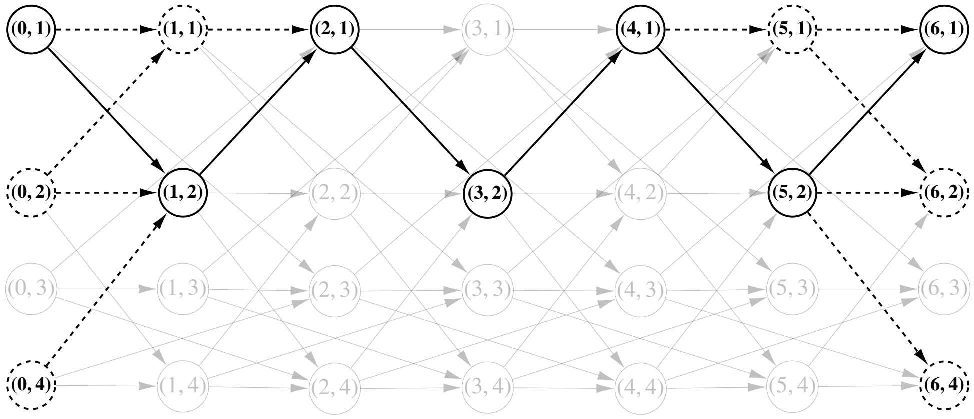


Figure 3.4: The reduced time-expanded network of the toy example in step 2

we propose an optimal method to solve this problem efficiently. This method is motivated by the concept of k-coloring in interval graphs. Next, we briefly describe the maximum weighted k-coloring problem and show how it is related to the problem at hand.

In graph theory, maximum weighted k-coloring of intervals is a well-known problem. In this problem, one is required to color a set of n' open intervals (from the real line), denoted by $I_1, I_2, \dots, I_{n'}$, with a set of K colors, such that two intersecting intervals receive distinct colors. Every interval is also associated with a positive weight of $w_i \; \forall \; i \in \{1, 2, \dots, n'\}$. Further, suppose that $v_1 < v_2 < \cdots < v_r$ represent the unique set of ordered endpoints of these intervals. It is shown in (Carlisle and Lloyd, 1995) that this problem can be formulated as a min-cost flow problem in a weighted DAG G' with nodes $s = v_0 < v_1 < \cdots < v_{r+1} = t$ and two sets of edges, called clique edges and interval edges. The clique edges $(v_{i-1}, v_i) \; \forall \; 1 \leq i \leq (r+1)$ have capacity K and cost 0. The interval edges (v_j, v_h), with v_j and v_h being defined as the left and right endpoint of interval $I_i \; \forall \; 1 \leq i \leq n'$, have capacity 1 and cost $-w_i$. Also let the demand and supply of source node s and target node t be K. In what follows, we present a result from (Carlisle and Lloyd, 1995) regarding this graph theoretic problem.

Lemma 3.1. *(Carlisle and Lloyd, 1995) Given a min-cost flow of size K in G', the intervals corresponding to the interval-edges of flow 1 are exactly the intervals in some maximum weight k-coloring of the n' intervals.*

Now, we show that the problem of finding the best subset of passengers to serve by a fixed route is a variant of the k-coloring problem. The trip of every passenger can be considered as an interval from its pick-up node to its drop-off node. We set the weight of an interval equal to the revenue obtained by serving a passenger from its associated pick-up node to its drop-off node. We can also think of the C seats in a shuttle as the distinct colors in the k-coloring problem. Hence, our objective is to assign these seats to passengers such that every seat can be occupied by at most one passenger at any point throughout the fixed route, while maximizing the total revenue obtained by serving these passengers. In Algorithm 3.3, we provide a pseudo-code to formulate the problem at hand as a min-cost flow problem in a weighted DAG, denoted by $\mathscr{G} = (\mathscr{N}, \mathscr{L}, \mathscr{W}, \mathscr{U})$, where $\mathscr{N}, \mathscr{L}, \mathscr{W}$, and $\mathscr{U}$ represent the set of nodes, edges, weights and capacities, respectively.

Note that there are two subtle difference between our problem and the k-coloring problem. First, the passenger intervals are closed from the left (i.e. at their pick-up nodes). In order to handle this issue, we define a drop-off and a pick-up node per each node $n_i \in \mathcal{X}$, denoted by n_i^{d} and n_i^{p}, respectively. As a result, we have two types of clique edges, namely the ones that connect the consecutive drop-off nodes and the ones that connect each drop-off node to its consecutive pick-up node. Second, the intervals in our problem are not unique. In other words, we can have more than one passenger request from node $n_i = (t, i)$ to node $n_j = (q, j)$. As a result, the capacity of interval edges is equal to the number of passengers that are available to be picked up at node $n_i = (t, i)$ and dropped off at $n_j = (q, j)$.

Now, let us describe the above procedure using an instance from our toy example. Suppose that $\mathcal{X} = \{(0, 3), (2, 4), (3, 2), (4, 1), (6, 3)\}$ and we have realized the demand profile Φ in Table 3.2. Table 3.3 shows the number of available customers, obtained from Φ, for each combination of origin-destination stations in route $\mathcal{X}$, e.g., the shuttle can pick up $\Phi(1, 4, 2) + \Phi(2, 4, 2) = 6 + 7 = 13$ customers from station 4 at time 2 and drop them off at station 2 at time 3.

The trip intervals of these customers are depicted in Figure 3.5. The first column, (t, i, q, j), shows the origin-destination combination, the second column, $\rho(t, i, j)$, shows the profit of each interval (i.e. shortest-path driving distance between stations i and j departing from i at time t), and

Algorithm 3.3: Find the sequence of pick-up and drop-offs that provide the optimal revenue, given a fixed route and demand forecast

```
Input: Fixed Route of shuttle: $\mathcal{X} = [n_1, n_2, \ldots, n_x]$
       Demand forecast: $\Phi$
Output: Optimal revenue of shuttle: $\gamma$
        Optimal number of served trips: $f$
$\mathcal{N} \leftarrow \{n^{\mathsf{d}}_1, n^{\mathsf{p}}_1, n^{\mathsf{d}}_2, n^{\mathsf{p}}_2, \ldots, n^{\mathsf{d}}_x, n^{\mathsf{p}}_x\}$ ;
$\mathcal{D}(n^{\mathsf{d}}_1) \leftarrow -C$; $\mathcal{D}(n^{\mathsf{p}}_x) \leftarrow C$; $\mathcal{D}(n^{\mathsf{d}}_i) \leftarrow 0 \ \forall\ i \in \{2, \ldots, x\}$; $\mathcal{D}(n^{\mathsf{p}}_i) \leftarrow 0 \ \forall\ i \in \{1, \ldots, x-1\}$ ;
$\mathcal{L} \leftarrow \{\}$ ;
for $i = 1, 2, \ldots, x-1$ do
    $\mathcal{L} \leftarrow \mathcal{L} \cup \{n^{\mathsf{d}}_i, n^{\mathsf{p}}_i\}; \mathcal{W}(n^{\mathsf{d}}_i, n^{\mathsf{p}}_i) \leftarrow 0; \mathcal{U}(n^{\mathsf{d}}_i, n^{\mathsf{p}}_i) \leftarrow C$ ;
    $\mathcal{L} \leftarrow \mathcal{L} \cup \{n^{\mathsf{d}}_i, n^{\mathsf{d}}_{i+1}\}; \mathcal{W}(n^{\mathsf{d}}_i, n^{\mathsf{d}}_{i+1}) \leftarrow 0; \mathcal{U}(n^{\mathsf{d}}_i, n^{\mathsf{d}}_{i+1}) \leftarrow C$ ;
for $i = 1, 2, \ldots, x-1$ do
    for $j = i+1, i+2, \ldots, x$ do
        $(p, o) \leftarrow n^{\mathsf{p}}_i$ ;
        $(l, d) \leftarrow n^{\mathsf{d}}_j$ ;
        if $d$ not visited again between $n^{\mathsf{p}}_i$ and $n^{\mathsf{d}}_j$ then
            if $(l - p) \leq \tau(p, o, d) + \Omega$ then
                $\mathcal{L} \leftarrow \mathcal{L} \cup \{n^{\mathsf{p}}_i, n^{\mathsf{d}}_j\}$ ;
                if $o$ visited again between $n^{\mathsf{p}}_i$ and $n^{\mathsf{d}}_j$ then
                    $\mathcal{U}(n^{\mathsf{p}}_i, n^{\mathsf{d}}_j) \leftarrow \Phi(p, o, d)$ ;
                else
                    $t_0 \leftarrow$ Last time $o$ visited in time window $[p - \omega, p]$ ;
                    $\mathcal{U}(n^{\mathsf{p}}_i, n^{\mathsf{d}}_j) \leftarrow \sum_{t=t_0+1}^{e} \Phi(t, o, d)$ ;
                $\mathcal{W}(n^{\mathsf{p}}_i, n^{\mathsf{d}}_j) \leftarrow -\rho(p, o, d)$ ;
Define graph $\mathcal{G} = (\mathcal{N}, \mathcal{L}, \mathcal{D}, \mathcal{W}, \mathcal{U})$ ;
Solve min-cost flow on $\mathcal{G}$ ;
$f \leftarrow$ Optimal flow on edges of type $(n^{\mathsf{d}}_i, n^{\mathsf{p}}_j)$ ;
$\gamma \leftarrow -1\times$ Optimal flow cost ;
```

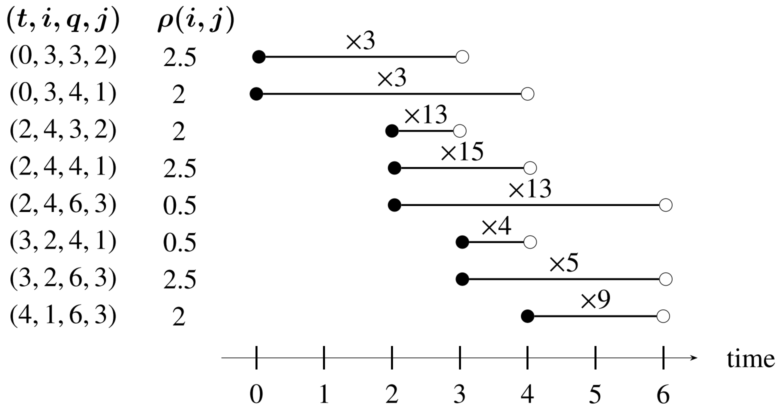


Figure 3.5: Trip intervals of the demand in Table 3.3

the third column shows the trip intervals on a timeline. The value on top of each interval presents the frequency of that trip interval obtained from Table 3.3. This figure clearly indicates that we are not able to serve all of the customers. For instance, at time 2, 3+3+13+15+13=47 intervals intersect while we have only 10 seats in our shuttle. Therefore, we aim to find the most profitable subset of intervals that does not violate the capacity of the shuttle at any time.

As stated above, this can be formulated as a min-cost flow problem depicted in Figure 3.6(a). For each station, we define two nodes of drop-off (dashed line) and pick-up (solid line) except the last one for which we only have a drop-off node. We further show the clique edges with dashed lines and the interval edges with solid lines. The tuple on top of each edge shows the negative of the cost and the capacity of that edge. Finally, we assume that the first drop-off node is a source node with supply of 10 and the last drop-off node is a sink node with demand of 10. The solution to this capacitated min-cost flow problem, depicted in Figure 3.6(b), provides the most profitable subset of customers in Φ that can be served by route $\mathcal{X}$. Note that the reader must distinguish between the capacity of interval edges in min-cost flow problem and the capacity of shuttles, C, as the former represents the number of available customers for a certain origin-destination combination while the latter is a given input that are reflected in the capacity of clique edges. In our toy example, for instance, the capacity of arc $(2, 4, 6, 3)$ is 13 that is higher than capacity of shuttle, 10. That being said, for computational purposes, we can always replace the capacities above C with C since we

Table 3.3: The potential customers for route $\mathcal{X} = \{(0, 3), (2, 4), (3, 2), (4, 1), (6, 3)\}$ in the toy example

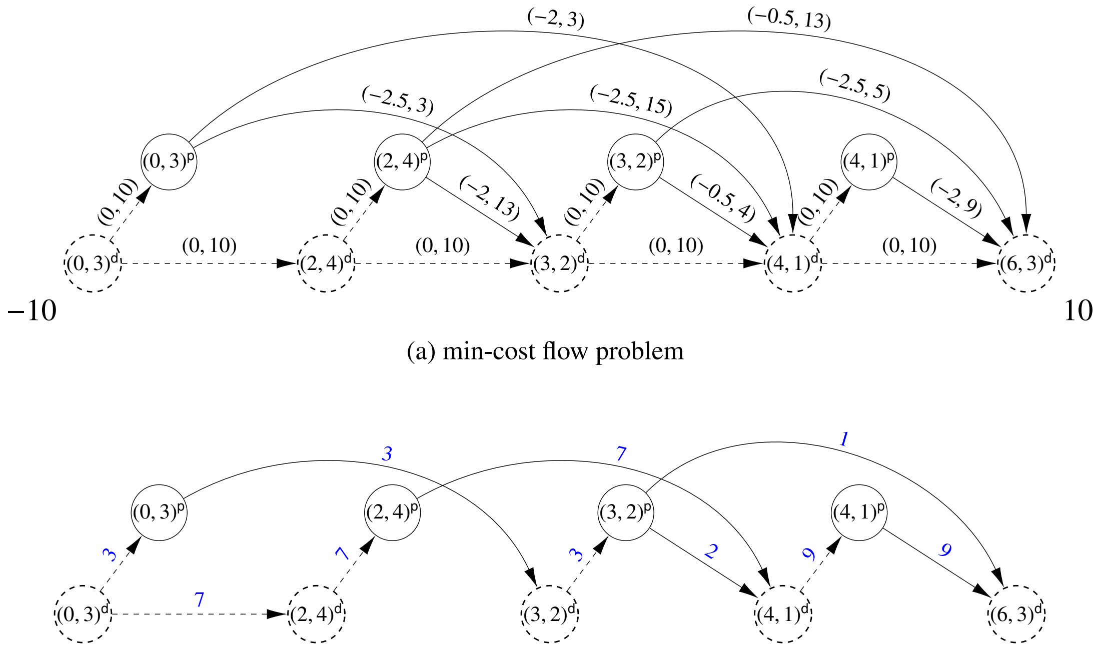
	(2,4)	(3,2)	(4,1)	(6,3)
(0,3)	0	3	3	0
(2,4)		13	15	13
(3,2)			4	5
(4,1)				9

know that there is no way to have a flow of greater than C in this min-cost flow problem.

We end this section by presenting a remark that shows the efficiency of the proposed min-cost flow algorithm.

Remark 3.1. *The optimal subset of passengers served by a single shuttle that visits a fixed route of x stations can be found by a min-cost flow problem with worst-case running time of $O(Cx^2)$.*

Proof. Following Lemma 3.1, one can easily show that the min-cost flow is a valid formulation for the problem at hand. In order to find the complexity of the algorithm, we divide the computation into two parts of constructing graph $\mathscr{G}$, and solving the min-cost flow problem on $\mathscr{G}$. For constructing the graph, the bottleneck is to find the interval edges. Since there are x stops in route $\mathcal{X}$, there are $\frac{x(x-1)}{2}$ possible combinations for pick-up and drop-off nodes for constructing the interval edges. Thus, the complexity of constructing graph $\mathscr{G}$ is $O(x^2)$. Also, note that graph $\mathscr{G}$ is an acyclic graph with integer capacities. According to Tarjan (1983), the min-cost flow in this graph has the time complexity of $O(|\mathscr{L}| + C\mathcal{S}(x, |\mathscr{L}|))$, where $\mathcal{S}(x, |\mathscr{L}|)$ is the complexity of finding the shortest path in a graph with x nodes and $|\mathscr{L}|$ edges. Using the Dijkstra's algorithm with Fibonacci heaps, it can be shown that the overall complexity of the min-cost flow in Algorithm 3.3 is $O(|\mathscr{L}| + C(|\mathscr{L} + x \log x|))$. In the most computationally-intensive scenario that arises when $|\mathscr{L}| = x^2$, the running time complexity of



(a) min-cost flow problem

(b) optimal flows with the optimal revenue of 46.5

Figure 3.6: The min-cost flow problem to find the most profitable set of passengers to be served through path $\mathcal{X} = \{(0, 3), (2, 4), (3, 2), (4, 1), (6, 3)\}$, and its optimal solution

the algorithm is $O(Cx^2)$, which is linear in C, and quadratic in x. □

It is worth mentioning that our min cost flow formulation, aiming to find the optimal set of passengers served by a fixed route, carries similarities with the min cost flow formulation in Jafari et al. (2016). However, as we have demonstrated in the Remark 3.1, our formulation differs from the min-cost flow formulation of Jafari et al. (2016) from two stand points. First, Our min-cost flow formulation does not depend on the number of passengers; rather, it depends on the number of origin-destination pairs. This results in significant computational savings in crowded metropolitan networks such as the Manhattan network. Second, our min-cost flow formulation has fewer number of edges, which again results in less computational effort.

3.4.1.2 shuttle Routing

Consider the problem of routing a single shuttle on road network G during time horizon $\mathcal{T}$. The route of a shuttle, denoted by $\mathcal{X}$, can be defined as a path in a time-expanded network G that connects a node with time 0 to a node with time T. Given a set of demand requests in Φ, the goal of this problem is to find a route that maximizes the revenue of served passengers. This problem is a well-known combinatorial problem that is NP-hard. In order to provide a high-quality solution to this problem, we propose a local search method in Algorithm 3.4. This local search algorithm is motivated by the concept of dynamic programming for finding the longest path in a weighted DAG. In this approach, the nodes will be observed in a topological order and the value of a node, which is defined as the longest path until that node, will be updated based on the value of its predecessor nodes and the weights of their edges connecting them to that node. After computing the value of all nodes, the longest path can be found by backtracking from the node with the highest value.

Similar to this approach, we consider a value at each node of $\mathcal{G}$ in our local search, denoted by $\varphi(n)$. This value can be defined as the highest revenue of routes that pass by node n. Since there are an exponential number of routes passing by each node, we propose an iterative framework that reduces the search space by fixing the nodes visited either prior to or after node n. We choose the route passing by the node with the highest value as the best route, denoted by $\mathcal{X}^*$. Let us define $\mathcal{F}_\alpha(n)$ as the partial route from a node in time 0 to node n at iteration α, hereafter referred to as the "predecessor route". Similarly, define $\mathcal{B}_\alpha(n)$ as the partial route from node n to a node in time T at iteration α, hereafter referred to as the "successor route". Note that connecting these two partial routes at node n yields a complete route, $\mathcal{X}$. Let us represent the connecting operator by $\oplus$.

The local search starts by constructing the predecessor route for all n in $\mathcal{N}$ at iteration 0 using Algorithm 3.5. In this algorithm we traverse the list of nodes in order of their time components, and for each node, pick one of its immediate predecessor nodes at random. This enables us to create a path from any node at time 0 to every node very quickly. In the next iteration, the predecessor

Algorithm 3.4: A local search for single shuttle routing

```
Input: Time-expanded network: $\mathcal{G}$
       Demand forecast: $\Phi$
Output: Local optimal shuttle route: $\mathcal{X}^*$
for $\pi = 1, 2, \cdots, \Pi$ do
    $\alpha \leftarrow 0$ ;
    $\Delta \leftarrow +\infty$ ;
    $\varphi_0^{*\mathcal{F}} \leftarrow 0$ ;
    $\mathcal{F}_0(n) \leftarrow$ Find an initial solution with $\mathcal{G}$ and $\Phi$ (see Algorithm 3.5);
    repeat
        $\alpha \leftarrow \alpha + 1$ ;
        $\varphi_\alpha^{*\mathcal{B}}, \mathcal{B}_\alpha(n) \leftarrow$ Solve backward local search with $\mathcal{G}$ and $\Phi$ and $\mathcal{F}_{\alpha-1}(n)$ (see Algorithm 3.6);
        $\varphi_\alpha^{*\mathcal{F}}, \mathcal{F}_\alpha(n), n^* \leftarrow$ Solve forward local search $\mathcal{G}$ and $\Phi$ and $\mathcal{B}_\alpha(n)$ (see Algorithm 3.7);
        $\Delta \leftarrow \frac{\varphi_\alpha^{*\mathcal{F}} - \varphi_{\alpha-1}^{*\mathcal{F}}}{\varphi_\alpha^{*\mathcal{F}}}$ ;
    until $\Delta < \zeta$;
    $\mathcal{X}^* \leftarrow \mathcal{F}_\alpha(n^*) \oplus \mathcal{B}_\alpha(n^*)$ ;
Retrieve the best solution of $\Pi$ replications ;
```

routes will be fixed from the previous iteration and Algorithm 3.6 finds the successor routes through a backward local search. This will be followed by a forward local search using Algorithm 3.7 that fixes the successor routes from the backward local search and updates the predecessor routes as well as the node with the highest value denoted by n^*. This process will continue until the relative gap between the highest node value of two consecutive forward local search is less than an improvement factor of $\zeta \in (0, 1)$. At convergence, one can retrieve the best route as the one that passes by the node n^*. Since the initial solution for routing is constructed randomly, we repeat the entire process Π times, and choose the solution that yields the highest node value at convergence. In what follows, we describe the backward and forward local search procedures in more detail.

In the backward local search, we assume that the predecessor route of each node is fixed from the previous iteration. We visit nodes in the reversed topological order of $\mathcal{G}$. For all nodes n with no outbound edges, we initialize the successor route to include only node n. At every node $n = (t, i)$ that has outbound edges, we consider all node v's that can be visited immediately from node n in graph $\mathcal{G}$. We create a complete route $\mathcal{X}$ by connecting the predecessor route of node n and the successor route of node v, and apply the min-cost algorithm to find the optimal revenue for fixed route $\mathcal{X}$ given demand forecast Φ. Upon completing this task for all v's, we choose the best v and denote it by v^*. As a result, the successor route of node n can be created by appending node n to the successor route of v^*. The whole process will be repeated until we reach the nodes with time 0 in $\mathcal{G}$. The highest node value will be stored in $\varphi^{*\mathcal{B}}$. The forward local search will be implemented next, with the similar general structure except the fact that the successor routes are fixed and the predecessor routes

Algorithm 3.5: An Initial solution for routing

```
Input: Time-expanded Network: $\mathcal{G}$
       Demand forecast: $\Phi$
Output: Predecessor route of node $n \in \mathcal{N}$: $\mathcal{F}_0(n)$
for $q = 0, 1, \ldots, T$ do
    for $j \in \mathcal{S}$ do
        $n \leftarrow (q, j)$ ;
        if $q = 0$ then
            $\mathcal{F}_0(n) \leftarrow [n]$;
            $\varphi_0(n) \leftarrow 0$;
        else
            $u = (t, i) \leftarrow$ draw randomly from $\delta^{-}(n)$ ;
            $\mathcal{F}_0(n) \leftarrow \mathcal{F}_0(u) \oplus [n]$;
            $\varphi_0(n) \leftarrow 0$;
```

Algorithm 3.6: Backward local search for routing

```
Input: Time-expanded Network: $\mathcal{G}$
       Route of shuttle until node $n$: $\mathcal{F}_{\alpha-1}(n)$
       Demand forecast: $\Phi$
Output: Maximum revenue of shuttle at iteration $\alpha$: $\varphi_\alpha^{*\mathcal{B}}$
        Successor route of node $n \in \mathcal{N}$: $\mathcal{B}_\alpha(n)$
$\varphi_\alpha^{*\mathcal{B}} \leftarrow 0$ ;
for $t = T, T-1, \ldots, 0$ do
    for $i \in \mathcal{S}$ do
        $n \leftarrow (t, i)$ ;
        if $\delta^{+}(n) = \emptyset$ then
            $\mathcal{B}_\alpha(n) \leftarrow [n]$;
            $\varphi_\alpha(n) \leftarrow 0$;
        else
            $\gamma^* \leftarrow -\infty$;
            for $v \in \delta^{+}(n)$ do
                $\mathcal{X} \leftarrow \mathcal{F}_{\alpha-1}(n) \oplus \mathcal{B}_\alpha(v)$;
                $\gamma \leftarrow$ Find the optimal revenue given route $\mathcal{X}$ and demand $\Phi$ (Algorithm 3.3);
                if $\gamma > \gamma^*$ then
                    $\gamma^* \leftarrow \gamma$;
                    $v^* \leftarrow v$;
            $\mathcal{B}_\alpha(n) \leftarrow [n] \oplus \mathcal{B}_\alpha(v^*)$;
            $\varphi_\alpha(n) \leftarrow \gamma^*$;
            if $\varphi_\alpha(n) > \varphi_\alpha^{*\mathcal{B}}$ then
                $\varphi_\alpha^{*\mathcal{B}} \leftarrow \varphi_\alpha(n)$ ;
```

Algorithm 3.7: Forward local search for routing

```
Input: Time-expanded Network: $\mathcal{G}$
       Route of  shuttle from node $n$: $\mathcal{B}_\alpha(n)$
       Demand forecast: $\Phi$
Output: Maximum revenue of shuttle at iteration $\alpha$: $\varphi_\alpha^{*\mathcal{F}}$
        Predecessor route of node $n \in \mathcal{N}$: $\mathcal{F}_\alpha(n)$
        Node with the highest revenue: $n^*$
$\varphi_\alpha^{*\mathcal{F}} \leftarrow 0$ ;
for $j \in \mathcal{S}$ do
    $n \leftarrow (q, j)$ ;
    if $\delta^-(n) = \emptyset$ then
        $\mathcal{F}_\alpha(n) \leftarrow [n]$;
        $\varphi_\alpha(n) \leftarrow 0$;
    else
        $\gamma^* \leftarrow -\infty$;
        for $u \in \delta^-(n)$ do
            $\mathcal{X} \leftarrow \mathcal{F}_\alpha(u) \oplus \mathcal{B}_\alpha(n)$;
            $\gamma \leftarrow$ Find optimal revenue given route $\mathcal{X}$ and demand requests in $\Phi$ (see Algorithm 3.3) ;
            if $\gamma > \gamma^*$ then
                $\gamma^* \leftarrow \gamma$;
                $u^* \leftarrow u$;
        $\mathcal{F}_\alpha(n) \leftarrow \mathcal{F}_\alpha(u^*) \oplus [n]$;
        $\varphi_\alpha(n) \leftarrow \gamma^*$;
        if $\varphi_\alpha(n) > \varphi_\alpha^{*\mathcal{F}}$ then
            $\varphi_\alpha^{*\mathcal{F}} \leftarrow \varphi_\alpha(n)$ ;
            $n^* \leftarrow n$ ;
```

will be updated in a topological order of nodes in $\mathcal{G}$. Now, let us demonstrate the backward and forward routing algorithms in the context of our toy example. For this example, we assume that all edges in the time-expanded network are available. Further suppose that we are currently at iteration $\alpha = 1$, and we have implemented the backward routing algorithm until node $(3, 2)$. Figure 3.7 shows the step in which we want to decide about the node that the shuttle visits immediately after (3,2). There are three choices since $\delta^+((3, 2)) = \{(4, 1), (4, 2), (4, 4)\}$. The edges corresponding to these 3 options are shown in solid black lines. Also, assume that we have found the successor routes of (4,1), (4,2), and (4,4) in earlier steps as: $\mathcal{B}_1((4, 1)) = \{(4, 1), (6, 3)\}$, $\mathcal{B}_1((4, 2)) = \{(4, 2), (5, 1), (6, 2)\}$ and $\mathcal{B}_1((4, 4)) = \{(4, 4), (5, 2), (6, 4)\}$. These three partial routes are shown in blue dashed lines. Finally assume that from the initial solution the predecessor route of (3,2) has been chosen as $\mathcal{F}_0(3, 2) = \{(0, 3), (2, 4), (3, 2)\}$. This partial route is shown with red dotted arrows. Using the connecting operator $\oplus$, we can construct three complete routes of $\mathcal{X}_1 = \{(0, 3), (2, 4), (3, 2), (4, 1), (6, 3)\}$, $\mathcal{X}_2 = \{(0, 3), (2, 4), (3, 2), (4, 2), (5, 1), (6, 2)\}$, and $\mathcal{X}_3 = \{(0, 3), (2, 4), (3, 2), (4, 4), (5, 2), (6, 4)\}$. For

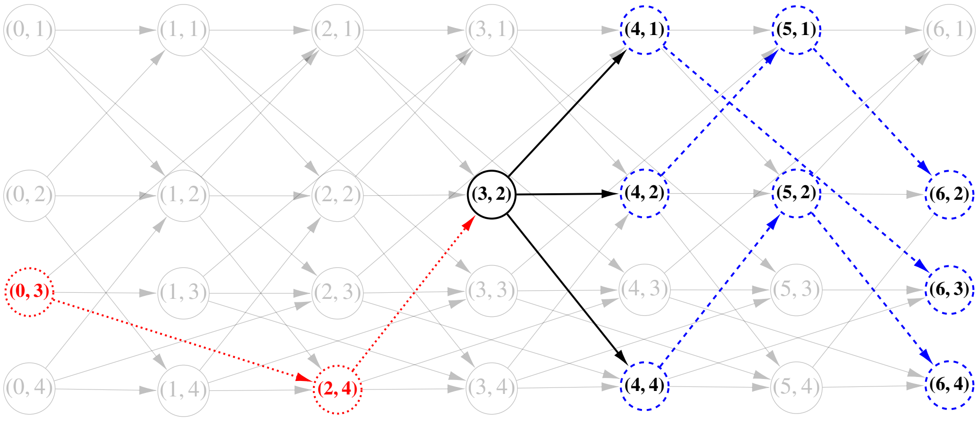


Figure 3.7: Backward local search for (3,2)

every complete route, we solve the min-cost flow problem in Algorithm 3.3. Suppose we obtain three values of 25.5, 27, and 31.5. Hence, we let $\varphi((3,2)) = 31.5$ and set $\mathcal{B}_1((3,2)) = \{(4,4),(5,2),(6,4)\}$. We follow the same procedure for the rest of the nodes in the reverse topological order. Next, we start from node (0,1) and traverse nodes in the forward direction by applying the forward routing algorithm. Let us consider again node (3,2), but this time in the forward routing algorithm as shown in Figure 3.8. Since $\delta^{-}((3,2)) = \{(2,1),(2,2),(2,4)\}$, we have three options for the node that the shuttle visits right before node (3,2). From earlier steps, assume that we have found $\mathcal{F}_1((2,1)) = \{(0,1),(1,2),(2,1)\}$, $\mathcal{F}_1((2,2)) = \{(0,2),(1,4),(2,2)\}$ and $\mathcal{F}_1((2,4)) = \{(0,3),(2,4)\}$. Also, we showed that from implementing the backward routing algorithm for (3,2), we got $\mathcal{B}_1((3,2)) = \{(3,2),(4,4),(5,2),(6,4)\}$. Using the connecting operator $\oplus$, we can construct three complete routes of $\mathcal{X}_1 = \{(0,1),(1,2),(2,1),(3,2),(4,4),(5,2),(6,4)\}$, $\mathcal{X}_2 = \{(0,2),(1,4),(2,2),(3,2),(4,4),(5,2),(6,4)\}$, and $\mathcal{X}_3 = \{(0,3),(2,4),(3,2),(4,4),(5,2),(6,4)\}$. For every complete route, we solve the min-cost flow problem in Algorithm 3.3. Suppose we obtain three values of 35, 37.5, and 31.5. Hence, we let $\varphi((3,2)) = 37.5$ and set $\mathcal{F}_1((3,2)) = \{(0,2),(1,4),(2,2),(3,2)\}$. We continue the whole process in Algorithm 3.4 until convergence.

Next, we present a lemma followed by a proposition that together prove the convergence of the local search algorithm.

Lemma 3.2. *At each step of the local search algorithm in 3.4, the objective function either improves or stays the same.*

Proof. At every move of the backward/Forward local search, we have the option to visit the same route or choose a node that results in a route with higher revenue. As a result the overall revenue either increases or stays the same at every move of the algorithm. □

Proposition 3.1. *Given an integer table of demand forecast, the objective, revenue, of the local search algorithm in 3.4 converges to a local optimum in polynomial time.*

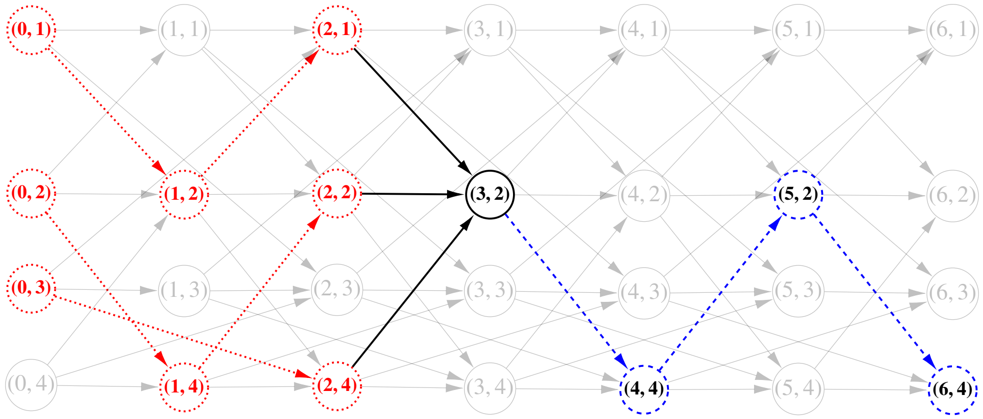


Figure 3.8: Forward local search for (3,2)

Proof. (1) There are possibly many, but finite number of ways to serve requests in table Φ. Therefore, the solution set is finite; (2) based on lemma 3.2, we infer that the solution always improves or stays the same, at which point we stop. Since the objective function is bounded from above, the algorithm is guaranteed to converge. From (1) and (2) the algorithm converges in finite number of iterations. In order to find a bound on the number of iterations before convergence, we assume that we iterate between the forward and backward local search only when $\varphi^{*\mathcal{F}}_{\alpha} - \varphi^{*\mathcal{F}}_{\alpha-1} \geq \zeta\ \varphi^{*\mathcal{F}}_{\alpha}$ where $\zeta \in (0, 1)$. Thus, for every two consecutive forward local searches, the inequality $\varphi^{*\mathcal{F}}_{\alpha-1} \leq (1-\zeta)\ \varphi^{*\mathcal{F}}_{\alpha}$ holds. If we start with the initial solution $\varphi^{*\mathcal{F}}_{0}$ and terminate with solution $\varphi^{*\mathcal{F}}_{n}$ after n iterations, then $\varphi^{*\mathcal{F}}_{0} \leq (1-\zeta)^n\ \varphi^{*\mathcal{F}}_{n} \leq (1-\zeta)^n\ \varphi^*$ where φ^* denotes the optimal node value. Since $1 + x \leq e^x$ for every $x \in \mathbb{R}$, we have $\varphi^{*\mathcal{F}}_{0} \leq e^{-\zeta n}\ \varphi^*$. This bounds the number of iterations n by $O(\frac{1}{\zeta}\log(\frac{\varphi^*}{\varphi^{*\mathcal{F}}_{0}}))$, which is polynomial. Moreover, at every forward/backward local search, we solve a min-cost flow problem per each edge of the time-expanded network. Thus, based on Proposition 3.1, the worst-case running time of a forward/backward local search is $O(|\mathcal{L}|\ Cx^2)$. Therefore, the entire local search algorithm in 3.4 has a polynomial time complexity of $O(\frac{1}{\zeta}\log(\frac{\varphi^*}{\varphi^{*\mathcal{F}}_{0}})\Pi\ |\mathcal{L}|\ Cx^2)$. Note that since φ^* is unknown, we can replace it with a valid upper bound $\bar{\varphi}$, which denotes the highest node value and can be calculated as $Cx\bar{\rho}$ where $\bar{\rho} = \max\limits_{\forall\,(t,o,d)} \rho$. □

In Algorithm 3.1, we introduce a procedure to find the base routes. This procedure iterates over some clustering steps until the total revenue, z, converges. We finalize this section by presenting a Proposition regarding the convergence of z.

Proposition 3.2. *If we initiate local search algorithms in 3.4 from previous solutions, the procedure in Algorithm 3.1 converges to a fixed value in a finite number of steps.*

Proof. We know that the total revenue is bounded from above. Therefore, it suffices to show that in every step of our procedure, the value of z improves or stays the same. Let us assume any initial partitioning of passengers among shuttles. Based on Lemma 3.1, we know that the revenue of each shuttle improves or stays the same. Therefore, after applying the local search to each partition, the total revenue either improves or stays the same. Also, the distance function 3.3 ensures that any trip that can be served by a shuttle has the distance of zero. Therefore, the clustering problem in 3.1 assigns those requests that were served by every shuttle in the previous clustering step to the same shuttle. Now, if we incorporate the previous route of every shuttle in the initial solution of the local search algorithm in 3.4, we can ensure that revenue for each shuttle does not decrease. Finally, in every re-clustering step, we only assign the unserved requests to the next closest shuttle, which guarantees again that the shuttle can achieve the same revenue by only serving the requests from the previous step in the worst case. As a result, the revenue of all shuttles continuously improves and the result follows. □

3.4.2 Online Routing

In this section, we introduce an online routing framework based on the results of the offline planning discussed in the previous section. Algorithm 3.8 provides a high-level description of the proposed method. Let us denote the simulation time by θ. Also let us introduce four tables of ψ_k, ψ^{ca}, ψ^{sc}_k, and ψ^{on}_k. Table ψ_k holds the expected future demand that can be served by shuttle k, and will be set to the average demand assigned to shuttle k in the offline phase, i.e., $\psi_k = \sum_{m=1}^{M} \Phi^k_m/M$. Note that ψ_k can be set to a high-quality demand forecast if one is available. Table ψ^{ca} holds the number of realized passengers that have arrived before time θ. The last two tables, ψ^{sc}_k and ψ^{on}_k, hold the number of scheduled and on-board passengers on shuttle k at time θ, respectively. The scheduled passengers are those whose demand has been realized and assigned to a shuttle in the past, but have not been picked up by the shuttle yet. The algorithm starts by initializing the route of each shuttle, denoted by $\mathcal{R}^k \ \forall\ k \in \mathcal{K}$, to the base route from the offline phase, i.e., $\mathcal{P}^k_1$. Also, we initialize the realized number of requests to 0. We set the simulation clock to 0 and increment it by 1 unit of time step until the end of the time horizon. At each simulation time, we repeat the following procedure.

We realize a set of passengers to arrive at the system during the interval $[\theta - 1, \theta]$ and update ψ^{ca}, accordingly. Similar to the offline phase, we adopt a multi-step clustering approach to distribute requests between shuttles. To this end, we define a proximity measure, denoted by $\chi^k(e, o, d)$, between trip (e, o, d) and shuttle k as follows:

$$\chi^k(e, o, d) = \frac{1}{M} \sum_{m=1}^{M} I(\text{route } \mathcal{P}^k_m \text{ can serve trip } (e, o, d)), \tag{3.5}$$

where $I(.)$ is an indicator function that takes the value 1 if the statement is true and 0 otherwise. This equation simply states that the proximity of a trip to a shuttle is equal to the probability of shuttle

Algorithm 3.8: Online framework

```
Input: Pool of shuttle routes: $\mathcal{P}$
       Average demand forecast for shuttle $k$: $\psi_k \quad \forall\, k \in \mathcal{K}$
Output: Optimal revenue of shuttle: $\mathcal{Z}$
$\mathcal{R}^k \leftarrow \mathcal{P}_1^k, \quad \forall\, k \in \mathcal{K}$;
$\psi^{ca}(e,o,d) \leftarrow 0 \quad \forall\, e \in \mathcal{T}, \forall\, o \in \mathcal{S}, \forall\, d \in \mathcal{S}$ ;
$\psi_k^{sc}(e,o,d) \leftarrow 0 \quad \forall\, k \in \mathcal{K}, \forall\, e \in \mathcal{T}, \forall\, o \in \mathcal{S}, \forall\, d \in \mathcal{S}$ ;
$\psi_k^{on}(e,o,d) \leftarrow 0 \quad \forall\, k \in \mathcal{K}, \forall\, e \in \mathcal{T}, \forall\, o \in \mathcal{S}, \forall\, d \in \mathcal{S}$ ;
for $\theta = 0, 1, \dots, T$ do
    $\psi_k(e,o,d) \leftarrow 0 \quad \forall\, k \in \mathcal{K}, \forall\, e \in \{0, \dots, \theta\}, \forall\, o \in \mathcal{S}, \forall\, d \in \mathcal{S}$ ;
    $\psi^{ca} \leftarrow$ Update realized requests based on passenger arrivals ;
    $\chi^k(e,o,d) \leftarrow$ Find the proximity between every trip $(e,o,d)$ in $\psi^{ca}$ and base route $\mathcal{P}_1^k$,
      $\forall\, k \in \mathcal{K}, \forall\, e \in \mathcal{T}, \forall\, o \in \mathcal{S}, \forall\, d \in \mathcal{S}$;
    Create the clustering problem:

        max   $\sum_{k=1}^{K} \sum_{(e,o,d)} \chi^k(e,o,d)\ \psi_k^{ca}(e,o,d)$                                                      (3.4a)
        s.t.  $\sum_{k=1}^{K} \psi_k^{ca}(e,o,d) = \psi^{ca}(e,o,d), \qquad \forall\, k \in \mathcal{K}, \forall\, e \in \mathcal{T}, \forall\, o \in \mathcal{S}, \forall\, d \in \mathcal{S}.$   (3.4b)

    for $r = 1, 2, \dots, R$ do
        $\psi_k^{ca} \leftarrow$ Distribute demand between shuttles by solving the clustering problem in 3.4;
        for $k \in \mathcal{K}$ do
            $n_y \leftarrow$ Find the the next node to be visited on $\mathcal{R}^k$ ;
            $\psi_k^{on}, \psi_k^{sc} \leftarrow$ Update on-board and scheduled passengers on shuttle $k$;
            $\gamma^* \leftarrow 0$;
            for $m = 1, 2, \dots, M$ do
                $n_{y'} \leftarrow$ Find the next node to be visited on $\mathcal{P}_m^k$ ;
                if $n_y = n_{y'}$ then
                    if Route $\mathcal{P}_m^k$ satisfies passengers in $\psi_k^{on}$ and $\psi_k^{sc}$ then
                        $\mathcal{X} \leftarrow$ Route $\mathcal{P}_m^k$ from $n_{y'}$;
                        $\gamma, f_k^{ca} \leftarrow$ Find optimal revenue given $\mathcal{X}$, $\psi_k$, $\psi_k^{ca}$, $\psi_k^{on}$, $\psi_k^{sc}$ (see Algorithm
                          3.9);
                        if $\gamma > \gamma^*$ then
                            $\gamma^* \leftarrow \gamma$; $m^* \leftarrow m$; $f_k^{ca^*} \leftarrow f_k^{ca}$;
            if $r = R$ then
                $\mathcal{R}^k \leftarrow$ Update route of  shuttle $k$ based on route $\mathcal{P}_{m^*}^k$ from $n_y$;
                Update pick-up and drop-off times of on-board and scheduled ;
            else
                Add $\psi_k^{ca}(e,o,d) \le f_k^{ca}(e,o,d)$ to the clustering problem in 3.4,    $\forall\, e \in \mathcal{T}$,
                  $\forall\, o \in \mathcal{S}, \forall\, d \in \mathcal{S}$;
                $\mathcal{Z} \leftarrow \mathcal{Z} + \gamma^*$;
```

k serving this request using different alternative routes. After finding the proximity between all realized trips and shuttles, we construct the clustering problem in 3.4. Next, we repeat the following procedure for R times.

We first obtain the candidate requests for each shuttle, denoted by ψ_k^{ca}, by solving the clustering problem in 3.4. For every shuttle k, we find the next stop on its route, denoted by n_y. Also, we update on-board and scheduled passengers from the previous simulation time step. Next, a cost-benefit analysis will be conducted (Algorithm 3.9) to find the set of realized passengers that can be served by this shuttle, taking into account the possibility of changing the shuttle's route at node n_y. This can be accomplished by finding all possible alternative routes for shuttle k that intersect at node n_k and satisfy the time windows and origin/destination stations of on-board and scheduled passengers, hereafter refereed to as the set of feasible alternatives. After finding the best feasible alternative, if this is the last clustering step ($r = R$), we change the route of the shuttle and update the time windows of on-board and scheduled passengers; Otherwise, we add a number of constraints to the problem in (3.4) to make sure the unserved requests will be assigned to the next closest shuttle.

We implement Algorithm 3.9 on the set of feasible alternatives to carry out a cost-benefit analysis. The result of this cost-benefit analysis determines the alternative route the shuttle follows. In this algorithm we first find the pick-up/drop-off nodes of on-board and scheduled passengers. Next, we construct a graph similar to the one in the offline phase, with a few changes. In the online graph, we create three copies of each node in route $\mathcal{X} = \{n_y, \ldots, n_i, \ldots, n_x\}$, denoted by $n_i^{\mathsf{d}}, n_i^{\mathsf{p}_1}$, and $n_i^{\mathsf{p}_2}$. The first copy accounts for the drop-off at node n_i, $\forall\ i:\ y \leq i \leq x$. The next two copies model the pick-up of the realized and predicted passengers at node n_i, respectively. In addition, we multiply the cost on edges of type $(n_i^{\mathsf{p}_2}, n_i^{\mathsf{d}})$ by a factor $\beta \in (0, 1)$ to take into account the fact that traversing these edges corresponds to serving demand that is only expected, and has not yet been realized. Finally, we alter the demand of origin and destination nodes of on-board and scheduled customers to ensure that these customers will be picked up by the min-cost flow algorithm. More precisely, we add the number of on-board/scheduled customers to the demand of their origin nodes and subtract it from the demand of their destination nodes.

Let us demonstrate a small instance of this cost-benefit analysis using our toy example. Suppose that we have found two alternative routes of $\mathcal{P}_1^1 = \{(0,2),(1,1),(2,2),(3,1),(4,2),(5,4),(6,2)\}$ and $\mathcal{P}_2^1 = \{(0,2),(1,4),(2,2),(3,4),(4,2),(5,1),(6,2)\}$ for our shuttle in the offline phase. These routes are shown in Figure 3.9 with dashed and solid lines, respectively. Suppose that the current simulation time is $\theta = 2$, and the current route of the shuttle is $\mathcal{R} = \mathcal{P}_1^1 = \{(0,2),(1,1),(2,2),(3,1),$ $(4,2),(5,1),(6,2)$. Suppose that we want to evaluate the revenue of the shuttle by changing the route to $\mathcal{P}_2^1$ at node $(2,2)$. Assume that the shuttle has already picked up 2 customers at time 1 from station 1, whose destination station is 4. Also, 7 customers are scheduled to be picked up at time 2 from station 2, where 5 of them want to go to station 4 and the rest want to go to station 1.

Also, at this simulation time, $\theta = 2$, the system has received 10 trip requests at station 2, of which 6 go to station 4 and the rest go to station 1, and 5 trip requests at station 4 that go station 2.

Algorithm 3.9: Find optimal revenue in real-time

```
Input: Fixed Route of shuttle: $\mathcal{X} = [n_y, n_{y+1}, n_{y+2}, \ldots, n_x]$
       Demand forecast: $\psi_k$
       Realized demand: $\psi_k^{ca}$
       On-board passengers count: $\psi_k^{on}$
       Scheduled passengers count: $\psi_k^{sc}$
Output: Optimal revenue of shuttle: $\gamma$
        Optimal number of served passengers from realized demand: $f^{ca}$
pick-up times of on-board passengers $\leftarrow n_y^{\mathsf{p}_1}$ ;
Find drop-off times of on-board passengers based on route $\mathcal{X}$ ;
Find pick-up and drop-off times of scheduled passengers based on route $\mathcal{X}$ ;
$\mathcal{N} \leftarrow \{n_y^{\mathsf{d}}, n_y^{\mathsf{p}_1}, n_y^{\mathsf{p}_2}, n_{y+1}^{\mathsf{d}}, n_{y+1}^{\mathsf{p}_1}, n_{y+1}^{\mathsf{p}_2}, \ldots, n_x^{\mathsf{d}}, n_x^{\mathsf{p}_1}, n_x^{\mathsf{p}_2}\}$ ;
$\mathcal{D}(n_y^{\mathsf{d}}) \leftarrow -C;\ \mathcal{D}(n_x^{\mathsf{p}}) \leftarrow C;\ \mathcal{D}(n_i^{\mathsf{d}}) \leftarrow 0\ \forall\ i \in \{y+1, \ldots, x\};\ \mathcal{D}(n_i^{\mathsf{p}}) \leftarrow 0\ \forall\ i \in \{y, \ldots, x-1\}$;
$\mathcal{L} \leftarrow \{\}$ ;
for $i = y, y+1, \ldots, x$ do
    $\mathcal{L} \leftarrow \mathcal{L} \cup \left\{n_i^{\mathsf{d}}, n_i^{\mathsf{p}_1}\right\};\ \mathcal{W}(n_i^{\mathsf{d}}, n_i^{\mathsf{p}_1}) \leftarrow 0;\ \mathcal{U}(n_i^{\mathsf{d}}, n_i^{\mathsf{p}_1}) \leftarrow C$ ;
    $\mathcal{L} \leftarrow \mathcal{L} \cup \left\{n_i^{\mathsf{p}_1}, n_i^{\mathsf{p}_2}\right\};\ \mathcal{W}(n_i^{\mathsf{p}_1}, n_i^{\mathsf{p}_2}) \leftarrow 0;\ \mathcal{U}(n_i^{\mathsf{p}_1}, n_i^{\mathsf{p}_2}) \leftarrow C$ ;
    $\mathcal{L} \leftarrow \mathcal{L} \cup \left\{n_i^{\mathsf{d}}, n_{i+1}^{\mathsf{d}}\right\};\ \mathcal{W}(n_i^{\mathsf{d}}, n_{i+1}^{\mathsf{d}}) \leftarrow 0;\ \mathcal{U}(n_i^{\mathsf{d}}, n_{i+1}^{\mathsf{d}}) \leftarrow C$ ;
for $i = y, y+1, \ldots, x-1$ do
    for $j = i+1, i+2, \ldots, x$ do
        $(p, o) \leftarrow n_i^{\mathsf{p}_1}$ ; $(l, d) \leftarrow n_j^{\mathsf{d}}$ ;
        if $d$ not visited again between $n_i^{\mathsf{p}}$ and $n_j^{\mathsf{d}}$ then
            if $(l - p) \leq \tau(p, o, d) + \Omega$ then
                $\mathcal{L} \leftarrow \mathcal{L} \cup \left\{n_i^{\mathsf{p}_1}, n_j^{\mathsf{d}}) \cup (n_i^{\mathsf{p}_2}, n_j^{\mathsf{d}}\right\}$ ;
                $\mathcal{D}(n_i^{\mathsf{d}}) \leftarrow \mathcal{D}(n_i^{\mathsf{d}}) + \psi_k^{sc}(p, o, d) + \psi_k^{on}(p, o, d)$ ;
                $\mathcal{D}(n_j^{\mathsf{d}}) \leftarrow \mathcal{D}(n_j^{\mathsf{d}}) - \psi_k^{sc}(p, o, d) + \psi_k^{on}(p, o, d)$ ;
                if $o$ visited again between $n_i^{\mathsf{p}}$ and $n_j^{\mathsf{d}}$ then
                    $\mathcal{U}(n_i^{\mathsf{p}_1}, n_j^{\mathsf{d}}) \leftarrow \psi_k^{ca}(p, o, d)$ ;
                    $\mathcal{U}(n_i^{\mathsf{p}_2}, n_j^{\mathsf{d}}) \leftarrow \psi_k(p, o, d)$ ;
                else
                    $t_0 \leftarrow$ Last time $o$ visited in time window $[p - \omega, p]$ ;
                    $\mathcal{U}(n_i^{\mathsf{p}_1}, n_j^{\mathsf{d}}) \leftarrow \sum_{t=t_0+1}^{\mathsf{p}} \psi_k^{ca}(t, o, d)$ ;
                    $\mathcal{U}(n_i^{\mathsf{p}_2}, n_j^{\mathsf{d}}) \leftarrow \sum_{t=t_0+1}^{\mathsf{p}} \psi_k(t, o, d)$ ;
                $\mathcal{W}(n_i^{\mathsf{p}_1}, n_j^{\mathsf{d}}) \leftarrow -\rho(p, o, d);\ \mathcal{W}(n_i^{\mathsf{p}_2}, n_j^{\mathsf{d}}) \leftarrow -\rho(p, o, d) \times \beta$ ;
Define graph $\mathcal{G} = (\mathcal{N}, \mathcal{L}, \mathcal{D}, \mathcal{W}, \mathcal{U})$ ;
Solve min-cost flow on $\mathcal{G}$ ;
$f^{ca} \leftarrow$ Optimal flow on edges of type $(n_i^{\mathsf{p}_1}, n_j^{\mathsf{d}})$ ;
$\gamma \leftarrow -1\times$ Optimal flow cost ;
```

This information is summarized in Table 3.4. In Table 3.4, the on-board and scheduled customers are specified in red (the first number in each cell) and candidate customers are specified in black

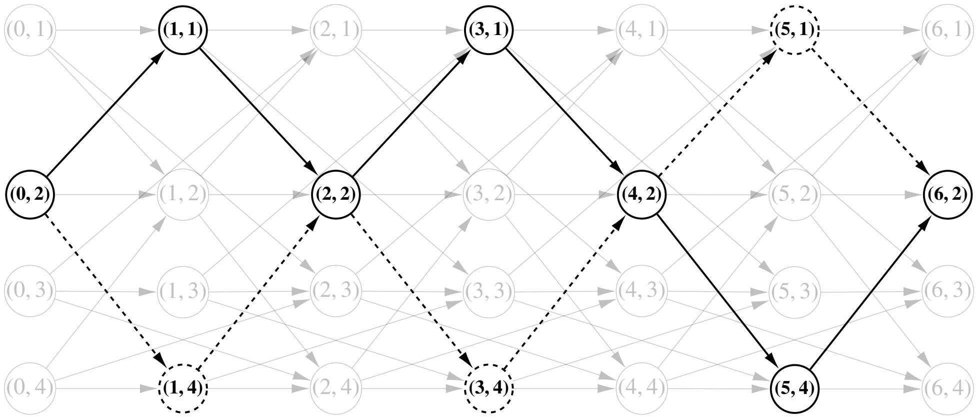


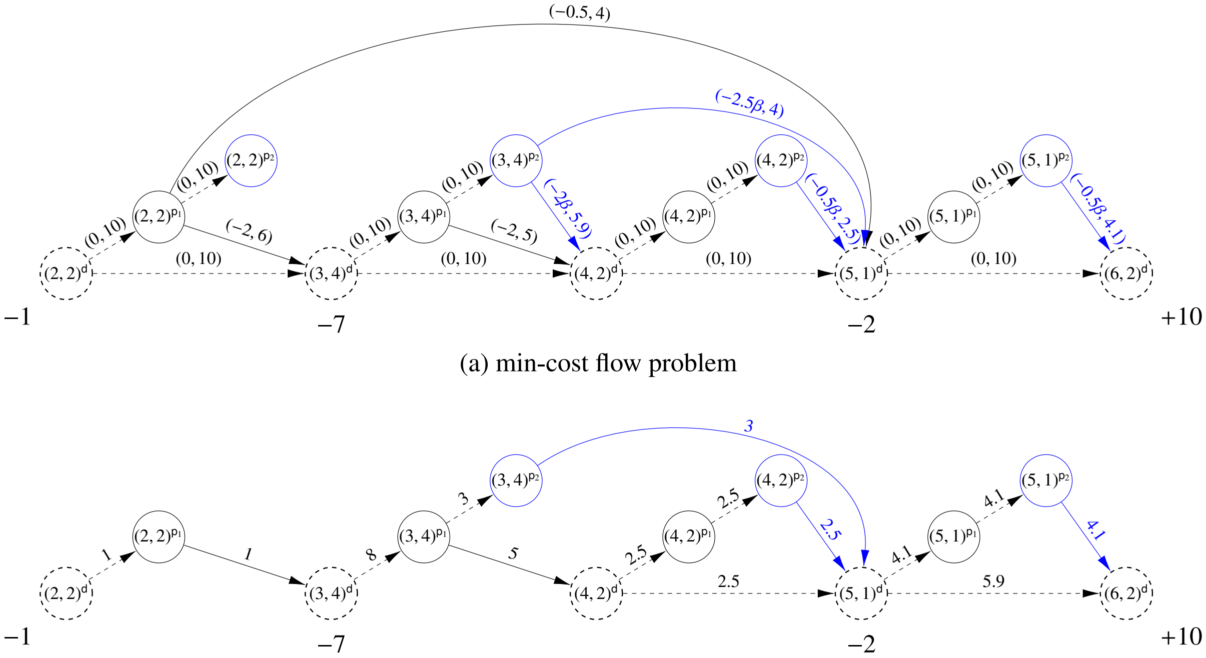
Figure 3.9: The current and alternative route for the shuttle

(the second number in each cell). Blue numbers (the third number in each cell) indicate the expected number of requests based on historical data.

We can model this online problem as a min-cost flow problem as described in Algorithm 3.9 and shown in Figure 3.9. As such, we define 3 distinct nodes in (2,2) for drop-off (black dashed circle), candidates' pick-up (black solid circle), and expected customers' pick-up (blue solid circle). Similarly, we define 3 distinct nodes in (3,4), (4,2), and (5,1). Finally, we define a single drop-off circle for node (6,2). Similar to the offline phase, the clique edges are shown with dashed lines and the interval edges with solid lines. However, there is a difference in the online phase when specifying the cost of interval edges. For expected customers, we multiply the original cost by a discount factor, β, to account for the fact that these requests are only expected and have not been realized yet. In addition, we initially set the demand of the drop-off nodes at (2,2), (3,4), (4,2), (5,1), and (6,2) to -10, 0, 0, 0, and 10, respectively. Next, we alter these values to make sure that the optimal flow satisfies the on-board and scheduled customers. As such, we add 9 units to the demand of the drop-off node at (2,2), because all the on-board and scheduled customers start from this node. Next, we subtract 7 units from the demand of the drop-off node at (3,4) and , since 7 customers will be dropped off at this node. Finally, we subtract 2 units from the demand of the drop-off node at

Table 3.4: The number of on-board/scheduled customers (in red), realized customers (in black), and expected number of requests (in blue) for path $\mathcal{P}_2^1$ at $\theta = 2$.

	(2,2)	(3,4)	(4,2)	(5,1)	(6,2)
(0,2)	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0
(1,1)	0, 0, 0	2, 0, 0	0, 0, 0	0, 0, 0	0, 0, 0
(2,2)		5, 6, 0	0, 0, 0	2, 4, 0	0, 0, 0
(3,4)			0, 5, 5.9	0, 0, 5.9	0, 0, 0
(4,2)				0, 0, 2.5	0, 0, 0
(5,1)					0, 0, 4.1



(a) min-cost flow problem

(b) optimal flows with the optimal revenue of 17.4 given $\beta = 0.5$

Figure 3.10: Min-cost flow for path $\mathcal{P}_2^1$ at $\theta = 2$ and its optimal solution. The black dashed circles model drop-offs, the black solid circles model candidate customers' pick-ups, and the blue solid circles model the expected pick-ups.

(5,1) where the rest of customers will be dropped off. As a result, the demand of the drop-off nodes at (2,2), (3,4), (4,2), (5,1), and (6,2) are set to −1, −7, 0, −2, and +10, respectively. As shown in Figure 3.10(b), solving the min-cost flow problem in this graph yields an optimal revenue of 17.4 given $\beta = 0.5$. If we follow the same procedure for the original route, $\mathcal{P}_1^1$, we obtain an optimal revenue of 14.5 for the same value of β. Therefore, our analysis suggests that the shuttle is better off by changing its route from $\mathcal{P}_1^1$ to $\mathcal{P}_2^1$ for the rest of its journey.

3.5 Numerical Experiment

In this section, we showcase the performance of our framework under two different case studies. In the first case study, we consider a simulated ridesharing dataset over a well-known small transportation network, namely the Nguyen-Dupuis network. We further conduct an extensive sensitivity analysis on different parameters in both the offline and the online phases of the framework. Next, in order to showcase the scalability of our framework, we evaluate the performance of the proposed method in serving the taxi trips in the Manhattan area using the 2016 New York Taxi dataset.

In all case studies, we compare the performance of our algorithm under three different configurations with two well-known state-of-the-art models proposed by Ma et al. (2013) and Alonso-Mora et al. (2017), which are modified based on the assumptions stated in Section 3.3. In the first configuration, we consider a full version of our method with each shuttle having 50 alternative routes. We use the average rate of demand ϕ, obtained from historical data, as our demand prediction, with the influence factor of β.

In the second configuration, we still use alternative routes for shuttles, but we assume that there is no future demand forecast, i.e., $\beta = 0$. Finally, in the third configuration, we assume that neither the alternative routes nor the demand forecast is available. This configuration leads to a system that resembles fixed-route transit. Hereafter, we refer to these three configurations as the "Full", "No Pred.", and "No Alter." methods, respectively. The T-share insertion method proposed by Ma et al. (2013) and the dynamic trip-shuttle assignment model proposed by Alonso-Mora et al. (2017), which are hereinafter referred to as "Insertion" and "Assignment", respectively, serve as benchmarks for performance evaluation. In order to evaluate the performance of these methods, we introduce the following measures:

- $\mathcal{Z}$: Total revenue of all shuttles (in miles)
- $\mathcal{M}$: Matching rate of customers served by all shuttles
- $\mathcal{W}$: Waiting time of served customers averaged over all shuttle and customers (in minutes)
- $\mathcal{D}$: Detour time of served customers averaged over all shuttle and customers (in minutes)
- $\mathcal{U}$: Utilization rate of shuttles' seats averaged over all shuttle and time steps
- $\mathcal{A}$: Number of times shuttles switch their routes averaged over all shuttle (only for Full and No Pred. methods)
- Θ: Average computation time (in milliseconds) of online framework over all time steps

In the following subsections, we denote the online performance measures corresponding to Full, No Pred., No Alter., Insertion and Assignment methods by indices 1, 2, 3, 4, and 5, respectively. We also show the value of the parameter β in the full configuration by a superscript for these measures. For instance, $\mathcal{Z}_1^{0.5}$ denotes the total revenue of shuttles using the full configuration of our proposed method with $\beta = 0.5$.

All experiments are conducted on a 56-core 3.50 GHz Intel Xeon machine with a 64-bit version of the Windows 10 operating system with 256.0 GB of RAM. All methods are coded in `Python 3.7` and min-cost flow problems are solved using the `OR-Tools` package. Next, we present the experiment setups and the results of various experiments for the two case studies.

3.5.1 Nguyen-Dupuis Case Study

The Nguyen-Dupuis network consists of 13 nodes, 38 links, and 13 OD pairs as shown in Figure 3.11. The italic number on every edge shows the constant link travel time in minutes. For this network, we assume a constant speed of 15 miles per hour on every link. Hence, the distance in miles between every two link can be found as $\rho(t, o, d) = 0.25 \times \tau(t, o, d),\ \forall\ t \in \mathcal{T}, \forall\ o \in \mathcal{S}, \forall\ d \in \mathcal{S}$.

We further assume that a fleet of 50 homogeneous shuttles with capacity 10 is available. For large-scale implementation, we assume a study time horizon of 60 minutes, with time steps of 1 minute. We further assume that from historical data, the arrival of demand requests for each time step and OD pair follows a Poisson distribution with average rate being stored in ϕ presented in Table 3.5. Note that we assume that ϕ only contains the average count of passengers that arrived at the system and could be served during the given time horizon. The expected total number of trips is assumed to be 1000 and the arrival time of a request and its origin and destination are determined based on a uniform random distribution. We also set the waiting time ω and the detour budget time Ω to 5 and 10 minutes, respectively.

3.5.1.1 Results

For generating base and alternative routes of each shuttle, we implemented the offline framework in Algorithm 3.1 with $M = 50$, $R = 5$, and $\epsilon = 10$ minutes. In other words, after finding the base routes, the time-expanded network for each shuttle was reduced to a sub-graph whose nodes are at most 10

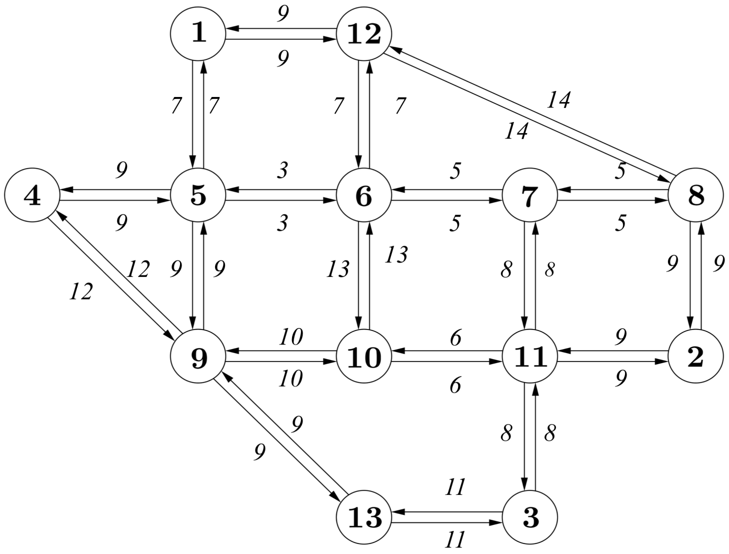


Figure 3.11: Nguyen-Dupuis network

Table 3.5: Hourly rate of demand between all pairs of origin-destination in the Nguyen-Dupuis case study

O\D	1	2	3	4	5	6	7	8	9	10	11	12	13
1	0	5	22	2	5	0	0	0	0	0	0	12	0
2	12	0	13	1	2	0	0	15	0	0	25	0	0
3	22	4	0	15	4	0	0	0	0	0	18	0	5
4	14	7	11	0	34	10	0	0	22	0	0	0	0
5	1	10	10	2	0	0	0	0	8	0	0	0	0
6	5	3	13	8	12	0	6	31	7	15	30	8	0
7	17	1	12	22	9	18	0	16	2	4	10	9	2
8	0	10	4	7	0	0	9	0	12	4	9	13	4
9	11	1	4	12	0	0	15	0	0	27	9	0	4
10	0	0	0	2	0	14	3	0	30	0	9	0	17
11	0	5	6	2	0	12	13	0	20	0	0	0	28
12	5	0	0	10	5	5	6	9	0	31	0	0	9
13	0	0	15	0	0	0	0	0	10	8	0	0	0

minutes away from the closest node on its base route. Next, we implemented demand clustering and applied Algorithm 3.4 (for five iterations) in the reduced graphs to obtain 49 alternative routes per base route using different realizations of a Poisson distribution with mean ϕ. We set the number of replications, Π, and improvement factor, ζ, in the local search algorithm to 10 and 0.01, respectively. Thus, for finding each route, we continue the local search until the relative gap is less than 0.01, and we repeat the entire process with 10 initial solutions and pick the one that yields the highest revenue. Figure 3.12 shows the convergence of the total revenue for the base routes, z, in the offline phase as stated in Proposition 3.2. This figure clearly shows that the total revenue increase at every step of the clustering and re-clustering approach outlined in Algorithm 3.1 until it converges to a fixed value.

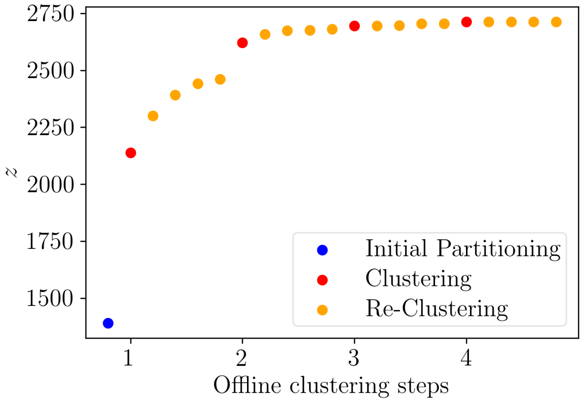


Figure 3.12: Convergence of the total revenue for the base routes in the offline phase of the Nguyen Dupuis case study

In the online phase, we consider 3 configurations of our proposed method and compare their results with the two benchmark methods. In these experiments, we define two additional parameters, μ and σ, to generate scenarios where the actual demand is a shifted (by μ) or scaled (by σ) version of values in ϕ. For instance, a scenario with $\mu = 0$, $\sigma = 5$, means that we simulate customer arrivals based on a Poisson distribution with the rate of $\phi' \sim N(\phi + 0, 5)$.

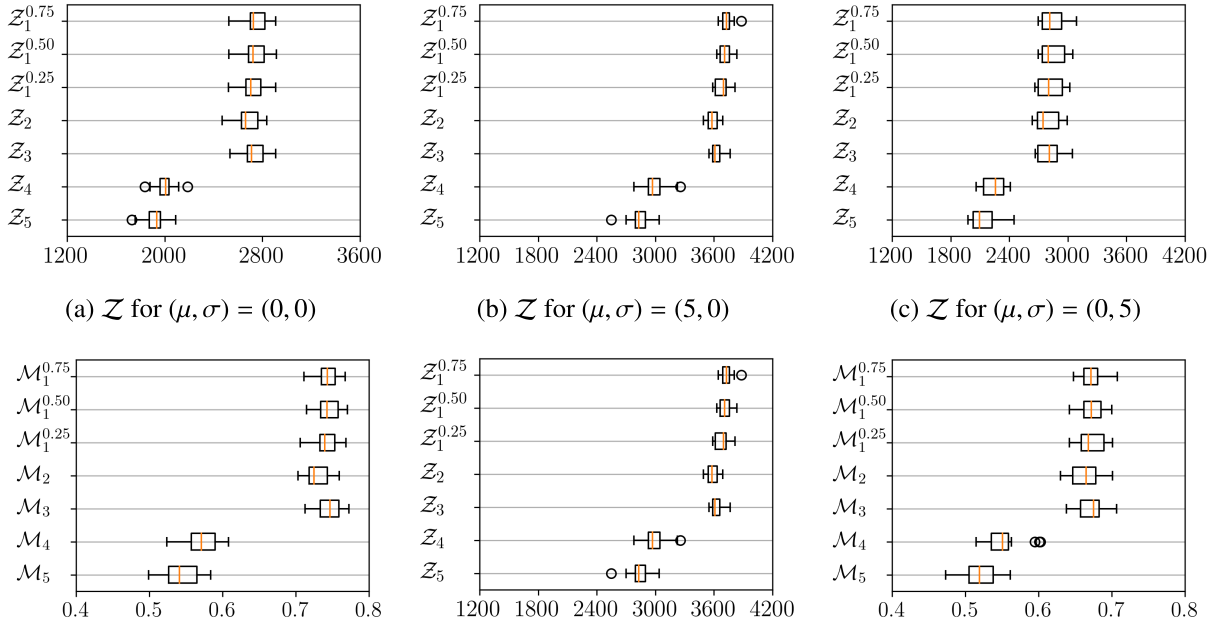
The box plots in Figure 3.13 demonstrate the revenues and matching rates of experiments for different methods under the three demand scenarios. Also, Table 3.6 summarizes the average revenue and matching rate over 20 simulation runs. In order to evaluate the significance of differences in the values of this table, we conducted paired t-test between all pairwise combinations of methods under the three scenarios, the p-value of which are summarized in Table B.14. Table 3.6 shows that our proposed method significantly outperforms both benchmark methods. This table also suggests that in cases where the expected demand is close to the realized one, it is more appropriate to stay on the base route, as $\mathcal{Z}_3$ yields high revenue in the first scenario of μ and σ. However, as the mean or variance of the realized demand deviates from the expected demand, we may benefit from switching to alternative routes.

In this table, the best value of β is embolden for each scenario. Although the highest value of β in all scenarios has been shown to provide the highest average revenue, the p-values in Table B.14 indicate that the right choice of its value depends on the performance of our demand forecast. More specifically, high values of β provide higher revenues when the forecast is accurate (scenario 1). However, as the mean/variance of demand forecast deviates further from the mean/variance of the realized demand, lower values of β become comparable or even preferable.

This table also suggests that the result of the Insertion method surprisingly outperforms that of the Assignment method, even through the latter considers all combinations of trip sharing and assigns the shuttles to the best combinations optimally at any time step. The reason behind this observation resides in the fact that assigning customers to shuttles optimally without considering the future unobserved demand may increase the level of nearsightedness of a method to the point that a FCFS insertion method can outperform it. Overall, these experiments suggests that the Full configuration of our method is the most robust strategy in a stochastic and dynamic environment although the margin of its outperformance start to shrink by deviating from a perfect demand forecast.

3.5.1.2 Sensitivity Analysis

In this section, we study the impact of ϵ by repeating the experiments with three values of 5, 10, and 15 minutes for this parameter. Tables 3.7, 3.8, B.2, B.3, B.4, B.5, and B.6 summarize the impact of changing ϵ on the performance measures averaged over 20 simulation runs. Also, Table B.16 contains the p-value of the pairwise comparisons between all combinations of ϵ for all measures, methods, and demand scenarios.



(a) $\mathcal{Z}$ for $(\mu, \sigma) = (0, 0)$

(b) $\mathcal{Z}$ for $(\mu, \sigma) = (5, 0)$

(c) $\mathcal{Z}$ for $(\mu, \sigma) = (0, 5)$

(d) $\mathcal{M}$ for $(\mu, \sigma) = (0, 0)$

(e) $\mathcal{M}$ for $(\mu, \sigma) = (5, 0)$

(f) $\mathcal{M}$ for $(\mu, \sigma) = (0, 5)$

Figure 3.13: The result of different methods on the performance measures in the Nguyen-Dupuis case study

Table 3.6: The results of different methods on the performance measures averaged over simulation runs in the Nguyen-Dupuis case study

μ	σ	β	**Full**		**No Pred.**		**No Alter.**		**Insertion**		**Assignment**	
			$\overline{\mathcal{Z}}_1$	$\overline{\mathcal{M}}_1$	$\overline{\mathcal{Z}}_2$	$\overline{\mathcal{M}}_2$	$\overline{\mathcal{Z}}_3$	$\overline{\mathcal{M}}_3$	$\overline{\mathcal{Z}}_4$	$\overline{\mathcal{M}}_4$	$\overline{\mathcal{Z}}_5$	$\overline{\mathcal{M}}_5$
0	0	0.95	2748.90	0.74	2677.08	0.73	2731.89	0.75	2003.29	0.57	1914.25	0.54
		0.50	2741.19	0.74								
		0.25	2725.04	0.74								
5	0	0.95	3729.16	0.56	3583.72	0.55	3629.71	0.56	2997.8	0.47	2835.92	0.44
		0.50	3713.06	0.56								
		0.25	3677.31	0.57								
0	5	0.95	2847.92	0.67	2785.86	0.66	2804.75	0.67	2242.68	0.55	2137.66	0.52
		0.50	2842.04	0.67								
		0.25	2823.32	0.67								

Table 3.7 suggests that for a given number of alternative routes, as ϵ increases, the total revenue of the Full method first increases and then it starts to decrease. This is due to the fact that although increasing ϵ allows the alternatives of a shuttle to cover a larger region, this happens at the price of having smaller number of similar spatio-temporal nodes between alternative routes. Thus, the chance of serving on-board or scheduled customers decreases as we increase the value of ϵ. Also, decreasing ϵ to a very small value prevents the shuttle to explore different regions, and hence, results in alternatives serving similar customers. It is worth mentioning that Table B.16 does not show a statistically significant difference between different values of epsilon for scenarios 1 and 3.

Table 3.7: Impact of the spatio-temporal threshold, ϵ, on the average revenue, $\mathcal{Z}$, in the Nguyen-Dupuis case study

μ	σ	β	**Full**			**No Pred.**			**No Alter.**	**Insertion**	**Assignment**
			$\epsilon = 5$	$\epsilon = 10$	$\epsilon = 15$	$\epsilon = 5$	$\epsilon = 10$	$\epsilon = 15$			
0	0	0.95	2745.39	2748.90	2740.58	2687.34	2677.08	2690.22	2732.71	2003.29	1905.72
		0.50	2736.75	2741.19	2733.26						
		0.25	2712.49	2725.04	2708.52						
5	0	0.95	3701.12	3729.16	3720.26	3592.11	3583.72	3554.56	3628.69	2997.80	2851.28
		0.50	3676.54	3713.06	3694.45						
		0.25	3626.26	3677.31	3638.58						
0	5	0.95	2838.06	2847.92	2840.58	2765.89	2785.86	2760.79	2807.85	2242.68	2138.80
		0.50	2836.90	2842.04	2828.25						
		0.25	2810.98	2823.32	2801.10						

Table 3.8: Impact of the spatio-temporal threshold, ϵ, on the average matching rate, $\mathcal{M}$, in the Nguyen-Dupuis case study

μ	σ	β	**Full**			**No Pred.**			**No Alter.**	**Insertion**	**Assignment**
			$\epsilon = 5$	$\epsilon = 10$	$\epsilon = 15$	$\epsilon = 5$	$\epsilon = 10$	$\epsilon = 15$			
0	0	0.95	0.74	0.74	0.74	0.73	0.73	0.73	0.75	0.57	0.54
		0.50	0.74	0.74	0.74						
		0.25	0.74	0.74	0.74						
5	0	0.95	0.56	0.56	0.56	0.55	0.55	0.55	0.56	0.47	0.44
		0.50	0.56	0.56	0.56						
		0.25	0.56	0.57	0.56						
0	5	0.95	0.67	0.67	0.67	0.66	0.66	0.66	0.67	0.55	0.52
		0.50	0.67	0.67	0.67						
		0.25	0.67	0.67	0.67						

Interestingly, Table B.16 shows a different trend for the No Pred. method. More specifically, it indicates that the lower the value of ϵ, the higher the revenue. This is mainly due to the fact that using alternative routes without predicted future demand causes the online cost-benefit analysis fail to find the best choice for the remaining part of a shuttle's trajectory. Overall, smaller choice of ϵ is preferred when we have access to a high-quality demand forecast. However, note that we may need to consider larger values of ϵ when the mean or variance of realized demand differs from what we predicted. Table 3.8 indicates that ϵ does not seem to have a major impact on matching rates of the Full and No Pred. methods.

The impact of ϵ on the rest of measures has been presented in Appendix B.2. In the online experiments above, we assumed that customer arrivals followed a Poisson distribution with rate ϕ. In order to see the impact of arrival distribution, we repeat the experiments by assuming a Uniform distribution for the arrival of customers over time, and compare its impact on different measures with those of Poisson distribution. The results of this analysis can be found in Appendix B.3.

3.5.2 Manhattan Case Study

In order to showcase the scalability of the framework, we conduct an experiment in the Manhattan area in New York City (NYC) using the NYC Taxi dataset. The dataset contains the longitude and latitude of trip origins and destinations and the pick-up times for all trips. We define 184 stations that are strategically positioned in the Manhattan area to ensure there is at least one station within the walking distance of a typical trip origin/destination (see Figure 3.14). The shortest-path driving distances and travel times between all stations are obtained from the Google API.

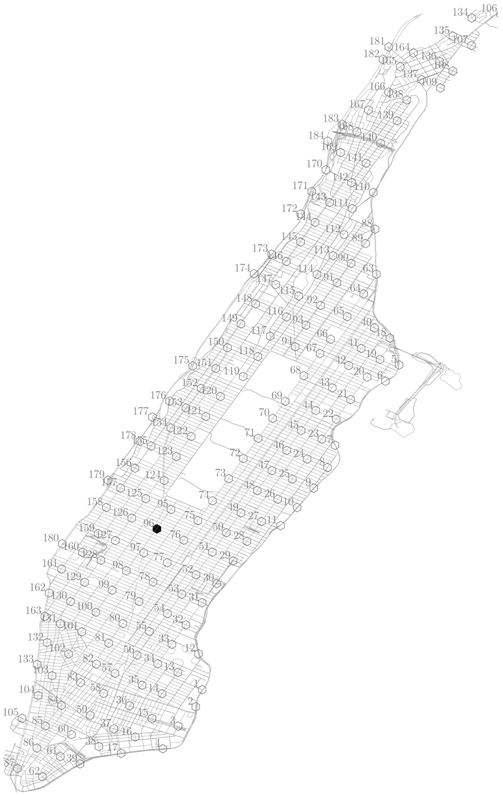


Figure 3.14: Road network of Manhattan, NY

In this case study, we assume that the ride-sourcing company possesses a fleet of 100 shuttles with capacity of 10 that operate during the evening pick hours in the Manhattan area. All shuttles start their trips from a station in the center of the region (the black station in Figure 3.14) at 18:00, and end their trips in the same station at 21:00. As such, we obtain a historical demand table ϕ by taking the average of trip counts for every minute and origin/destination stations from July 2015 to May 2016. We further reserve the demand of June 2016 for running simulations in the online framework. In the offline phase, we construct 50 alternative routes for each shuttle by assuming a Poisson distribution for the arrival of customers with rate of ϕ, and setting the parameters ϵ, R, Π, and ζ to 15 minutes, 5, 100, and 0.01, respectively.

3.5.2.1 Results

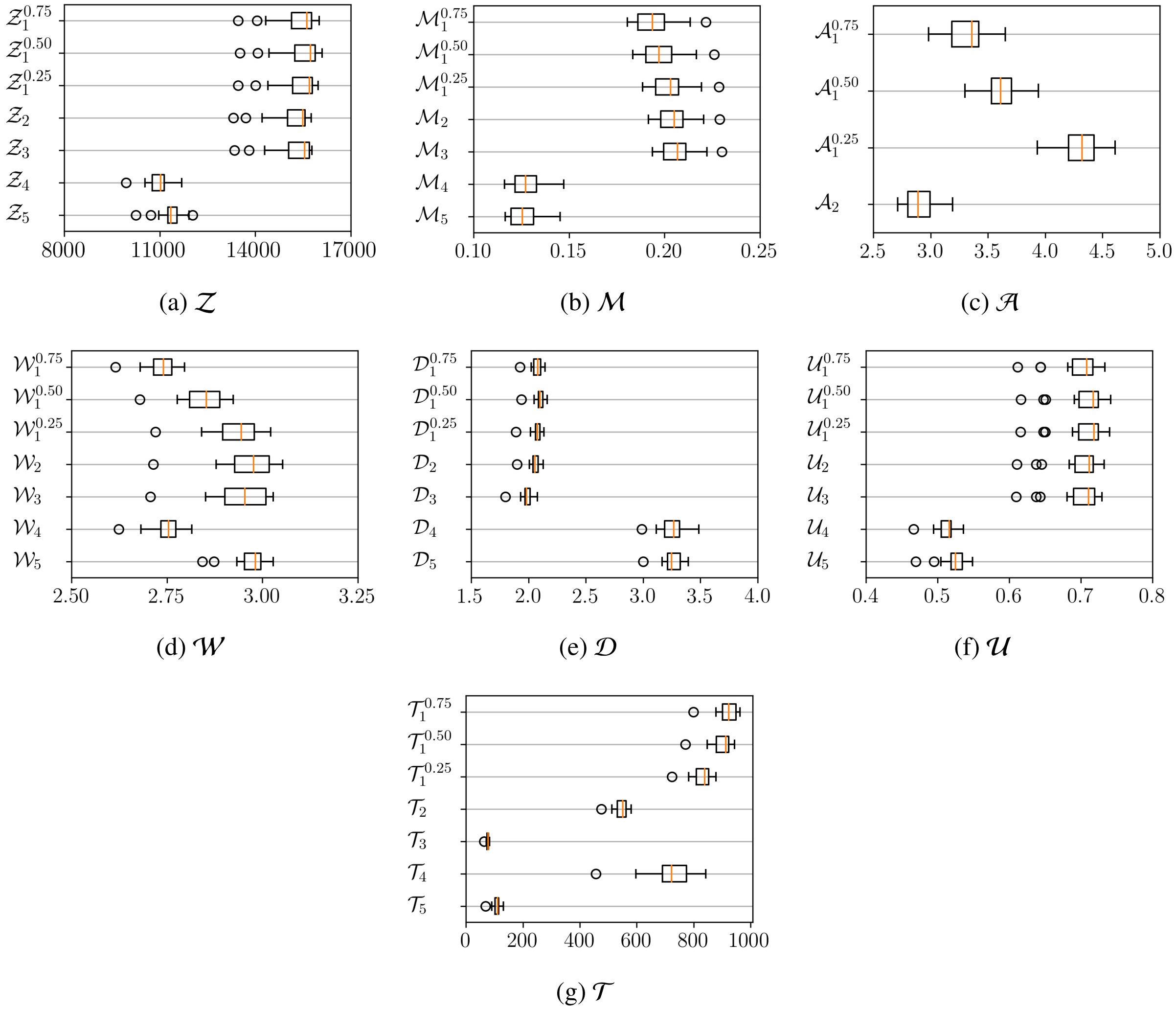
The result of different methods on performance measures in the online experiments are shown in Figure 3.15. The average of these measures are summarized in Table 3.9. Table B.15 provides the p-values of the corresponding paired t-tests for all pairwise comparisons. The presented performance measures in Table 3.9 are obtained by averaging the results of 30 days of simulation. Moreover, we considered three different values for the influence factor β in the full configuration of our proposed method. In terms of total revenue, this table clearly shows that our proposed methodology significantly outperforms the two myopic benchmark methods, as the revenue of our method (independent of the choice of configuration) is at least 34% larger than those of the benchmark methods. The main reason behind this substantial difference is that the benchmark methods tend to aggressively respond to the realized demand without considering the unobserved future demand. Therefore, a high percentage of future demand is likely to be rejected due to the limited capacity of shuttles, especially in circumstances with high demand rate. We can also infer that both allowing for changing routes and incorporating expected demand can lead to a slight (but statistically significant) increase in revenue.

Table B.15 also shows that 100 shuttles are able to serve 20 percent of all requests (around 50000 on average) under our proposed method, which is at least 7% larger than those of the benchmark methods. This table also indicates that we are more likely to switch a shuttle's route when using the expected demand. The highest value of β that yields the highest revenue for the full method indicates that our demand forecast is not near perfect. That is why shuttles can improve their revenues by diverging from the base routes to follow alternative routes. However, the comparison between the revenues of different configurations suggests that allowing for alternating between routes does not result in higher revenue unless we consider the effect of unobserved future demand in our cost-benefit analyses.

In terms of waiting time, we experience somewhat longer waits under our proposed method than

Table 3.9: The result of different methods on the performance measures averaged over simulation runs in the Manhattan case study

Method	β	$\overline{\mathcal{Z}}$	$\overline{\mathcal{M}}$	$\overline{\mathcal{A}}$	$\overline{\mathcal{W}}$	$\overline{\mathcal{D}}$	$\overline{\mathcal{U}}$	$\overline{\mathcal{T}}$
Full	0.75	15295.24	0.19	3.31	2.73	2.07	0.69	916.99
	0.50	15380.86	0.20	3.62	2.84	2.09	0.70	896.05
	0.25	15300.37	0.20	4.31	2.93	2.07	0.70	827.80
No Pred.	–	15093.66	0.21	2.91	2.96	2.05	0.69	544.10
No Alter.	–	15181.47	0.21	–	2.94	1.98	0.69	76.08
Insertion	–	10948.23	0.13	–	2.75	3.25	0.51	714.04
Assignment	–	11331.41	0.13	–	2.97	3.26	0.52	107.82



(a) $\mathcal{Z}$

(b) $\mathcal{M}$

(c) $\mathcal{A}$

(d) $\mathcal{W}$

(e) $\mathcal{D}$

(f) $\mathcal{U}$

(g) $\mathcal{T}$

Figure 3.15: The result of different methods on the performance measures in the Manhattan case study

the Insertion method. However, the detour time is significantly lower under our proposed method. The first observation originates from the fact that the Insertion method assigns customers to their nearby shuttles, and the second one can be justified by the fact that the myopic method inserts customers on a shuttle's route one at a time. Comparing the utilization rates of shuttles reveals that they are about 20% larger under our method than the benchmark methods. This is due to the fact that the number of customers served by the proposed method are much higher than the Insertion method. Hence, shuttle seats are occupied for a larger proportion of the study horizon.

Finally, this table indicates that all methods take less than 1 second to make a decision at each simulation time step, confirming that the proposed method can be implemented in close to real-time in practice. Note that all of the online computations for all the methods except the Insertion method have been parallelized over the shuttles. It is noteworthy that the No Alter. method is even faster than the Assignment method, while the other two configurations have higher computation times. There are 2 reasons for this difference: (1) the Full and No Pred. methods consider alternative routes per each shuttle, (2) the No Alter. method does not need to reschedule the on-board and scheduled customers. Also, incorporating the expected demand results in larger min-cost flow network problems that have longer computation times under the Full method. It should be emphasized that all computations are parallelized over shuttles on a 50 core environment. However, the cost-benefit analysis of alternative routes in the online phase can be also implemented on multiple cores and completed in parallel, resulting in significantly lower computation times of both Full and No Pred. methods.

Overall, the results of our case study once again confirm that the effectiveness of a shuttle dispatching system can be improved immensely by using the historical trip data. It is worth mentioning that although the No Pred. and No Alter. configurations do not use the expected demand in the cost-benefit analysis of online experiments, they exploit this information when generating the routes in the offline phase.

Video 1 visualizes the deployment of all shuttles to serve on-demand requests in the Manhattan area on June 4, 2017 from 7 a.m to 10 a.m. In this video, every shuttle is represented by a circle with a distinct color, and its size corresponds to the number of on-board customers. Also, video 2 demonstrates the deployment of a single shuttle on March 3, 2017 from 7 a.m to 10 a.m. In this video, different alternative routes are represented by different colors. Therefore, the shuttle's color changes when it switches between its alternative routes. The red and blue pins represent the pick-up and drop-off locations of customers, respectively.

3.6 Conclusion

In this chapter, we propose a general framework to solve the shuttle dispatching problem efficiently and with high level of accuracy. Our method includes an offline phase, where the computationally-heavy shuttle routing and scheduling problems are solved. In this phase, we generate a large pool of routes that could be valuable under different realizations of demand. These routes are deliberately generated with high spatio-temporal overlap between them to facilitate the transfer of a shuttle from one to another.

The efficiency of the offline phase stems from a number of polynomial-time, non-myopic local search algorithms that despite posing little computational burden, enable a near-thorough exploration of the feasible region. In the online portion of the framework, shuttles are routed proactively based on the results from the offline phase. Moreover, the online phase enables shuttles to switch between the pool of routes, allowing them to react to stochastic changes in travel patterns.

We applied our methodology to two networks with different network sizes and demand volumes and patterns. For the online experiments, we introduced three configurations of our methodology by posing assumptions about the existence of a demand forecast and availability of high-quality alternative shuttle routes. We compared the performance of these methods with two state-of-the art algorithms for large-scale dynamic dial-a-ride problems. Results indicated that regardless of the configuration, our method led to a substantial increase in the matching rate and the revenue of both case studies.

These experiments also suggested that customers generally experience lower detour times under the proposed method than the benchmark methods. Among the three configurations of our method, the Full method always yielded the highest revenue, but it came at the cost of increased computational time. However, this increase in the computational burden can be partially alleviated by evaluating the alternative routes on multiple cores in parallel.

Through various case studies, we inferred that only following the base routes provides sufficiently large revenues. However, we showed that we can improve the revenues even further with the help of alternating routes and incorporating demand forecast. Finally, the result of the Manhattan case study suggested that considering historical demand, even using a very simple predictive model such as a Poisson distribution, could significantly improve the performance of a shuttle dispatching system.

CHAPTER 4

A Traveler Incentive Program For Promoting Community-Based Ridesharing

4.1 Introduction

In recent years, traffic congestion has become a serious issue around the globe. In a contemporary study of 1,360 cities in 38 countries, Bloomberg CITYLab asserted that traffic congestion costs over $305 Billion per year only in the U.S. (Schneider, 2018). This spike in traffic congestion, especially during morning and evening peak hours, is mainly due to the rising number of solo-driver commuting trips. Despite tremendous expenditure on subsidy, public transit services have failed to alleviate congestion by shifting solo driving toward more sustainable forms of transport. In addition, not only has the recent proliferation of Transportation Network Companies (TNCs) such as Uber and Lyft not addressed this issue, but it has exacerbated congestion, particularly in large cities (Hawkins, 2019).

Peer-to-peer (P2P) ridesharing is a manifestation of the sharing economy business model in the mobility market, and provides a promising solution for mitigating traffic congestion. In contrast to TNCs that produce high empty miles, thereby adversely affecting traffic congestion, P2P ridesharing improves congestion by increasing the utilization rate of empty seats. Additionally, P2P ridesharing provides a unique opportunity for communities to augment transit services by serving their mobility needs internally. In community-based ridesharing—a form of P2P ridesharing for commuters—the members of the community who own cars, henceforth referred to as *drivers*, transport their peers, henceforth referred to as *riders*, along their routes while completing their own personal trips. Aside from its environmental benefits, community-based ridesharing provides a promising mobility solution for the following reasons:

1. **Existing trust**: Drivers and riders are members of the same small community, preventing lack of participation that may arise from lack of trust in large metropolitan areas.
2. **Lack of opportunity cost**: Drivers travel according to their own schedules, and may only take small detours to serve their fellow community members.

3. **Revenue for the community**: The fare of the rides will be set so as to compensate drivers fully for their detours and partially for their base travel costs, as they would be sharing the cost of their base trips with riders.

Despite abundant benefits of ridesharing systems, there are a few obstacles that hinder the adoption of such systems by commuters in practice. First, commuters are in general reluctant to leave their vehicles at home in favor of outsourcing their rides in the morning if they do not have a guarantee for a ride back home. Traditional carpooling services can provide such a guarantee for commuters who have fixed and common working hours. However, carpooling may not be a viable option for commuters whose working hours may shift from one day to the next. Second, in P2P ridesharing participants are available in the network only for a short period of time (compared to TNC drivers and transit providers), which leads to low spatio-temporal proximity among trips. These factors pose a challenge for P2P ridesharing systems achieving a critical mass, and thereby, becoming a viable transportation option in the long run.

This chapter tackles this timely issue by designing a traveler incentive program (TIP) that allocates monetary subsidies to riders and drivers to foster ridesharing participation rates and provide an opportunity for the commuters within a community to serve their mobility needs internally. To this end, this chapter develops a ridesharing system with ride-back guarantee that incentivizes commuters to share their morning and evening trips with their peers. TIP finds the optimal allocation of a set budget to participants so as to: (*i*) subsidize a selective set of rides, and (*ii*) change the travel behavior of a small, carefully selected set of commuters, with respect to their travel time windows. The ultimate goal of TIP is to maximize social welfare of system participants while ensuring that every dollar injected to the system will generate a higher value in social welfare.

In the rest of this chapter, Section 4.2 provides a review of the literature related to P2P ridesharing and using incentives to promote shared mobility services. Next, we carefully define the problem in this chapter and its underlying assumptions in Section 4.3. In Section 4.4, we present a mathematical formulation for the problem and a solution methodology to solve its large instances. Section 4.5 presents the results of several numerical experiments that evaluate different aspects of our proposed methodology. Finally, Section 4.6 finalizes this chapter by summarizing our findings and providing directions for future research.

4.2 Literature Review

In this section, we first present a brief overview of the literature in P2P ridesharing, followed by a review of studies in shared mobility that consider various types of incentives to promote their systems. Finally, we clearly state the contributions of this chapter.

4.2.1 Peer-to-Peer (P2P) Ridesharing

P2P ridesharing is a shared mobility platform that encourages users with similar routes and time schedules to share their rides together (Agatz et al., 2012). In spite of some similarities, one must distinguish P2P ridesharing from other forms of shared mobility platforms such as carpooling (see e.g., Baldacci et al. (2004)), since it does not require long-time commitments from users, as well as ride-sourcing (see e.g., Xu et al. (2020)) or taxi-sharing (see e.g., Alonso-Mora et al. (2017)) since drivers in P2P ridesharing are not treated as employees and their primary intention is to complete their own personal trips. In what follows, we describe a number of major characteristics of P2P ridesharing, henceforth referred to as ridesharing, and its variant forms. For further information on ridesharing, the interested reader is referred to the surveys by Agatz et al. (2012); Furuhata et al. (2013); Tafreshian et al. (2020).

Ridesharing systems can be roughly divided into the two categories of static ridesharing (see e.g., Regue et al. (2016); Wang et al. (2016b); Long et al. (2018)) and dynamic ridesharing (see e.g., Agatz et al. (2011); Lee and Savelsbergh (2015); Masoud and Jayakrishnan (2017c)). In the former case, the trip information of all users is known ahead of time, while in the latter case, users enter the system dynamically and register their trips shortly before their departure times. Most of the studies in P2P ridesharing consider the sets of riders and drivers as two mutually exclusive sets (see e.g., Agatz et al. (2011); Nourinejad and Roorda (2016); Najmi et al. (2017)). However, a few studies relax this assumption and let the system operator decide the most beneficial role (i.e., rider or driver) for each participant (see e.g., Amey (2011); Chen et al. (2019); Tafreshian and Masoud (2020a,b)). The core of a ridesharing system is a ride-matching problem, the solution of which determines the optimal assignment between riders and drivers, users' trip schedules, and drivers' routes. With an intent to make ridesharing convenient for both riders and drivers, many studies consider the simplest form of the ride-matching problem in which every user can be matched with at most one other user (see e.g. Ma et al. (2013); Najmi et al. (2017); Wang et al. (2017)). In order to increase the possibility of matching, however, a number of studies diverge from this assumption by allowing multiple riders per vehicle and transfers between vehicles (see e.g. Stiglic et al. (2015); Masoud and Jayakrishnan (2017b); Chen et al. (2019)). Moreover, a number of studies incorporate the choice of ride-back home guarantee in their models, which motivates rider participation, and increases the level-of-service offered by the system (see e.g. Regue et al. (2016); Lloret-Batlle et al. (2017b); Chen et al. (2019); Hasan et al. (2020)).

Based on the definitions above, the community-based ridesharing proposed in this chapter can be categorized as a static one-to-one ridesharing system with ride-back guarantee. The choice of a static system is supported by the outcome of a ridesharing survey in Berkeley, CA, which concludes that commuters prefer to learn about their rideshare arrangements at least a night before (Deakin et al., 2010). Also, given the assumption that all users select the shortest travel time path as their selected route, one-to-one ride-matching ensures that (*i*) riders do not experience any detour during their travel, and (*ii*) drivers' inconvenience due to pick-ups and drop-offs are minimized. Finally the results of a behavioral study by Brownstone and Golob (1992) indicates that the option of ride-back home guarantee motivates a high percentage of commuters to engage in ridesharing programs.

4.2.2 Incentives in Shared Mobility

Since the introduction of shared mobility services, several studies have emphasized the need for designing incentives to motivate solo drivers to participate in rideshare programs. The proposed incentives can roughly fall into two categories of indirect and direct (a.k.a. financial subsidies) incentives. The employer-provided parking discounts for high occupancy vehicles (HOV) is an example of an indirect incentive that can significantly incentivize daily commuters to shift toward ridesharing (Brownstone and Golob, 1992; Su and Zhou, 2012). Another indirect incentive that proves useful in practice is the possibility of using freeway HOV lanes that leads to savings in commute times (Brownstone and Golob, 1992; Lloret-Batlle et al., 2017b). Finally, integrating ridesharing with public transit and/or other transport programs can reduce the travel cost of commuting and encourage a higher number of commuters to share their rides (Deakin et al., 2010; Nam et al., 2018; Bian and Liu, 2019).

Aside from indirect incentives, many studies stress the necessity of adopting various form of financial subsidiary schemes to promote ridesharing among commuters (Chan and Shaheen, 2012; Agatz et al., 2012). The ultimate goal of financial (or direct) incentives is to maximize fleet utilization, and thereby reduce traffic congestion. Stiglic et al. (2016) conduct a comprehensive case study to evaluate the effect of time flexibility on the matching rate in one-to-one ridesharing systems. Based on the results of their experiments, they emphasize the importance of adopting an incentive scheme that provides monetary benefits to commuters to increase their time flexibility. In a study that focuses on the role of rider-driver cost-sharing strategies in the success of ridesharing programs, Wang et al. (2018b) show that providing ridesharing users with sufficient subsidies can reduce the cost of participation and turn ridesharing into a viable alternative to public transit. They further emphasize the need for designing appropriate subsidizing schemes.

The consideration of financial subsidies is not limited to ridesharing systems and has been studied in other types of shared mobility services. Qian et al. (2017), for instance, introduce ride incentives for groups of passengers in a shared-taxi service as discounts towards their trip fares. They further propose different algorithms that find the best ride incentives to improve total saved mileage. For the operation of ride-sourcing platforms such as Uber and DiDi Chuxing, Zhao and Chen (2019) compare the ex-ante and ex-post destination information models and show the effectiveness of subsidies in attracting more participants under the latter model. They further design a subsidy scheme based on the income of drivers that motivates them to serve farther distant passengers. Luo et al. (2019) develop a new dynamic games approach to find the optimal subsidy policy that accelerates the adoption of automated vehicles (AV's). In their approach, adaptive subsidies are computed based on the state of the AV market penetration process under uncertainty. They claim that the optimal subsidies further incentivize the AV manufacturers to improve their technology, and offer pricing incentives to potential consumers.

More recently, through a joint simulation of car-sharing, bike-sharing and ride-hailing for a city-scale transport system, Becker et al. (2020) found that the highest system-level impacts can be achieved when the operations of shared modes are subsidized. More interestingly, they showed that the total amount of subsidies required for these shared modes is lower than the amount paid for current regular public transport services. Finally, Xiong et al. (2020) propose an integrated system that provides personalized travel alternatives and monetary incentives for travelers. As a part of their system, to reduce traffic congestion they offer alternative departure times for commuters and compensate them with monetary subsidies. This incentive scheme has been commercialized as a mobile app, called incenTrip, and is currently used in the Washington area.

Based on the findings of these studies, we propose two types of monetary incentives that can help improve the social welfare of a community adopting a ridesharing system. The first incentive is targeted toward changing travelers' schedules to increase their chance of being matched. The other incentive attempts to compensate the negative externalities of riders and drivers sharing rides together.

4.2.3 Our Contributions

The contributions of this chapter are as follows:

- We introduce a traveler incentive program for a ridesharing system with guaranteed ride-back, and present a mixed integer nonlinear optimization model that determines the optimal matching, scheduling, and incentive allocation.
- We decompose the mixed integer nonlinear optimization model into a linear model that determines the optimal amount of incentives to each rider-driver pair, and a budget-constrained min-cost flow problem, which is known to be NP-complete.
- We propose a polynomial-time Lagrangian Relaxation method to efficiently find near-optimal solutions for large-scale instances of the problem, and provide a worst-case optimality bound for its performance.
- We propose a budget-balanced variant of the incentive program, which could be solved in polynomial time.
- We perform a comprehensive set of numerical experiments to showcase the impact of the

proposed incentive program on serving the mobility needs in a city.

4.3 Problem Statement

The main focus of this chapter is a static one-to-one ridesharing system for commuters (although In section 4.5 we will investigate a dynamic setting). This program guarantees ride-back services for the riders who register both their morning and evening trips in the system. Let N denote the set of all participants (users) that register their trips in a given day. A participant in the system can be either a driver or a rider in a given day. Therefore, set N can be further partitioned into two disjoint sets of riders, denoted by R, and drivers, denoted by D. We assume that the ridesharing system knows the following information regarding each user $n \in N$:

- $\mathsf{F}(n)$: the value of every unit of time spent on traveling
- $\mathsf{H}(n)$: the value of every unit of distance driven

These values can either be specified directly by having the users answer a short survey, or estimated based on the provided information upon registration for the first time in the system (e.g. occupation, place of residence, vehicle's make and model, etc.). Every user $n \in N$ may register a trip in the morning, represented by n', a trip in the evening, represented by n'', or both. Let $N' = R' \cup D'$ and $N'' = R'' \cup D''$ respectively denote the sets of all trips in the morning and evening. Also, let $D''' \subset D$, $R''' \subset R$, $N''' = R''' \cup D'''$ denote the set of drivers, riders, and users who register both their morning and evening trips in the system, respectively. Due to the similarities between the characteristics of morning and evening trips, we describe our assumptions using only the morning trips in the rest of this section. Table C.1 summarizes the notation used in this chapter.

It is assumed that the study region consists of a large number of stations from/at which trips originate/end. Thus, every trip $n' \in N'$ can be characterized by the following information:

- $\mathsf{I}(n')$: the origin station for the morning trip of user n
- $\mathsf{J}(n')$: the destination station for the morning trip of user n
- $\mathsf{T}(n')$: the desired earliest departure time of user n from the origin station in the morning
- $\mathsf{Q}(n')$: the desired latest arrival time of user n from the destination station in the morning

We further assume that the shortest-path travel time and driving distance between every pair of stations during the morning period are known and stored in the hash tables τ and ρ, respectively. Thus, $\tau_{i,j}/\rho_{i,j}$ represent the shortest-path travel time/distance from station i to station j. We refer to $[\mathsf{T}(n'), \mathsf{Q}(n')]$ as the morning time window of user n. The length of this time window is rather tight, but always greater than or equal to $\tau_{\mathsf{I}(n'),\mathsf{J}(n')}$.

To provide a high quality of service for both riders and drivers participating in the ridesharing system, we assume that in a given time period (e.g., morning) ,each driver will give a ride to at most one rider, and riders complete their trips with at most one driver. This limits the length of detours

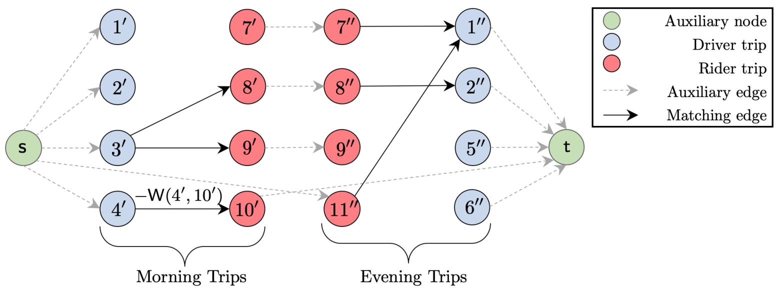


Figure 4.1: Graph $G = (V, E)$ for a ridesharing system with 11 users where $D' = \{1', 2', 3', 4'\}$, $D'' = \{1'', 2'', 5'', 6''\}$, $R' = \{7', 8', 9', 10'\}$, and $R'' = \{7'', 8'', 9'', 10''\}$.

incurred by drivers, and guarantees no detour and transfer for riders.

4.3.1 Static One-to-One Ridesharing with Ride-Back Guarantee

The problem of one-to-one ride-matching with guaranteed ride-backs was first considered by Agatz et al. (2011), where they formulated it as a weighted matching problem with the addition of a set of bundle constraints that relate riders' inbound trips to their outbound trips. In general, this problem may no longer have the unimodularity property, and thus, requires an MIP solver to solve. However, in the cases where inbound and outbound trips occur in two non-overlapping periods (e.g., all the outbound trips occur in the morning peak hours and inbound trips occur in the evening peak hours), Lloret-Batlle et al. (2017b) show that the ride-matching problem can be formulated as a min-cost max flow problem in a weighted directed graph $G = (V, E)$, where the set of nodes is denoted by $V = \{\mathsf{s}, \mathsf{t}\} \cup N' \cup N''$, and the edge set is denoted by E. Figure 4.1 shows an example of such a graph with 11 users. The auxiliary nodes s and t respectively represent the source node and the target node. The edge set E consists of two edge subsets: (i) the auxiliary edges with capacity of 1 unit and 0 units of cost, and (ii) the potential matching edges with capacity of 1 unit and $-\mathsf{W}$ units of cost. The auxiliary edges connect (i) node s to all driver trips $d' \in D'$ and all rider trips $r'' \in R''$ when $r \notin R'''$, (ii) all driver trips $d'' \in D''$ and all rider trips $r' \in R'$ when $r \notin R'''$ to target node t, and (iii) every rider trip $r' \in R'$ to rider trip $r'' \in R''$ when $r \in R'''$. The potential matching edges connect the morning drivers to the morning riders and the evening riders to the evening drivers. A potential match edge exists in graph G if (i) the match is spatio-temporally feasible, and (ii) both parties prefer the match to their other available options outside the ridesharing system, i.e., the match is individually rational.

In what follows, we develop the mathematical equations for spatio-temporal feasibility and individual rationality conditions for the morning period. Similar equations can be derived for the evening period. Spatio-temporal feasibility requires a driver to be capable of providing a ride to a rider within the rider's time window while completing their own trip within their time window.

Thus, the following two equations must be satisfied simultaneously:

$$\max\{\mathsf{T}(d') + \tau_{\mathsf{I}(d'),\mathsf{I}(r')}, \mathsf{T}(r')\} + \tau_{\mathsf{I}(r'),\mathsf{J}(r')} \leq \mathsf{Q}(r'), \tag{4.1a}$$

$$\max\{\mathsf{T}(d') + \tau_{\mathsf{I}(d'),\mathsf{I}(r')}, \mathsf{T}(r')\} + \tau_{\mathsf{I}(r'),\mathsf{J}(r')} + \tau_{\mathsf{J}(r'),\mathsf{J}(d')} \leq \mathsf{Q}(d'). \tag{4.1b}$$

Equations (4.1a) and (4.1b) respectively ensure that the match allows rider r and driver d to complete their trips within their specified time windows. While these equations describe the spatio-temporal feasibility of a match, for a match to be deemed valuable and accepted by both parties, the fare for the ride should be set to an amount that is acceptable by both the driver and the rider. Let $\mathsf{U}(r'|d')$ and $\mathsf{U}(d'|r')$ respectively denote the valuations of rider r and driver d of sharing a ride together. This valuation can be defined as the difference between the cost of sharing the ride and the cost of driving alone (in case the rider does not own a car, this cost is set to the cost of taking a taxi cab). Based on the elicited information from both parties, these valuations can be found as:

$$\mathsf{U}(r'|d') = \mathsf{H}(r)\,\rho_{\mathsf{I}(r'),\mathsf{J}(r')}, \tag{4.2a}$$

$$\begin{aligned}\mathsf{U}(d'|r') = &-\mathsf{H}(d)\,(\rho_{\mathsf{I}(d'),\mathsf{I}(r')} + \rho_{\mathsf{I}(r'),\mathsf{J}(r')} + \rho_{\mathsf{J}(r'),\mathsf{J}(d')} - \rho_{\mathsf{I}(d'),\mathsf{J}(d')})\\ &- \mathsf{F}(d)\,(\tau_{\mathsf{I}(d'),\mathsf{I}(r')} + \tau_{\mathsf{I}(r'),\mathsf{J}(r')} + \tau_{\mathsf{J}(r'),\mathsf{J}(d')} - \tau_{\mathsf{I}(d'),\mathsf{J}(d')}),\end{aligned} \tag{4.2b}$$

where $\mathsf{F}(n) > 0$ and $\mathsf{H}(n) > 0$ respectively denote the values of time (in \$ per unit of time) and distance (in \$ per unit of distance) for user $n \in N$. The statements in parentheses in Equations (4.2b) respectively represent the distance and time of the detour incurred by driver d to serve rider r in the morning. If we assume that all users follow a quasi-linear utility, then the sum of valuations in (4.2a) and (4.2b) yields the potential monetary saving (gain) due to driver d providing a ride to rider r, denoted as $\mathsf{W}(d', r') = \mathsf{U}(d'|r') + \mathsf{U}(r'|d')$. If the gain for a match is non-negative, that is, if $\mathsf{W}(d', r') > 0$, then there exists a pricing mechanism to split the benefits between users r and d so as to ensure they both have non-negative utilities, i.e., both choices are individually rational (e.g., they share the profit based on the length of the two trips). Note that $\mathsf{W}(r'', d'') = \mathsf{U}(r''|d'') + \mathsf{U}(d''|r'')$ can be computed similarly for the evening trips.

4.3.2 Incentive Design

The graph corresponding to the ridesharing problem with guaranteed ride-back can be sparse, as demonstrated in the example in Figure 4.1. This sparsity is partly due to individuals' rather tight travel time windows, and partly a result of the heterogeneity in their valuations of options. It is not surprising to see trip requests with tight time windows, as this ensures the rides to be in congruence with travelers' preferences. Furthermore, it is realistic to assume that individuals who own cars are in a superior financial status than those who do not. As such, having drivers with higher values of time, and possibly distance, get compensated for their detours by riders may lead to the ridesharing option not being affordable for a large portion of riders. To tackle these issues, we introduce two types of incentives to increase the number of edges between trip nodes in graph G, eventually leading to higher percentage of matches, and possibly higher levels of social welfare in the community.

Behavioral Adjustment (BA) Incentives: When registering a trip, the desired earliest departure time and latest arrival time are two of the trip characteristics that a participant has to specify. A wider time window for a driver implies a potentially longer detour and more flexibility in departure time. For riders, a wider time window only implies higher flexibility in departure time, as a rider's travel duration would be that of their shortest-path travel time. Clearly, by providing tighter time windows, participants would receive matches that are more congruent with their preferences. A behavioral adjustment (BA) incentive encourages participants to be more flexible with their travel time windows. Let us subsidize user n by an incentive rate of $\mathsf{F}(n)$, equal to their value of time, to widen their morning trip time window for $\gamma(n')$ units, with the assumption that their disutility from leaving earlier than $\mathsf{T}(n')$ and arriving later than $\mathsf{Q}(n')$ are the same and are proportional to $\mathsf{F}(n)$. If we let $\gamma^{-}(n')$ and $\gamma^{+}(n')$ respectively denote the amount of time extension in time window of trip n' from left and right, the adjusted time window can be shown as $[\mathsf{T}(n') - \gamma^{-}(n'), \mathsf{Q}(n') + \gamma^{+}(n')]$ subject to the constraint $\gamma^{-}(n') + \gamma^{+}(n') = \gamma(n')$. Note that the right value for $\gamma(n')$ depends on other users' trip information; hence, it must be determined by the system operator as a part of the ride-matching problem.

Individual Rationality (IR) Incentives: Consider a rider-driver pair (d', r') for whom spatio-temporal conditions (i.e., Equations (4.1a)-(4.1b)) are satisfied, but the individual rationality conditions are violated (i.e., $\mathsf{W}(d', r') < 0$). An individual rationality (IR) incentive, denoted by $\lambda(d', r')$, provides subsidies that allow for introducing such a link in graph G with a non-negative gain.

The consideration of these incentives in our ridesharing system has two important consequences. First, the Equations in (4.1) can no longer be used to determine the spatio-temporal feasibility of a match since the adjusted time windows depend on the unknown values of the BA incentives. Secondly, the rider and driver valuations from sharing a ride in (4.2) as well as the gains of potential matches are dependent upon the unknown values of both incentive types. These consequences clearly suggest that no longer can we find the optimal ride-matching using the min-cost max flow problem described above. As such, in this chapter we develop a ride-matching problem that determines the optimal matching, trip scheduling, and incentive allocation that maximizes the social welfare given a monetary budget of B dollars. In order to make sure that the available budget is used wisely, we further require that every dollar spent on subsidy contributes more than one dollar to the system's social welfare.

4.4 Solution Methodology

In this section, we first present the mathematical formulation of the TIP for a simpler system involving only one-time trips with no ride-back guarantee (e.g., the morning trips) given the assumptions provided in the previous section. Next, we modify this mathematical formulation to model the TIP for a system that includes both the morning and evening trips. We propose an efficient algorithm to solve large-scale instances of this problem in a timely manner, and prove a worst-case optimality bound for its performance. Finally, we propose a budget-balanced counterpart of the TIP through taxation.

4.4.1 Mathematical Formulation For the Morning Trips

Let us consider a ridesharing system that includes only the morning trips N'. Also, let A' denote the set of all driver-rider trip pairs in the morning, i.e., $A' = \{(d', r') \in D' \times R'\}$. Based on the assumptions provided in Section 4.3, the optimal matching, scheduling, and incentive allocation of the system can be obtained by solving the mixed integer nonlinear program (MINLP) presented in (4.3). In this formulation, there are eight sets of decision variables. The decision variable $x(a')$ is a binary variable that holds the value 1 if the pair of trips in $a' \in A'$ share their rides together, and the value 0 otherwise. The continuous decision variables $t(n')$ and $q(n')$ denote the start and end time of the trip $n' \in N'$, respectively. The decision variable $\gamma(n')$ can be defined as the amount of extension in the time window of trip n'. We further define two variables, $\gamma^-(n')$ and $\gamma^+(n')$, to denote the amount of extension in the time window of the morning trip n' from the left and the right, respectively. Finally, the decision variables $\lambda(a')$ and $w(a')$ respectively represent the IR incentive and the gain for the match between a driver-rider trip pair $a' \in A'$.

Let ϵ be an infinitesimal positive value. The objective in (4.3a) maximizes the difference between the system's social welfare and the total amount of subsidies spent on the BA and IR incentives. Choosing this objective enables us to maximize social welfare while allocating subsidies only when the added value to social welfare is strictly higher than the amount of subsidy; that is, for each dollar spent on subsidy, a return-to-investment of more than one dollar can be obtained on social welfare. Constraint (4.3b) defines the adjusted gain of match (d', r') in the morning as the sum of the original savings due to driver d sharing their ride with rider r and the subsidies allocated to these participants and their coalition. Constraint (4.3c) ensures that if users r and d are determined to share a ride in the morning, i.e., if $x(d', r') = 1$, then their coalition is individually rational, i.e., $w(d', r') \geq 0$. Constraints (4.3d) and (4.3e) together guarantee that every trip starts and ends within its adjusted time windows. Constraint (4.3f) states that if driver trip d' is matched with rider trip r', the difference between their trips' start times must be at least equal to the shortest-path travel time between their origin stations. Constraint (4.3g) defines the end time of rider trip r' as the sum of its start time and the duration of the trip. Constraint (4.3h) ensures that the difference between the end time of driver trip d' and rider trip r', if matched together, must be equal to the shortest-path travel time from the rider's destination station to driver's destination station. Constraint (4.3i) defines the total extension in a trip's time window as the sum of the extensions from the left and the right. Constraints (4.3j) and (4.3k) respectively ensure that every rider is matched with at most one driver, and each driver serves at most one rider in the morning. Finally, Constraint (4.3l) ensures that the total amount of allocated subsidy is lower than the available budget. Constraints (4.3m)-(4.3o) are the non-negativity and integrality constraints.

$$\max \quad \sum_{a'\in A'} w(a')\,x(a') - (1+\epsilon)\Big(\sum_{n'\in N'} \mathsf{F}(n)\,\gamma(n') + \sum_{a'\in A'} \lambda(a')\Big) \tag{4.3a}$$

$$\text{s.t.} \quad w(d',r') = \mathsf{W}(d',r') + \lambda(d',r')\,, \qquad \forall\,(d',r')\in A'\,, \tag{4.3b}$$

$$w(a')\,x(a') \geq 0\,, \qquad \forall\, a'\in A'\,, \tag{4.3c}$$

$$t(n') \geq \mathsf{T}(n') - \gamma^{-}(n')\,, \qquad \forall\, n'\in N'\,, \tag{4.3d}$$

$$q(n') \leq \mathsf{Q}(n') + \gamma^{+}(n')\,, \qquad \forall\, n'\in N'\,, \tag{4.3e}$$

$$x(d',r')\,\big(t(r') - t(d') - \tau_{\mathsf{I}(d'),\mathsf{I}(r')}\big) \geq 0\,, \qquad \forall\,(d',r')\in A'\,, \tag{4.3f}$$

$$q(r') = t(r') + \tau_{\mathsf{I}(r'),\mathsf{J}(r')}\,, \qquad \forall\, r'\in R'\,, \tag{4.3g}$$

$$x(d',r')\,\big(q(d') - q(r') - \tau_{\mathsf{J}(r'),\mathsf{J}(d')}\big) = 0\,, \qquad \forall\,(d',r')\in A'\,, \tag{4.3h}$$

$$\gamma(n') = \gamma^{-}(n') + \gamma^{+}(n')\,, \qquad \forall\, n'\in N'\,, \tag{4.3i}$$

$$\sum_{d'\in D'} x(d',r') \leq 1\,, \qquad \forall\, r'\in R'\,, \tag{4.3j}$$

$$\sum_{r'\in R'} x(d',r') \leq 1\,, \qquad \forall\, d'\in D'\,, \tag{4.3k}$$

$$\sum_{n'\in N'} \mathsf{F}(n)\,\gamma(n') + \sum_{a'\in A'} \lambda(a') \leq \mathsf{B}\,, \tag{4.3l}$$

$$t(n'),\ q(n'),\ \gamma^{-}(n'),\ \gamma^{+}(n') \geq 0\,, \qquad \forall\, n'\in N'\,, \tag{4.3m}$$

$$\lambda(a') \geq 0\,, \qquad \forall\, a'\in A'\,, \tag{4.3n}$$

$$x(a') \in \{0,1\}\,, \qquad \forall\, a'\in A'\,. \tag{4.3o}$$

The formulation in model (4.3) involves nonlinear statements in both the objective function and constraints, which make the ride-matching problem intractable to solve, especially for real-size networks. However, there are a few properties of this formulation that help us reduce its size considerably. First, let us emphasize the fact that the objective function in (4.3a) implies that the unmatched users in any feasible solution will not receive any subsidy. Additionally, this objective function indicates that the optimal solution never assigns any IR incentive to any user, as one unit of IR incentive would decrease the objective function by ϵ. (Note that IR incentives will not be trivially zero when we introduce the ride-back guarantee component in section 4.4.2). Another observation is that Constraints (4.3b)-(4.3i) indicate that the optimal solutions for the incentives and trip start and end times depend on the matching variables. As such, we present a pre-processing procedure in Algorithm 4.1 that allows us to reduce the problem in (4.3) to a well-known combinatorial problem.

This algorithm takes the trip information of all users and the original gains of all pairs of drivers and riders as its input. The primary goal of this algorithm is to determine the set of all potential matches in the morning, denoted by A', the optimal values of incentives required for driver d and rider r to have a feasible match, denoted by $\Psi(d', r')$, and the objective coefficient of this pair, denoted by $\mathsf{C}(d', r')$. The algorithm starts with letting A' be an empty set. Next, it iterates over all pairs of riders and drivers to check whether they can be a potential match and if the answer is yes, further find the optimal required subsidies for such pairs. More specifically, in lines 3 to 6, we check the spatio-temporal feasibility of pair (d', r') by solving a linear problem, and storing the optimal trip start times and BA incentives of driver d and rider r conditional on $x(d', r') = 1$, denoted by $\bar{t}(d'|r')$, $\bar{\gamma}(d'|r')$, $\bar{t}(r'|d')$, and $\bar{\gamma}(r'|d')$, respectively. Next, in lines 7, we determine the optimal IR incentive which is only positive if the corresponding original gain is negative. Finally, we calculate the objective coefficient $\mathsf{C}(d', r')$ as the difference between the original gain and incurred dis-utilities of driver d and rider r from sharing their morning rides together. Also, the total allocated subsidy to (d', r') conditional on them sharing their rides together, denoted by $\Psi(d', r')$, can be computed as the sum of the optimal BA and IR incentives.

Upon this polynomial-time pre-processing procedure, the problem in (4.3) reduces to a budget-constrained matching problem presented in (4.5). Note that this formulation clearly implies that those pairs of trips whose IR incentives are positive cannot be a part of the optimal solution.

$$\begin{aligned}
\max \quad & \sum_{a' \in A'} \mathsf{C}(a')\, x(a') && && \text{(4.5a)} \\
\text{s.t.} \quad & \sum_{d' \in D'} x(d', r') \leq 1 \ , && \forall\ r' \in R' \ , && \text{(4.5b)} \\
& \sum_{r' \in R'} x(d', r') \leq 1 \ , && \forall\ d' \in D' \ , && \text{(4.5c)} \\
& \sum_{a' \in A'} \Psi(a')\ x(a') \leq \mathsf{B} \ , && && \text{(4.5d)} \\
& x(a') \in \{0, 1\} \ , && \forall\ a' \in A' \ . && \text{(4.5e)}
\end{aligned}$$

By a reduction from the knapsack problem, the problem in (4.5) can be shown to be NP-hard.

Algorithm 4.1: The pre-processing procedure for the morning trips

```
Input: $\mathsf{I}$, $\mathsf{J}$, $\mathsf{T}$, $\mathsf{Q}$, $\mathsf{F}$, $\mathsf{W}$ .
Output: $A'$, $\mathsf{C}, \Psi$ .
Initialize $A' \leftarrow \varnothing$ ;
for $(d', r') \in D' \times R'$ do
    Solve the following linear problem:
        min  $z = \mathsf{F}(d)\,(\gamma^-(d') + \gamma^+(d')) + \mathsf{F}(r)\,(\gamma^-(r') + \gamma^+(r'))$    (4.4a)
        s.t. $t(n') \geq \mathsf{T}(n') - \gamma^-(n')$ ,    $\forall\ n' \in \{d', r'\}$ ,    (4.4b)
             $t(d') + \tau_{\mathsf{I}(d'),\mathsf{I}(r')} \leq t(r')$ ,    (4.4c)
             $t(r') + \tau_{\mathsf{I}(r'),\mathsf{J}(r')} \leq \mathsf{Q}(r') + \gamma^+(r')$ ,    (4.4d)
             $t(r') + \tau_{\mathsf{I}(r'),\mathsf{J}(r')} + \tau_{\mathsf{J}(r'),\mathsf{J}(d')} \leq \mathsf{Q}(d') + \gamma^+(d')$ ,    (4.4e)
             $t(d'),\ t(r'),\ \gamma^-(d'),\ \gamma^+(d'),\ \gamma^-(r'),\ \gamma^+(r') \geq 0$ .    (4.4f)
    if the problem in (4.4) is FEASIBLE then
        Retrieve optimal solution $(t^*(d'), t^*(r'), \gamma^{-^*}(r'), \gamma^{+^*}(r'), \gamma^{-^*}(d'), \gamma^{+^*}(d))$ and objective $z^*$ ;
        Let $\bar{t}(d'|r'),\ \bar{t}(r'|d') \leftarrow t^*(d'),\ t^*(r')$ ;
        Let $\overline{\gamma}(d'|r'),\ \overline{\gamma}(r'|d') \leftarrow \gamma^{-^*}(d') + \gamma^{+^*}(d'),\ \gamma^{-^*}(r') + \gamma^{+^*}(r')$ ;
        Let $\overline{\lambda}(d', r') \leftarrow \max\{0, -\mathsf{W}(d', r')\}$ ;
        Let $\mathsf{C}(d', r') = \mathsf{W}(d', r') - (1 + \epsilon)\, z^* - \epsilon\, \overline{\lambda}(d', r')$ ;
        Let $\Psi(d', r') \leftarrow \mathsf{F}(d)\, \overline{\gamma}(d'|r') + \mathsf{F}(r)\, \overline{\gamma}(r'|d') + \overline{\lambda}(d', r')$ ;
        Update $A' \leftarrow A' \cup \{(d', r')\}$ ;
```

As such, Berger et al. (2011) propose a polynomial time approximation scheme (PTAS) to solve the problem. The core of this PTAS is a Lagrangian relaxation-based method the solution of which has an objective that is different from the optimal one by at most $2\,\mathsf{C}_{\max}$, where $\mathsf{C}_{\max}$ denotes the largest weight. In Section 4.4.3, we extend this method to find a near-optimal solution for the problem involving both the morning and evening trips.

4.4.2 Mathematical Formulation For the Morning and Evening Trips

Let A'' denote the set of all rider-driver trip pairs in the evening, i.e., $A'' = \{(r'', d'') \in R'' \times D''\}$. This set can be generated by applying the pre-processing procedure described in Algorithm 4.1 to the evening trips. As a result, the ride-matching problem with the morning and evening trips can be formulated as the binary program in (4.6). The objective function in (4.6a) seeks to maximize social welfare while ensuring that each dollar spent on subsidy returns more than a dollar in social welfare. Constraints (4.6b)-(4.6e) ensure that all users are served at most once both in the morning and in the evening. Constraint (4.6f) ensures that any rider in R''' is served in the evening if and only if served in the morning. Constraint (4.6h) sets a limit on the allocated budget.

$$
\begin{aligned}
\max \quad & \sum_{a \in A' \cup A''} \mathsf{C}(a)\, x(a) && && (4.6a)\\
\text{s.t.} \quad & \sum_{d' \in D'} x(d', r') \leq 1 \,, && \forall\, r' \in R' \,, && (4.6b)\\
& \sum_{r' \in R'} x(d', r') \leq 1 \,, && \forall\, d' \in D' \,, && (4.6c)\\
& \sum_{d'' \in D''} x(r'', d'') \leq 1 \,, && \forall\, r'' \in R'' \,, && (4.6d)\\
& \sum_{r'' \in R''} x(r'', d'') \leq 1 \,, && \forall\, d'' \in D'' \,, && (4.6e)\\
& \sum_{d' \in D'} x(d', r') = \sum_{d'' \in D''} x(r'', d'') \,, && \forall\, r \in R''' \,, && (4.6f)\\
& \sum_{a \in A' \cup A''} \Psi(a)\; x(a) \leq \mathsf{B} \,, && && (4.6g)\\
& x(a) \in \{0, 1\} \,, && \forall\, a \in A' \cup A'' \,. && (4.6h)
\end{aligned}
$$

Following our discussion in Section 4.3, we observe that the problem in (4.6) is similar to that of Lloret-Batlle et al. (2017b) with the addition of Constraint (4.6g). Thus, by defining a set of auxiliary nodes and edges, we can construct a min-cost max flow network similar to the one explained in Section 4.3 with the exception that the cost of potential matching edges can be set as $-\mathsf{C}$. Now, if for every potential matching edge we define a new cost, namely the usage fee, and set it to Ψ, problem (4.6) turns out to be a budget-constrained min-cost flow problem (Holzhauser et al., 2016) as follows:

$$
\begin{aligned}
\max \quad & \sum_{e \in E} \mathsf{C}(e)\, x(e) && && (4.7a)\\
\text{s.t.} \quad & \sum_{e \in \delta^+(v)} x(e) - \sum_{e \in \delta^-(v)} x(e) = 0 \,, && \forall\, v \in N' \cup N'' \,, && (4.7b)\\
& \sum_{e \in E} \Psi(e)\, x(e) \leq \mathsf{B} \,, && && (4.7c)\\
& x(e) \in \{0, 1\} \,, && \forall\, e \in E && (4.7d)
\end{aligned}
$$

where A''' is the set of auxiliary edges as described in Section 4.3, $E = A' \cup A'' \cup A'''$, and $\delta^+(v)$ and $\delta^-(v)$ denote the set of in-going and out-going edges of node v in graph G, respectively. It is worth mentioning that, unlike the previous case, the IR incentives are not always equal to zero. Consider the case that $\mathsf{C}(d_1', r') < 0$, $\mathsf{C}(r'', d_2'') > 0$, and $\mathsf{C}(d_1', r') + \mathsf{C}(r'', d_2'') > 0$ for arbitrary users d_1, r, d_2. In this case, we may have $x^*(d_1', r') = x^*(r'', d_2'') = 1$ which implies that $\lambda^*(d_1', r') > 0$.

This problem is clearly a generalization of the problem in (4.5), and thus, can be shown to be NP-hard. Due to the similarities of these two problems, we present a solution methodology that extends the method proposed by Berger et al. (2011) to find near-optimal solutions for large-scale instances of problem (4.7). This method solves in polynomial time and does not require a commercial optimization engine.

We finalize this section by describing a post-processing procedure to find the optimal trip start time, trip end time, incentives, and gains for the original MINLP based on the optimal matching x^*. For any pair (d', r') such that $x^*(d', r') = 1$, the optimal values for these variables can be calculated as:

$$t^*(r') = \bar{t}(r'|d')\,, \tag{4.8a}$$

$$q^*(r') = t^*(r') + \tau_{\mathsf{I}(r'),\mathsf{J}(r')}\,, \tag{4.8b}$$

$$\gamma^*(r') = \bar{\gamma}(r'|d')\,, \tag{4.8c}$$

$$t^*(d') = \bar{t}(d'|r')\,, \tag{4.8d}$$

$$q^*(d') = \bar{t}(d') + \tau_{\mathsf{I}(r'),\mathsf{J}(r')} + \tau_{\mathsf{J}(r'),\mathsf{J}(d')}\,, \tag{4.8e}$$

$$\gamma^*(d') = \bar{\gamma}(d'|r')\,, \tag{4.8f}$$

$$w^*(d', r') = \mathsf{W}(d', r') + \bar{\lambda}(d', r')\,, \tag{4.8g}$$

$$\lambda^*(d', r') = \bar{\lambda}(d', r')\,, \tag{4.8h}$$

4.4.3 A Lagrangian Relaxation Based Method

Lagrangian Relaxation (LR) is a well-known decomposition technique that proves to be useful in finding a lower bound (in minimization problems) for large MIP problems (Fisher, 1981). The MIP problems usually involve some complicating constraints that make them hard to solve. LR takes advantage of such constraints by dualizing them in the objective function and solving an easier subproblem that does not include these constraints. In problem (4.7), for instance, removing the budget constraint leaves us with a min-cost max flow problem. In what follows, we present a solution procedure based on LR that solves in polynomial time and yields a lower bound for the optimal objective of problem (4.7). This procedure is summarized in Algorithm 4.2. For illustrative purposes, consider the small min-cost max flow network G in Figure 4.2.

The LR subproblem can be defined by relaxing Constraint (4.7c) and dualizing it in the objective function with a non-negative Lagrange multiplier, denoted by β, as follows:

$$\max \quad \sum_{e \in E} \mathsf{C}(e)\, x(e) - \beta \Big(\sum_{e \in E} \Psi(e)\, x(e) - \mathsf{B} \Big) \tag{4.9a}$$

$$\text{s.t.} \quad (4.7b) \text{ and } 0 \le x(e) \le 1,\ \forall\, e \in E\,.$$

The objective function in (4.9a) can be simply written as:

$$\mathsf{LR}(\beta) = \sum_{e \in E} \mathsf{C}_\beta(e)\, x(e) + \beta\, \mathsf{B}\,, \tag{4.10}$$

where $\mathsf{C}_\beta(e) = \mathsf{C}(e) - \beta\, \Psi(e)$. Therefore, given the value of β, the cost of the potential matching edge e in graph G can be set to $-\mathsf{C}_\beta(e)$, and the difference between the constant $\beta\,\mathsf{B}$ and the optimal cost of network yields an upper bound for the original problem in (4.7). We know that LR is a convex piece-wise linear function of β, and thus, this upper bound will be minimized at β^*. Also note that the min-cost max flow problem in a directed acyclic graph (DAG) with unit capacities can be solved in strongly polynomial time of $O(|V|^2|E|)$ using the successive shortest path algorithm (Ahuja et al., 1988). These two facts lead us to the conclusion that β^* can be found in polynomial time of $O(|V|^4|E|^2)$ using the parametric search method proposed by Megiddo (1978).

If we let ϵ be a positive infinitesimal number, solving the min-cost max flow in graph G with costs respectively set as $\mathsf{C}_{\beta^*-\epsilon}$ and $\mathsf{C}_{\beta^*+\epsilon}$ gives rise to two flows $S_l \subset E$, and $S_h \subset E$ that minimize the LR subproblem. It is easy to show that unless the solution of both is the same which implies optimality, S_l, violates the budget constraint while S_h satisfies it and hence attributes to a feasible solution for problem in (4.7). The main issue is that the objective of flow S_h may be arbitrarily far from the optimal one denoted by OPT. In order to tackle this issue, we modify the patching method proposed by Berger et al. (2011) to construct a solution based on the two flows, S_l and S_h, that yields a worst-case optimality bound for our problem. For our small example, for instance, this search can result in two flows S_h and S_l presented in Figures 4.3(a) and 4.3(b).

Algorithm 4.2 details our proposed patching method that utilizes a feasible flow, S_h and an infeasible one, S_h to construct a high-quality solution. Let us define the set of forward edges, E_f, as all edges in S_l that are not in S_h, and the set of backward edges, E_b, as all edges in S_l that are not in S_h. Also, denote the set of reversed edges in E_b by $\overline{E}_b$. Two flows S_l and S_h are adjacent if and only if the graph induced by edges in $\overline{E}_b \cup E_f$, denoted by G', has only one cycle (Gallo and Sodini, 1978). In Algorithm 4.2, we first turn S_l and S_h into two adjacent extreme flows of the solution polytope of the LR subproblem given β^*.

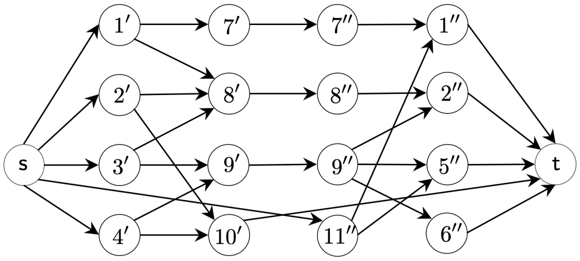


Figure 4.2: A small example of graph G.

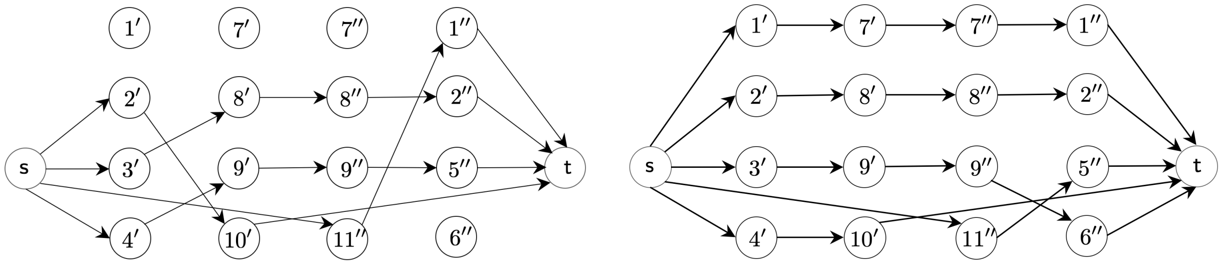
Algorithm 4.2: A near-optimal solution for P2P ridesharing with ride-back guarantee

```
Input: $\mathsf{C}, \Psi, G = (V, E)$ .
Output: $\tilde{S}$ .
Find the optimal Lagrangian multiplier $\beta^*$ and two sets of matching edges $S_h$ and $S_l$ such that
  $\mathsf{C}_{\beta^*}(S_h) = \mathsf{C}_{\beta^*}(S_l)$ and $\Psi(S_h) \le \mathsf{B} < \Psi(S_l)$ ;
Let $E_f, E_b \leftarrow S_l \setminus S_h, S_h \setminus S_l$ ;
Find the set of alternating cycles $\mathscr{C}$ in Graph $G' = (V, \overline{E}_b \cup E_f)$ ;
repeat
    Pick an arbitrary cycle $X \in \mathscr{C}$ ;
    Let $X_f, X_b \leftarrow X \cap E_f, \overline{X} \cap E_b$ ;
    Let $S \leftarrow (S_h \cup X_f) \setminus X_b$ ;
    if $\Psi(S) \le \mathsf{B}$ then
        Update $S_h \leftarrow S$ ;
    else
        Update $S_l \leftarrow S$ ;
    Update $E_f, E_b \leftarrow S_l \setminus S_h, S_h \setminus S_l$ ;
    Find the set of alternating cycles $\mathscr{C}$ in Graph $G' = (V, \overline{E}_b \cup E_f)$ ;
until $|\mathscr{C}| = 1$;
if $\Psi(S_h) = \mathsf{B}$ then
    Let $\tilde{S} \leftarrow S_h$ ;
else
    Find sequence $Y = Y_f \cup Y_b$ in cycle $X$ using the Gasoline Lemma ;
    Find $\tilde{S}$ by solving a min-cost max flow in graph $G'' = (V, (S_h \cup Y_f) \setminus Y_b)$ with costs $\mathsf{C}$ ;
```

The following claim states that $\overline{E}_b \cup E_f$ upon minor adjustments contains a set of cycles.

Claim 4.1. *Graph G' contains a non-empty collection of cycles denoted by $\mathscr{C}$.*

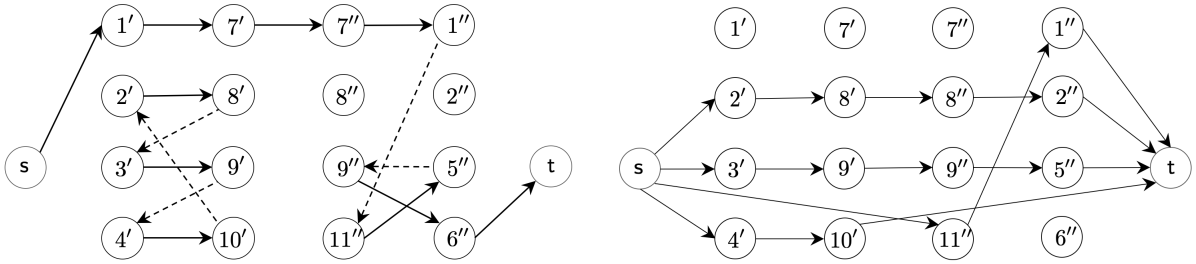
Proof. Based on the definition of $\overline{E}_b \cup E_f$ and the fact that both S_l and S_h satisfy Constraint (4.7b),



(a) S_h

(b) S_l

Figure 4.3: A budget-feasible and -infeasible optimal flow for LR subproblem of the small example



(a) $\overline{E}_b \cup E_f$ (b) S_h after cancelling a cycle

Figure 4.4: The cycles in $\mathscr{C} = \{X_1, X_2\}$ and the flow S_h after cancelling cycle induced by the edges in $X_1 = \{(2', 8'), (8', 3'), (3', 9'), (9', 4'), (4', 10'), (10', 2')\}$. In part (a), the solid arrows denote edges in E_f and the dashed arrows denote the edges in $\bar{E}_b$.

we infer that all nodes in V except the source and target node have in-degrees and out-degrees of either 1 or 0. We further infer that the net outflow of s is equal to the net inflow of t. As a result, $\overline{E}_b \cup E_f$ can be decomposed into a combination of cycles and paths that include at least one of the following:

- simple path(s) from s to t
- simple path(s) from t to s
- simple cycle(s)

Note that by adding zero-cost auxiliary edge(s) from t to s in the first case, and zero-cost auxiliary edge(s) from s to t in the second case, paths can turn into cycles. □

If the cardinality of $\mathscr{C}$ is one, by definition S_l and S_h are adjacent. Otherwise, until the cardinality of $\mathscr{C}$ is equal to one, we repeat the process described in lines 4 to 13 of Algorithm 4.2. More specifically, we draw one arbitrary cycle from $\mathscr{C}$ at a time and add its forward edges to S_h and remove its backward edges from S_h. This yields a new feasible flow, denoted by S, the Lagrangian objective of which is equal to those of S_h and S_l. Now, depending on whether this flow satisfies the budget constraint, we replace it by S_h or S_l. At the end of this process, if flow S_h depletes all the budget, we claim that it is the optimal solution. Otherwise, we continue to the patching method described below. In Figure 4.4(a), we show that $\mathscr{C}$ consists of two cycles $X_1 = \{(2', 8'), (8', 3'), (3', 9'), (9', 4'), (4', 10'), (10', 2')\}$ and $X_2 = \{(\mathsf{s}, 1'), (1', 7'), (7', 7''), (7'', 1''), (1'', 11''), (11'', 5''), (5'', 9''), (9'', 6''), (6'', \mathsf{t})\}$ in our small example. Note that X_2 is indeed a path from s to t which can turn into a cycle by adding the auxiliary edge (t, s). Since $|\mathscr{C}| = 2$, we need to cancel one of these cycles using the procedure described above. Note that the forward edges in X_1 and X_2 are shown by solid lines and backward edges are shown by dashed lines. In Figure 4.4(b), we show the result of cancelling cycle X_1 which provides a new flow S_h assuming that the budget constraint remains feasible.

Now, let us denote the last remaining cycle in $\mathscr{C}$ by $X = X_f \cup \overline{X}_b$, where X_f and $\overline{X}_b$ respectively denote the forward edges and backward edges in X. If we use the procedure described above to remove the last cycle X, flow S_h will turn into flow S_l, i.e., the two solutions will collapse, providing an infeasible solution with respect to the budget constraint. Also note that:

$$\mathsf{C}(S_h) \geq \mathsf{OPT} - \beta^* \left(\mathsf{B} - \Psi(S_h)\right).$$

Therefore, finding a solution whose subsidy is closer to B improves the optimality bound of our solution for problem (4.7). To this end, we present a method to alter flow S_h using a subsequence of edges in cycle X. Similar to the patching method in (Berger et al., 2011), the core of our method is based on the well-known Gasoline Lemma (see e.g. Lin and Kernighan (1973)).

Let us define the following weights for edges $e = (i, j)$ in X:

$$\alpha(i,j) = \begin{cases} \mathsf{C}_{\beta^*}(i,j), & \text{if } (i,j) \in X_f, \\ -\mathsf{C}_{\beta^*}(j,i), & \text{if } (i,j) \in X_b. \end{cases} \tag{4.11}$$

For any rider $r \in R'''$ in X, we further adjust the weight of the edge that starts from or ends at r'' as follows:

$$\alpha(r'', j) = \alpha(r'', j) + \max\left\{\mathsf{C}(d'_l(r'), r'),\ \mathsf{C}(d'_h(r'), r'),\ 0\right\}, \tag{4.12}$$

$$\alpha(i, r'') = \alpha(i, r'') - \max\left\{\mathsf{C}(d'_l(r'), r'),\ \mathsf{C}(d'_h(r'), r'),\ 0\right\}, \tag{4.13}$$

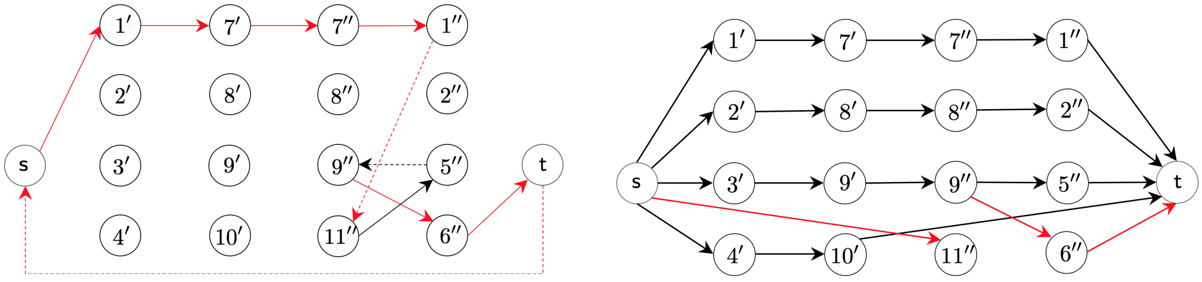
where $d'_l(r')$, $d'_h(r')$ respectively denote the driver trips that are matched with rider trip r' in flows S_l and S_h. If there is no such matches in either of these flows, we let the corresponding C value be zero. Also, note that the α values for edges in X add up to zero. Next, we apply the Gasoline Lemma to the edges in X with the weights set as α. The Gasoline Lemma is presented in the following.

Lemma 4.1. *(Gasoline Lemma) Given a sequence of k real numbers $\alpha_0, ..., \alpha_{k-1}$ such that $\sum_{j=0}^{k-1} \alpha_j = 0$, there is an index $i \in \{0, ..., k-1\}$ such that, for any $0 \leq h \leq k-1$,*

$$\sum_{j=i}^{i+h} \alpha_{j(\bmod k)} \geq 0, \quad \forall\, h \in \{0, ..., k-1\}.$$

Proof. Let $i' \in \{0, ..., k-1\}$ be the index for which $\sum_{j=0}^{i'} \alpha_j$ is minimum and let i be $(i'+1)(\bmod k)$. Thus, we have:

$$\sum_{j=i}^{i+h} \alpha_{j(\bmod k)} = \sum_{j=0}^{i+h} \alpha_{j(\bmod k)} - \sum_{j=0}^{i'} \alpha_{j(\bmod k)} \geq 0, \quad \forall\, h \in \{0, ..., k-1\}.$$



(a) X and Y (b) infeasible flow S and feasible flow $\tilde{S}$ in graph G''

Figure 4.5: Cycle X, subsequence Y, graph $G'' = (V, S)$ and the final flow $\tilde{S}$

□

This lemma gives us an edge e_1 in X such that any subsequence X' of X that starts with edge e_1 has the following property:

$$\sum_{e \in X'} \alpha(e) \geq 0\,. \tag{4.14}$$

Let $Y = Y_f \cup \overline{Y}_b$ be the longest subsequence X' such that adding the edges in Y_f to S_h and removing the edges in Y_b from it do not violate the budget constraint. We further remove edges from the start and end of Y that originate from or end at the source and target nodes, respectively. (Note that doing so does not affect the property in (4.14) since the value of α is zero for all such edges.) Let S denote the flow built by adding edges in Y_f to S_h and removing the edges in Y_b from it. Note that flow S satisfies the budget constraint due to the choice of Y. However, it may not attribute to a feasible flow as it may violate the flow conservation constraints in (4.7b). Therefore, we solve a min-cost max flow in graph $G'' = (V, S)$, with costs of edges set as C, to obtain a feasible flow $\tilde{S}$. This implies that $\tilde{S} \subseteq S$.

Let us implement this procedure for our small example. In Figure 4.5(a), we show the remaining cycle $X = X_2$ after removing cycle X_1. We further show a possible case for subsequence Y of X, which is represented by the edges in red, starting from node $9''$ and ending at node $11''$. Figure 4.5(b) demonstrates graph $G' = (V, S)$. Note that S does not satisfy the flow conservation for nodes $9''$ and $11''$. By solving a min-cost max flow in graph G, we obtain flow $\tilde{S}$ which can be obtained by removing the edges in red from flow S.

Finally, we claim that $\tilde{S}$ yields a worst-case optimality bound, and solves in strongly polynomial time.

Proposition 4.1. *Algorithm 4.2 provides a solution for problem (4.7) with worst-case optimality bound of* $3\,\mathsf{C}_{\max}$, *i.e.* $\mathsf{C}(\tilde{S}) \geq \mathsf{OPT} - 3\,\mathsf{C}_{\max}$.

Remark 4.1. *Algorithm 4.2 provides a solution for the problem in (4.7) in strongly polynomial time.*

Proof. As mentioned earlier, the value of β^* can be found in $O(|V|^4|E|^2)$ using the Megiddo's parametric search technique. Also, all simple cycles in graph G' can be found in $O(|V|^2)$ using the algorithm proposed by Johnson (1975), given the fact that at most $O(|V|)$ simple cycles can exist in this graph. After each loop, at least one cycle can be cancelled. Thus, the whole cycle-cancelling process takes $O(|V|^3)$. The Gasoline Lemma can be applied in $O(|V|)$, and flow $\tilde{S}$ can be found by solving a min-cost max flow in $O(|V|^3)$, given the fact $O(|E|) = O(|V|)$ in graph G''. Thus, the running time of Algorithm 4.2 is bounded by that of the parametric search and the result follows. □

We finalize this section by stating that one can turn Algorithm 4.2 into a PTAS by guessing the heaviest edges that will be in the optimal solution similar to the procedure described by Berger et al. (2011). However, in large-scale instances of our problem $\mathsf{C}_{\max}$ is sufficiently small compared to OPT, which implies that the optimality gap is reasonably low and we do not need such an expensive procedure.

4.4.4 A Budget-Balanced Variant of the Incentive Program

In Section 4.3, we assumed that the ridesharing system relies on an external budget of size B to incentivize the participants. In this subsection, we introduce a budget-balanced variant of our traveler incentive program whose funding comes from taxing the matches with positive adjusted gain. As such, let ϕ denote a flat tax rate that will be applied to any matched pair whose gain after adding subsidies is positive. In this case, we can replace the budget constraint in (4.7c) with the following constraint:

$$\sum_{e\in E} \Psi(e)\, x(e) \;\le \phi \sum_{e\in E} (\mathsf{W}(e) + \overline{\lambda}(e))\, x(e)\,. \tag{4.15}$$

In problem (4.7), we chose to maximize the difference between system's social welfare and allocated subsidy to ensure that every external dollar added to the system generates a positive return on investment. In the case of a budget-balanced system, however, the budget will be provided internally, and thus, we choose to maximize the system's social welfare. As a result, the after-tax social welfare of the budget-balanced system can be calculated:

$$(1-\phi) \sum_{e\in E} (\mathsf{W}(e) + \overline{\lambda}(e))\, x(e)\,, \tag{4.16}$$

Note that the pre-processing procedure in Algorithm 4.1 is still valid in this case, since taxing affects neither the spatio-temporal feasibility nor the individual rationality of any pair. Hence, given a constant value for ϕ, we can find the optimal solution to the new problem using the methodology proposed in the previous subsection with minor adjustments.

For a budget-balanced system, an important question that needs to be addressed is that "what would be an appropriate value for ϕ?". On the one hand, increasing the value of ϕ raises our available budget to subsidize more users, which leads to a higher matching rate and system-level social welfare. On the other hand, raising the tax rate adversely affects the social welfare of those participants who could be matched with lower values of ϕ. In what follows, we present a proposition that helps us answer this question.

Proposition 4.2. *The optimal flat tax rate that yields the maximum social welfare for the budget-balanced variant of the traveler incentive program can be found as:*

$$\phi_{\max} = \frac{\sum_{e \in E} \Psi(e)\, x_{\max}(e)}{\sum_{e \in E} (\mathsf{W}(e) + \bar{\lambda}(e))\, x_{\max}(e)}, \tag{4.17}$$

where $x_{\max}$ denotes the optimal solution to problem (4.7) *when Constraint* (4.7c) *is relaxed.*

Proof. Let us rewrite the objective function in (4.16) as:

$$\max \quad \sum_{e \in E} \mathsf{C}(e)\, x(e) + \Big(\sum_{e \in E} \Psi(e)\, x(e) - \phi \sum_{e \in E} (\mathsf{W}(e) + \bar{\lambda}(e))\, x(e) \Big).$$

Note that the statement inside the parentheses is always non-positive due to the budget constraint in (4.15). Therefore, its value will be maximized if Constraint (4.15) is binding. Thus, solution $(x_{\max}, \phi_{\max})$ maximizes both the statements inside and outside the parentheses, and the result follows. □

This proposition further implies that the budget-balanced variant of our program is no longer NP-hard, as stated in the following remark.

Remark 4.2. *The optimal solution to the budget-balanced variant of the traveler incentive program as a function of ϕ can be found in strongly polynomial time.*

Proof. Since problem (4.7) without Constraint (4.7c) can be modeled as a min-cost max flow and thus the worst-case running time complexity of finding $x_{\max}$ is $O(|V|^2|E|)$. Also, the pre-processing procedure has a time complexity of $O(|V|^2)$. Hence, the overall complexity of the budget-balanced variant of the incentive program is the same as that of the min-cost max flow problem which is fully polynomial in the input size and the result follows. □

4.5 Numerical Experiment

In this section we conduct a comprehensive set of numerical studies using the New York City taxi dataset (http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml) to evaluate the performance of the proposed traveler incentive program for a ridesharing system. In the next three subsections, we carefully define our dataset, parameter settings, and several performance metrics that help us quantify the impact of the proposed methodology. Afterwards, we present the result of various experiments.

All the experiments are implemented on a 3.50 GHz Intel Xeon machine with a 64-bit version of the Windows 10 operating system with 128.0 GB RAM. The data preparation and the subgradient optimization algorithm are coded in `Python 3.7`, and all the optimization problems are solved using `GUROBI 9.0`.

4.5.1 Dataset

In our numerical experiments, we assume that the proposed ridesharing system is practiced by the travelers in the Manhattan area of the New York City. Therefore, the road network of the Manhattan area is extracted from `Open Street Map`, which consists of 4500 nodes (stations) and 9800 transportation links. We further use the `Google Map API` to find the shortest-path travel time and driving distance between every pair of stations in the morning and evening. For the morning and evening trip information of the participants, we use the daily average of the historical trips in the New York City taxi dataset from Feb 1, 2016 to Feb 10, 2016 subject to the following assumptions and modifications:

- For a user that participates in the system only in the morning or evening peak hours, the origin and destination stations as well as the desired earliest departure time are obtained directly from the dataset.
- To construct the set of users who are interested in both the morning and evening trips, we pick the trip information of their morning trips from the dataset. For the evening trip, we randomly pick a desired earliest departure time from the evening peak hour period and flip the origin and destination of their morning trip to form the origin and destination of the evening trip.
- In order to obtain a tight time window for every trip, we generate a random number following a Normal distribution with mean of 5 minutes and standard deviation of 1 as the trip time flexibility. As a result, the desired latest arrival time of every trip can be set as the sum of desired earliest departure time, the shortest-path travel time between their origin and destination stations, and the trip time flexibility.
- We only consider trips whose shortest-path travel times are at least 10 minutes.
- We randomly generate the user's values of time and distance from Normal distributions with

means $\mu(\mathsf{F})$ and $\mu(\mathsf{H})$ and standard deviations $\mu(\mathsf{F})/5$ and $\mu(\mathsf{H})/5$, respectively.

4.5.2 Experiment setup and Parameter Settings

In our numerical experiments, we consider different scenarios of the ridesharing network with parameters defined as follows:

- $|N|$: the total number of participants in the system,
- $|R|/||N|$: the percentage of riders,
- $|N'''|/||N|$: the percentage of participants with both morning and evening trips,
- B : the total available budget for subsidy,
- $\mu(\mathsf{F})$: the mean of a user's time valuation,
- $\mu(\mathsf{H})$: the mean of a user's distance valuation,
- M and E : the morning and evening peak hour periods, respectively.

We define a base scenario whose parameters are shown in Table 4.1. We define other scenarios by changing the value of a single parameter in the base scenario at a time.

Table 4.1: The parameter settings for the base scenario.

$\lvert N\rvert$	$\frac{\lvert R\rvert}{\lvert N\rvert}$	$\frac{\lvert N'''\rvert}{\lvert N\rvert}$	B	$\mu(\mathsf{F})$	$\mu(\mathsf{H})$	M	E
	(%)	(%)	($)	($/min.)	($/mile)	(am/am)	(pm/pm)
6,000	50	50	1,000	0.35	3.0	[7:00, 10:00]	[5:00, 8:00]

Let us first describe how we generate the different sets, used in developing our methodology, based on the value of the first three parameters in Table 4.1. Consider the base scenario where the set of all users can be shown as $N = \{1, \ldots, 6000\}$ based on the first parameter. The set of all riders and drivers can be respectively shown as $R = \{1, \ldots, 3000\}$ and $D = \{3001, \ldots, 6000\}$ based on the value of 50% for the second parameter. Finally, the value of 50% for the third parameter allows us to find all other sets as $R' = \{1', \ldots, 2250'\}$, $R'' = \{751'', \ldots, 3000''\}$, $R''' = \{751, \ldots, 2250\}$, $D' = \{3001', \ldots, 5250'\}$, $D'' = \{3751'', \ldots, 6000''\}$, and $D''' = \{3751, \ldots, 5250\}$. We also set ϵ in the parametric search to 10^{-6}.

4.5.3 Performance Metrics

In order to quantify the performance of the proposed methodology, we introduce the following performance metrics:

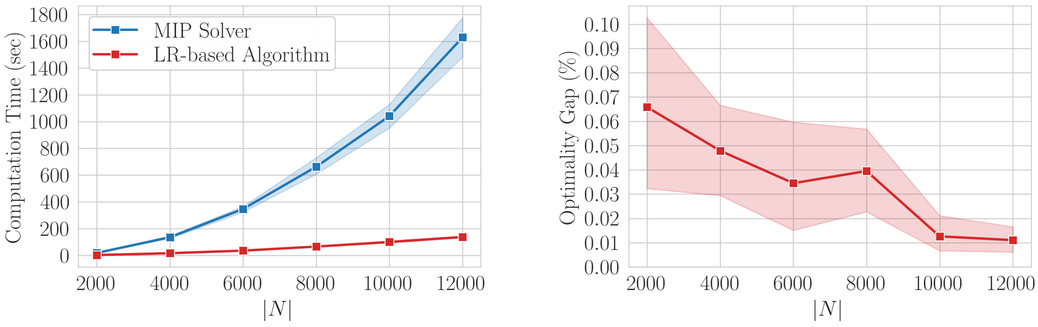
- Social Welfare (SW) = $\sum\limits_{e \in E} w^*(e)\, x^*(e)$,

- Subsidy Impact Rate (SIR) = $(\text{SW} - \text{SW without Subsidy})\Big/\big(\sum_{e\in E} \Psi(e)\, x^*(e)\big)$,
- Subsidized Matches Rate (SMR) = $100 \times \big(\sum_{e\in E} 1_{\Psi(e)>0}\big)\Big/\big(\sum_{e\in E} x^*(e)\big)$,
- Time Window Extension (TWE) = $\big(\sum_{n\in N'\cup N''} \gamma^*(n)\big)\Big/\big(\sum_{n\in N'\cup N''} 1_{\gamma^*(n)>0}\big)$,

where “Social Welfare without Subsidy” represents the negated total flow cost of the original min-cost flow problem in Section 4.3 without considering any subsidy.

4.5.4 The Performance of the LR-Based Solution Method

In this study, we show the merits of our proposed solution method in solving the budget-constrained min cost flow problem in (4.7). To this end, we compare the computation time and the quality of the solution obtained by the proposed LR-based method with that of solving the problem in (4.7) directly using a MIP solver. For this comparison, we generated 10 random instances of the problem using the parameters in the base scenario and applied both methods to solve them. We also repeated our experiments with the number of users set as 2000, 4000, 6000, 8000, 10000, and 12000. Figure 4.6(a) shows the computation time of both methods for different number of users averaged over 10 instances. Figure 4.6(b) shows the average optimality gap of our solution method for different number of users. The shaded region in both figures demonstrate the 95% confidence intervals of the average values. Figure 4.6(a) indicates that the MIP solver takes almost half an hour to solve the problem with 12000 customers while the proposed method in Algorithm 4.2 takes less than 3 minutes to find a near-optimal solution. Also, this figure clearly shows that the rate of increase in the computation time of the MIP solver is super-linear while the computation time of our solution method increases fairly linearly with the number of users, highlighting the scalability of the proposed method. Also, Figure 4.6(b) suggests that, in general, the quality of the solutions obtained using Algorithm 4.2 improves with the number of system participants. Finally, it is worth



(a) Computation time

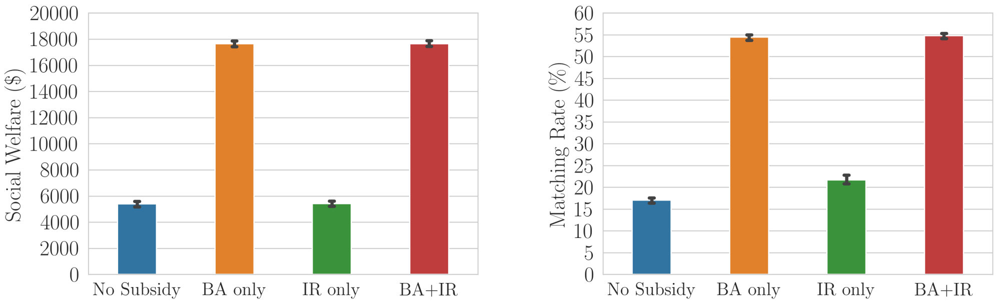
(b) Optimality gap

Figure 4.6: The comparison between the performance of the MIP solver and the LR-based Method

mentioning that we observed that the MIP solver uses different heuristics to solve the problem in the reported times. Without these heuristics, the pure Branch-and-Cut algorithm takes a considerable amount of memory and hours of computation time to provide any high quality solution to any instance of the problem with more than 6000 users.

4.5.5 The Impact of the Two Proposed Incentives

In this chapter, we consider two types of incentives, namely the behavioral adjustment and the individual rationality incentives. Here, we compare the performance of a ridesharing system with guaranteed-ride-back under four different cases of “no subsidy”, “only the BA incentive”, “only the IR incentive”, and “both the BA and IR incentives” using 10 different randomly-generated instances of the base scenario. Figure 4.7(a) demonstrates the average system-level social welfare over 10 runs for these four cases. Clearly, no subsidy yields the lowest values in social welfare. Subsidizing the system only using the BA incentive type shows a promising increase in the social welfare. However, this figure suggests that the IR incentive does not change the social welfare in the absence of the BA incentive, and improves the social welfare in the presence of the BA incentive only minusculely. There are two main reasons for this observation. First, the IR incentive cannot turn a spatio-temporally infeasible pair into a feasible pair by itself. That is the reason why in Algorithm 4.1 we first check the spatio-temporal feasibility of a pair, and only when a pair is spatio-temporally feasible do we inspect the individual rationality of that pair. Secondly, we stated in Section 4.4.2 that the IR incentive can be positive only if the served rider is in R''' and one of their trips in the morning or evening has negative original gain and the other has positive original gain, with sum of their original gains being positive. Otherwise, the IR incentive cannot add a higher value than its magnitude to the system. This fact highly narrows down the number of edges with positive IR incentive, and thus, its impact on the system-level social welfare diminishes. From the experiments



(a) Maximize the Social Welfare (b) Maximize the Matching Rate

Figure 4.7: The comparison between the performance of the ridesharing under different cases of subsidy for the objective of (a) maximizing the social welfare, and (b) maximizing the matching rate.

above, one may come to the conclusion that the IR incentive seems to be useless and should not be considered. In order to show the merit of such an incentive in promoting a ridesharing system, we change the objective function in problem (4.7) to maximizing the matching rate as follows:

$$\max \quad \frac{100 \times 2}{|N'| + |N''|} \sum_{e \in E} x(e) \,.$$

It is easy to show that under this objective, we can still use the pre-processing procedure in Algorithm 4.1. Figure 4.7(b) demonstrates the result of maximizing the matching rate under the four cases. This figure indicates that the IR incentive can increase the new objective, i.e., the matching rate, in a statistically significant manner compared to the no subsidy case, even in the absence of the BA incentive. However, this figure also suggests that the BA incentive has more benefits than the IR incentive to offer due to the first reason discussed above. This figure also shows that in the presence of the BA incentive, introducing the IR incentive does not add considerable value.

4.5.6 The Impact of Tax Rate in the Budget-Balanced Incentive Program

In Section 4.4.4, we introduced a budget-balanced variant of our incentive program in which the required subsidy is collected internally through taxing the matches with positive adjusted gain. We further showed that the optimal flat tax rate $\phi_{\max}$ can be found analytically. In this experiment, we demonstrate the empirical results for an arbitrary instance of the base scenario for different values of $0 \leq \phi \leq 100$. Figure 4.8 shows the after-tax social welfare in (4.16) against the tax rate. This figure clearly shows that the social welfare of the budget-balanced variant of the incentive program increases sub-linearly until it reaches $\phi_{\max} = 16.87\%$, and then it starts to collapse until it gets to zero at 100%. Overall, this figure suggests that subsidizing the system internally can significantly increase its social welfare from less than \$6000 (without subsidy) to more than \$20,000. Also, it indicates that even a small tax rate as low as 1% can double the system's social welfare.

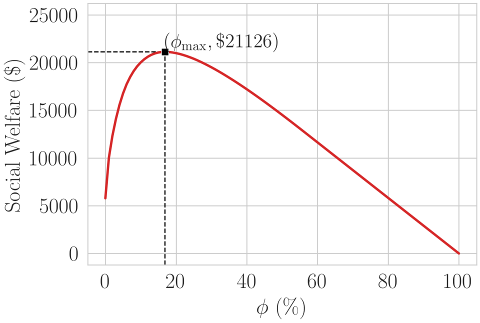
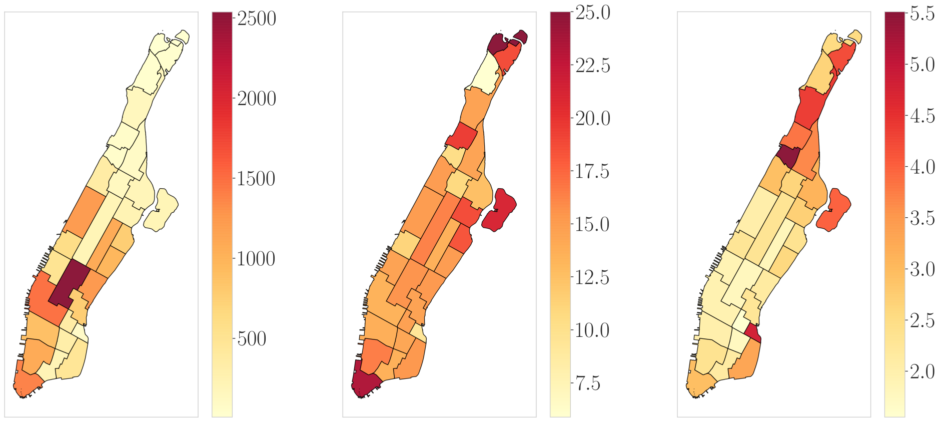


Figure 4.8: The social welfare for different values of tax rate in a budget-balanced incentive program

4.5.7 The Distribution of the BA Incentive Based on Trip Origins and Destinations

In this experiment, we are interested in learning the characteristics of the users who are more likely to change their travel behavior, and thus, receive the BA incentive. To this end, we investigate the chance of receiving the BA incentive as well as the average amount of subsidy based on the origin and destination geo-coordinates of participants. Figure 4.9 presents three heat maps that help us analyze the distribution of the BA incentive among participants for an arbitrary instance of the base scenario. Note that due to having a large number of stations, we aggregate them into larger zones that represent different neighborhoods in the Manhattan area. Figure 4.9(a) shows the number of trips in the morning and evening whose origin or destination stations falls within each zone. This figure clearly shows that most trips originate/end form/at the lower Manhattan area, more specifically in the neighborhood where the Time Square is located. Figure 4.9(b) shows the percentage of the subsidized users for each neighborhood. This figure suggests that those users who start or end their trips in less popular zones are more likely to receive the BA incentive. This is not surprising, because the chances of finding users with compatible trips is lower for such trips. Therefore, they are more likely to extend their time windows to match with other users. Figure 4.9(c) shows the average amount of the BA incentive per number of users that received such an incentive for each neighborhood. This figure clearly indicates that as the popularity of a neighborhood decreases, the trips whose origin and/or destination stations fall into that region should expand their time windows by a larger extent. Overall, this experiment suggests that those users who are located in the areas that are poorly supported by public transit can benefit more from



(a) Frequency of the Origin and Destination Zones

(b) Relative Frequency of Subsidized Zones

(c) Average Subsidy per Subsidized Users of Zones

Figure 4.9: The heatmap of the origin and destination zones

the introduction of such an incentive program.

4.5.8 A Dynamic Implementation of the Incentive Program

Throughout this chapter, we considered a static ridesharing system for which all trip information for a given day is known prior to solving the ride-matching problem. This requires all users to announce their trips before the start of the morning period. In this experiment, we relax this requirement by considering a dynamic system in which users are allowed to announce their trip information shortly before their desired earliest departure times. Let $\mathsf{L}(n)$ denotes the trip announcement time for user $n \in N$. Note that we assume that users in N''' announce both of their trips at the same time in the morning. In order to solve the dynamic ride-matching problem, we adopt the rolling horizon approach that re-optimizes the static ride-matching problem in short time intervals, say every 1 minute, using all the available trip information collected by the onset of each interval (see e.g. Agatz et al. (2011)). There are two important considerations for applying the rolling horizon approach in the existence of an incentive program: (*i*) how to allocate the total budget B to the re-optimization intervals, and (*ii*) when to finalize the matches between riders and drivers. Let us denote the re-optimization points (i.e., the starting times of the re-optimization intervals) by set $P = P' \cup P''$, where P' and P'' respectively denote the re-optimization points in the morning and evening. It is easy show that the pre-processing (with a minor adjustment) and solution method can still be applied in this case. More specifically, we only need to add the following constraint to the problem in (4.4):

$$t(n') \geq p + 1\,, \qquad\qquad \forall\, n \in \{d, r\}\,. \tag{4.18}$$

This constraint ensures that the trip's start time is always after the end of the re-optimization interval. In what follows, we introduce two different policies for each of these considerations and evaluate their impacts on the system's social welfare.

For allocating the budget to the intervals, we consider two policies: (1) a naive policy that allows for using the entire available budget in each interval, and (2) dividing the total budget among intervals in a two-step process. In the first step, we divide the total budget between the morning and the evening intervals according to a pre-determined rate $\pi \in (0, 1)$. In the second step, we divide $\pi\,\mathsf{B}$ dollars uniformly among the intervals in P', and $(1 - \pi)\,\mathsf{B}$ dollars uniformly among the intervals in P''. Also, we roll over the unused budget in each interval to the next interval.

For finalizing the matches found after solving the problem in (4.7) at each time $p \in P$, we consider two policies: (1) we finalize the matches for users as soon as they got matched in a re-optimization interval, (2) we postpone the matching finalization for any user as much as possible. More precisely, in the second policy we only finalize the matching between pairs of riders and

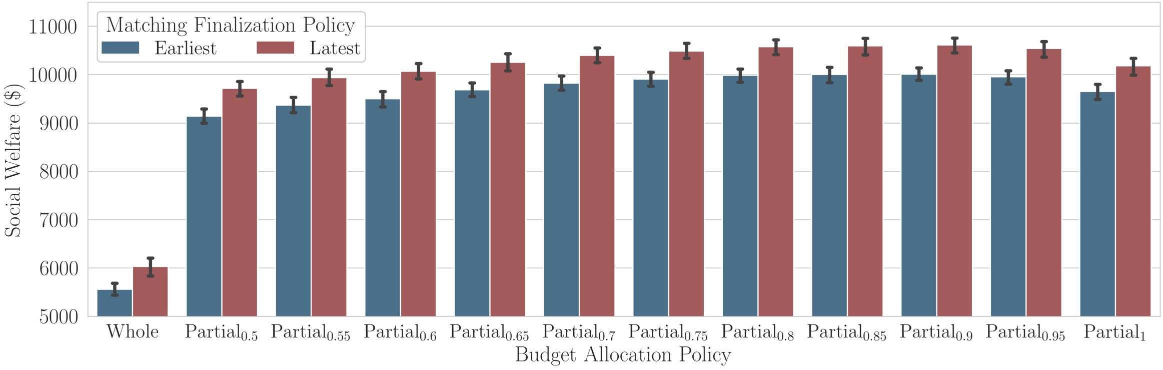


Figure 4.10: The social welfare of a dynamic incentive program under different policies for allocating budget and finalizing matches

drivers who cannot get matched together in the next re-optimization interval, or when the amount of subsidy allocated to them needs to increase in the next interval. This can be done by solving the problem in (4.4) once for $p + 1$, and once for $p + 2$ at every re-optimization point $p \in P$.

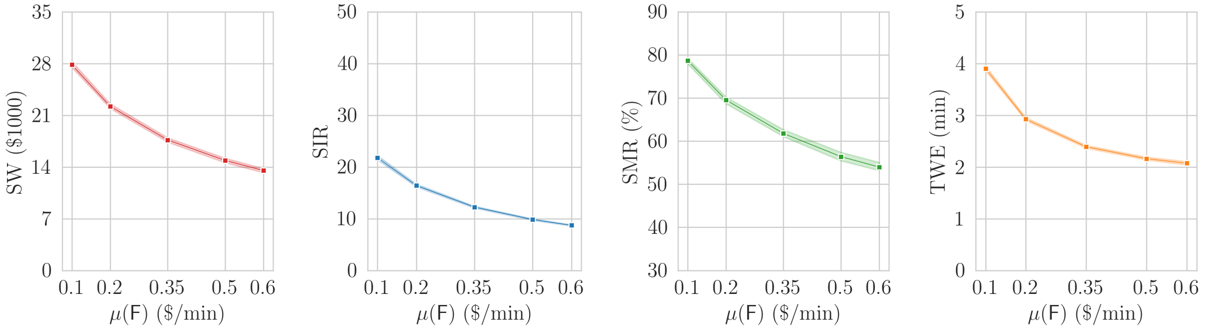
Figure 4.10 demonstrates the results of applying these policies to 10 randomly-generated instances of the base scenario. For each scenario, the trip announcement times are calculated by subtracting a random number following the Normal distribution with mean 5 and standard deviation of 1 from the corresponding desired earliest departure times. Moreover, for partial budget allocation policy, we consider different values of π from 0.5 to 1. The scenario “whole” in this figure corresponds to the naive budget allocation strategy of using as much of the budget as desired in the earlier intervals, and only using the roll-over budget in the subsequent ones. Scenarios “Partial$_\pi$” indicate the results for the fraction π of the total budget being allocated to the morning period. Under each scenario, the blue and red bars indicate the total social welfare for the two match fixing policies of finalizing the matches as early and late as possible, respectively.

Figure 4.10 clearly indicates that the combination of a partial budget allocation and latest time to finalize the matches yields the highest system-level social welfare. This is due to the following facts: under the Whole budget policy, the available budget will be depleted in the first few re-optimization points, which results in a huge opportunity cost of not being able to subsidize the ride-matching problems in the following intervals. Additionally, postponing the matching finalization to the latest possible interval enables us to obtain more information on future trips, and thus, find possibly more beneficial matches (with higher welfare and lower subsidy) for each user. This figure further demonstrates that the highest social welfare can be obtained by setting π to 0.8, which is slightly higher than the rate of the morning trips in the system (0.75 in the base scenario) in the base scenario. One possible explanation for this observation is that all the available budget will not be used in each interval, and hence, letting π be slightly larger than the rate of the morning trips in the system can help us distribute the total available budget more uniformly between the morning and evening trips; i.e., if there is no need for the extra 5% of budget, it will be rolled over to the next interval, but not providing the possibility of using this budget may lead to an opportunity cost.

4.5.9 Sensitivity Analysis

In this subsection, we consider different scenarios, by changing one parameter of the base scenario at a time, to analyze the sensitivity of the performance metrics (see Section 4.5.3) on parameter values (see Section 4.5.2). The results of this analysis are plotted in Figures 4.11, 4.12, C.1, C.2, C.3, C.4, and C.5. Each plot shows the average and the 95% CI of the corresponding performance metric over 10 instances. In what follows, we analyze the results of Figures 4.11 and 4.12. The rest of the figures and their interpretations are presented in Appendix C.3.

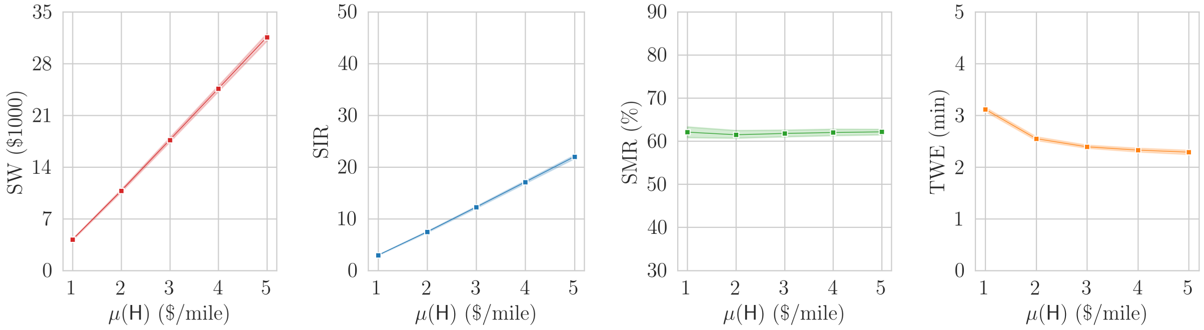
In the base scenario, we assumed that the user's value of time, $\mu(\mathsf{F})$, has a mean of \$0.35 per minute. In order to study the relation between different values of this parameter and the performance metrics, we change the value to \$0.1, \$0.2, \$0.5 and \$0.6 per minute. The results are presented in Figure 4.11. In this figure, we observe that all the performance metrics decrease sub-linearly with an increase in the value of $\mu(\mathsf{F})$. This is not surprising because the amount of subsidy allocated to a user to change their travel behavior is a linear function of their value of time. Therefore, as a user's value of time increases, the system has to compensate them with more subsidy to change their travel behavior, i.e. expand their time window.



(a) Social Welfare (b) Subsidy Impact Rate (c) Subsidized Matches Rate (d) Time Window Extension

Figure 4.11: Impact of average value of time

Next, we consider the impact of changes in the average user's value of distance, denoted by $\mu(\mathsf{H})$, on the performance metrics. Figure 4.12 presents the results of changing this value from \$3 per mile in the base scenario to \$1, \$2, \$4 and \$5 per mile. Figure 4.12(c) shows no significant changes in the percentage of subsidized matches, which is due to the fact that the BA incentive allocated to users does not get affected by the value of this parameter, and that most of the available budget is spent on the BA incentive (see Section 4.5.5). However, Figures 4.12(a) and 4.12(b) indicate that both the social welfare and the subsidy impact rate increase linearly as a result of an increase in the value of $\mu(\mathsf{H})$. The main reason behind these increasing trends is that an increase in the valuation of distance linearly increases the savings due to matching, which leads to a higher number of individually rational matches. Also, this increase in the number of individually rational matches allows users to find matches that not only have higher gains but also require smaller changes in their time windows, as shown in Figure 4.12(d).



(a) Social Welfare (b) Subsidy Impact Rate (c) Subsidized Matches Rate (d) Time Window Extension

Figure 4.12: Impact of average value of distance

4.6 Conclusion and Future Work

In this chapter, we consider a community-based ridesharing system with guaranteed-ride-back for commuters. To promote such a ridesharing system among commuters, we introduce a traveler incentive program (TIP) that offers two types of incentives, namely the behavioral adjustment (BA) and the individual rationality (IR) incentives. We formulate the joint problem of matching, scheduling, and incentive allocation as a mixed integer nonlinear program (MINLP). Using a pre-processing procedure and by utilizing linear programming, we reduce the MINLP problem to a budget-constrained min-cost flow problem. For solving large-scale instances of this problem, we devise a polynomial-time Lagrangian Relaxation-based algorithm, and obtain a worst-case optimality bound for its performance. We further introduce a budget-balanced variant of the incentive program that does not require external budget. Finally, we conduct several experiments using the New York City taxi dataset to evaluate different aspects of the TIP and the solution methodology. Our findings from the results of these experiments can be summarized as follows:

1. The proposed TIP considerably increases the social welfare by tripling that of the system without subsidy. Also, every dollar spent on subsidy increases the social welfare by 12 dollars

and matching rate by more than 40%.

2. The proposed LR-based solution method significantly reduces the computational effort needed to solve the large-scale instances of the problem when compared to a MIP solver. Also, the relative optimality gap of the LR method does not exceed 0.15% for any large-scale instance, and converges to zero as the size of the problem increases.
3. When the objective of the system is to maximize social welfare, the impact of the IR incentive is negligible. However, we show that this type of incentive can be effective when the system operator aims to maximize the matching rate. Nonetheless, when BA incentives are introduced, adding IR incentives to the incentive basket improves neither the social welfare nor the matching rate in a statistically significant manner. This signifies the importance of using limited resources to change travel behavior, rather than providing direct monetary subsidies to travelers.
4. Those users who start or end their trips in regions that are less populated and/or farther away from the business districts are more likely to receive subsidies. In addition, these users experience larger increases in their time windows.
5. For dynamic implementation of the incentive program, it is more beneficial to separate the budget for the morning and evening peak hours and distribute the available budget among re-optimization intervals.

There are at least three directions to extend the work in this chapter. First, the fare paid by riders and the compensation received by drivers depend on their personal information (the valuations of time and distance), which we assumed could be estimated from their historical information. However, this information can also be directly solicited from the participants upon registration. In this case, it is essential to design a mechanism that ensures incentive compatibility, i.e.,the individuals cannot benefit by not being truthful in reporting their private information (i.e., gaming the system). For future work, we consider designing a mechanism to determine the fare paid/received by riders/drivers based on their marginal contributions to social welfare (i.e., fairness), and also guarantees incentive compatibility (i.e., truthfulness). Moreover, this chapter assumes the simplest form of ride-matching problem, i.e., the one-to-one ride-matching problem. As a result of this assumption, we observe that the IR incentive does not contribute much to improving social welfare. However, in case of allowing multiple riders or the possibility of transfers between vehicles, we might observe significant impacts by both types of incentives. Finally, this chapter considers a simple linear model to incorporate the effect of subsidies on a user's utility. It is interesting to consider more complicated forms of subsidization in ridesharing, as Fang et al. (2020) shows that the multi-threshold subsidy programs are more effective in practice. Additionally, participants' valuations of expanding their time windows from right or left (leaving later or earlier, respectively) could be different, especially for commuter trips.

CHAPTER 5

Role Assignment Stability in Peer-to-Peer Ridesharing Markets

5.1 Introduction

Considering the significant growth in the world's population in recent years, transportation researchers are incessantly searching for new methods to handle the resultant increase in transportation demand. These methods can be roughly divided into two major categories: Transportation System Management (TSM) and Transportation Demand Management (TDM). TSM approaches, in general, aim to increase the capacity of transportation supply, which requires large capital investments on infrastructure along with considerable analyses to ensure economic efficiency in the long run. TDM, on the other hand, provides innovative approaches to reduce, shift, or repack transportation demand while taking infrastructure limitations into account as an input to the system.

As a result of continued progress in advanced communication technologies, TDM techniques have become more compelling to both researchers and practitioners due to several advantages over TSM strategies. Ridesharing is an example of such techniques that has been proved to be successful in increasing the utilization rate of current transportation supply. Ridesharing can result in reducing the number of single-occupancy vehicles and, to some extent, improving the utilization of available seat capacity (Regue et al. (2016); Nam et al. (2018); Lloret-Batlle et al. (2017b); Masoud et al. (2017b); Zhang et al. (2020); Masoud and Jayakrishnan (2016)). This advantage combined with other significant benefits, such as reduction in fuel consumption and congestion, has resulted in an increase of interest to invest in the ridesharing companies such as Scoop in recent years.

Peer-to-peer (P2P) ridesharing, as a special case of ridesharing systems, has emerged in the past decades due to the enabling of the sharing economy, rooted from prevalence of high-speed internet, online payment systems, and smartphone platforms that enable easy and fast communication between individuals. A P2P ridesharing system provides a platform to match a group of riders to their peer drivers while taking some spatiotemporal constraints into account. Ridematching over large-scale transportation networks is a common problem in all ridesharing systems, where a system operator attempts to pair riders and drivers with compatible trips based on a pre-determined objective (Masoud and Jayakrishnan (2017c,b); Lloret-Batlle et al. (2017b)).

In the economics literature, ridematching has been traditionally thought of as a two-sided matching market. In such markets agents belong to one of the two mutually exclusive sets with predefined roles, e.g., transportation service providers and travelers. Since agents may have different tastes in partners, finding an effective matching strategy that takes into account individual preferences is crucial. In the absence of such strategies, agents may have incentives to look for better coalitions. In order to address this issue, game theory suggests the concept of "stable matching". In stable matching, no two unmatched agents would mutually prefer one another, or being alone, to staying with their current partners.

In spite of many similarities between P2P ridesharing and other ridesharing or ride-sourcing systems, there is a subtle difference that makes stability analysis different in P2P ridesharing. In a P2P ridesharing platform, each participant who owns a vehicle can make a trip under the rider or the driver role. Therefore, the conventional assumption of two mutually exclusive sets with predefined roles in the traditional two-sided markets does not hold anymore. In this chapter, we define a new type of market, where (for a subset of agents), the market intermediator decides the set to which an agent belongs; that is, the two sets of agents are not mutually exclusive. Moreover, it is worth mentioning that the concept of stability usually arises in environments in which agents have high level of interactions with each other such that the utilities and preferences of agents are (or can become) common knowledge among them. This is due to the fact that otherwise agents may have no information based on which they deviate from the matching offered by the system. As such, in this chapter, we focus on a static version of P2P ridesharing in which users have recurrent pre-determined trips (e.g., home to/from work commuting trips), and the information regarding their trips may become common knowledge among all registered users due to the long-run interactions between them. Given these assumptions, the objective of this chapter is to present a joint matching and payment system, combined with role assignment, for a static P2P ridesharing system that aims to maximize social welfare. Since maximizing social welfare does not necessarily give rise to stable outcomes, agents may not accept these outcomes and try to establish a more profitable coalition on their own. To avoid such scenarios, we study the stability of such a matching market, and present a mathematical formulation that finds a stable outcome for yes-instances and a stable outcome with minimum amount of subsidy injected to the system for no-instances.

The rest of this chapter can be summarized as the following: In Section 5.2 we review the related work in the economics and transportation literature. The mathematical formulation of the problem and its corresponding market game are introduced in Section 5.3. In Section 5.4, we introduce a mathematical formulation that finds a stable outcome with minimum amount of subsidy. Further on in this section, an approximate solution approach based on Lagrangian Relaxation algorithm is proposed for finding a stable outcome. The result of a numerical experiment and some interesting aspects of our proposed model are discussed in Section 5.5. We conclude our findings and provide some suggestions for future research in Section 5.6.

5.2 Literature Review

In this section, we first review the concept of stability in matching markets with ordinal and cardinal preferences. Next, we present a number of major economic studies in ridesharing. We finalize this section with a clear statement about the contribution of this chapter.

5.2.1 Stable Matching

The problem of matching in a two-sided market is a well-studied area in economics. A two-sided market consists of two mutually exclusive sets of agents who negotiate for a bi-lateral trade. The intention of one-to-one matching in such markets is to bring together pairs of agents from these two sets to form a coalition. A matching is rational if no matched individual prefers singlehood to being matched with their designated partner. Stable matching is an individually rational matching that has no pair of agents that would rather leave their current partners in order to match together. The problem of finding a stable matching in two-sided markets with ordinal preferences is widely known as the "marriage problem" introduced by Gale and Shapley (1962). They proved that the marriage problem with complete lists of preferences always has a stable matching, and proposed an efficient algorithm called "deferred-acceptance", which provides an optimal stable matching. Later, Vate (1989) proposed a linear program to solve the optimal stable marriage problem. A marriage game with a cardinal list of preferences is called an assignment game, in which utilities can be transferred between agents. Shapley and Shubik (1971) were the first to study stable matching in an assignment game. They showed that the set of stable outcomes are the dual values of the solutions to the optimal assignment problem, and that there is a connection between the geometric structure of the core and the optimal solutions in each side of the market. Roth and Sotomayor (1996) showed the similarity of conclusions that can be drawn for the marriage problem with strict preferences, and an assignment game based on the same assumptions. More precisely, they prescribed a unified treatment that proved to have common results in both the discrete and continuous cases of a two-sided matching market by following a simple assumption that the core coincides with the core defined by weak domination. In this chapter, we modify their definition of a stable outcome in accordance with the structure of our proposed market. For more detailed information on two-sided matching and its various applications, see Roth and Sotomayor (1992); Knuth (1997), and the references therein.

In their seminal work, Gale and Shapley (1962) also introduced the concept of stability in one-sided markets, widely known as the “roommate problem”. They demonstrated through an example that stable matching does not always exist in one-sided markets. Later, Irving (1985) proposed an efficient algorithm that for any instance of the roommate problem either finds a stable matching or determines that one does not exist. Abeledo and Rothblum (1994) used linear algebra to analyze different characteristics of general stable matching.

The roommate problem with transferable utilities (TU) was first introduced by Eriksson and Karlander (2001). They found a necessary condition for odd-cycles in non-bipartite graphs to guarantee the existence of a stable matching. More specifically, a graph has a stable outcome if in every odd-cycle of the graph, there exist a feasible match whose weight is greater than or equal to the sum of the weights of all unmatched edges in that cycle. They also showed the similarity between their results and those of the roommate problem with non-transferable utilities. Our proposed market will be shown to be a special case of the TU roommate problem, and hence it inherits all the properties of such problems. Since the roommate problem does not necessarily admit a stable outcome, a lot of effort has been made in recent years to either develop near-stable outcomes (see e.g., Abraham et al. (2005); Biró et al. (2014, 2016)) or convert the graph into a stable one with minimal modifications (see e.g., Ito et al. (2017); Ahmadian et al. (2018); Chandrasekaran et al. (2019)). In Section 5.4, we discuss a number of these approaches in further detail and show their application in the context of ridesharing.

5.2.2 Economics in Ridesharing

In the past decade, several economic studies have been conducted in the area of ridesharing. These studies focus on applying useful economic techniques to increase the likelihood of user participation in a shared system. In dynamic ridesharing systems, Kleiner et al. (2011) suggest using the sealed-bid second price auction to find the matching and pricing solutions. In their proposed mechanism riders bid for their rides, while drivers rank riders based on the disutility of incurring detours, and pick the ones that are most profitable for them. This paper provides some insights about the trade-off between maximum cardinality matching and minimum system-wide vehicle kilometers travelled. Shen et al. (2016b) recommend an integrated online posted-price mechanism for the problem of matching and pricing in an autonomous mobility-on-demand ridesharing system. Their mechanism works in three steps: fare estimation, pickup assignment, and payment calculation. The results of the simulation studies conducted in this paper confirm the superiority of their method over the optimal assignment solution and the offline, auction-based mechanisms. Zhao et al. (2014) design a VCG mechanism for ridesharing problems that selects the role of commuters, finds matches, and determines prices. They discuss different techniques such as fixed payment or double reserved price schemes to address the budget deficit resulting from the mechanism. Thaithatkul et al. (2015)

study the effect of passenger preferences on the rate of sharing a taxi ride. They formulate the passenger matching as a stable roommate problem, and consider personality and its steadiness (variance) as the factors that affect user preferences. Lloret-Batlle et al. (2017a) propose a matching allocation combined with a VCG-based pricing mechanism for a P2P ridesharing system that guarantees ride-backs for the matched riders. In their pricing mechanism, they propose a novel approach to determine the role of participants based on a multi-parameter reserved price. They further prove that the introduction of a reserved price can help ensure budget balancedness. A P2P ride exchange mechanism is proposed by Masoud et al. (2017a) to increase the matching rate and customer retention in a ridesharing system. They design a mechanism that provides riders with the opportunity to purchase other riders' itineraries in the intention of encouraging a high number of drivers to participate, and increasing the service rate in the system.

Wang et al. (2017) study the effects of incorporating stable matching on the system-wide performance of dynamic ridesharing systems as a marriage problem. They present different mathematical formulations to find stable or near-stable outcomes in the context of ridesharing systems. Their experiments indicate that including stability does not significantly compromise the optimal system-level vehicle-miles savings. Rasulkhani and Chow (2019) consider the problem of assigning riders to drivers' route where routes have a finite capacity. They formulate this problem as a many-to-one assignment game and develop solution methods to find user-optimal and operator-optimal stable outcomes. Peng et al. (2020) design a stable payment system for the one-to-one matching in ridesharing with the addition of equity constraints. They further introduce a compensation system that helps riders incentivize drivers for taking longer detours. Finally, we should emphasize the fact that the notion of role flexibility in P2P ridesharing is not new (see e.g., Agatz et al. (2010); Amey (2011); Tafreshian and Masoud (2020a)). However, all these papers merely focus on operational aspects of the ride-matching problem in such systems (see e.g., Tafreshian et al. (2020)). Also, it is worth mentioning that one must not confuse the concept of role assignment discussed in this chapter with that of ridesharing user equilibrium (RUE) (see e.g., Xu et al. (2015); Di et al. (2018)), where the role of travellers are determined at equilibrium based on traffic congestion.

5.2.3 Our Contribution

This chapter contributes to the literature in the following ways: (*i*) this study is the first to introduce the concept of stable matching with side payments in P2P ridesharing systems; (*ii*) we incorporate role assignment in the mathematical formulation of one-to-one matching in ridesharing systems, and investigate the stability of the one-to-one ride-matching problem with role flexibility; (*iii*) we introduce a mixed integer program that finds the minimum amount of subsidy required for stabilizing the system; (*iv*) we propose a decomposition algorithm based on Lagrangian Relaxation that finds bounded near-optimal solutions for the proposed formulation in large-scale networks; and (*v*) we show the relationship between two problems that serve as treatments for no-instances of the roommate problem with side payments, namely the minimum subsidy problem and the minimum blocking value problem.

5.3 Problem Statement

In this section, we describe how the joint role assignment matching problem that arises in a static P2P ridesharing system can be modeled as a roommate problem with side payments. Let $V = \{v\}$ denote the set of participants (users) in the system during the study horizon. Set V is a subset of all registered commuters with recurrent trips in our ridesharing system. We assume that each participant v provides the following information regarding their trip:

- o_v: the origin location (e.g., in longitude and latitude),
- d_v: the destination location (e.g., in longitude and latitude),
- t_v: the earliest acceptable time to depart from the origin location,
- q_v: the latest acceptable time to arrive at the destination location

In this chapter, the simplest form of matching, known as one-to-one matching, is considered. In one-to-one matching, each participant can only provide/receive a ride to/from at most one participant. Thus, the problem can be represented by a weighted, directed non-bipartite graph $G = (V, E)$, where $E = \{e\}$ is the set of weighted edges. An edge $e = (u, v)$ exists in graph G if participant u is capable of giving a ride to participant v while completing their own trip within their time window. Let $\tau(A, B)$ and $\delta(A, B)$ denote the shortest-path travel time and driving distance between locations A and B, respectively. Thus, an edge e exists in graph G if equations (5.1a) and (5.1b) are satisfied simultaneously. Equations (5.1a) and (5.1b) respectively ensure that participants v and u can complete their trips within their specified time windows.

$$\max\{t_u + \tau(o_u, o_v)\ ,\ t_v\} + \tau(o_v, d_v) \leq q_v\ , \tag{5.1a}$$

$$\max\{t_u + \tau(o_u, o_v)\ ,\ t_v\} + \tau(o_v, d_v) + \tau(d_v, d_u) \leq q_u\ . \tag{5.1b}$$

In graph G, there is a positive weight, w_{uv}, associated with every edge $e = (u, v)$. In the game theory literature, w_{uv} is called the potential for profit of matching u to v (Shapley and Shubik, 1971). Since the value of a match between a rider and driver is proportional to the potential mileage saving due to participant u providing a ride to participant v, we define w_{uv} as the vehicle miles traveled (VMT) saving in the context of our problem. If we assume that the participants have a quasi-linear utility function, then the potential gain of matching u to v is defined as:

$$w_{uv} = \max\{0, \eta_{uv} + \theta_{uv}\}\,, \tag{5.2}$$

where η_{uv} and θ_{uv} denote respectively the valuations of users u and v when user u provides a ride to user v. These valuations can be found as the difference between the cost of sharing the ride and the cost of driving alone. If user u provides a ride to user v, without loss of generality, we use Equations (5.3a) and (5.3b) to define their valuations:

$$\eta_{uv} = \delta(o_u, d_u) - \big(\delta(o_u, o_v) + \delta(o_v, d_v) + \delta(d_v, d_u)\big)\,, \tag{5.3a}$$

$$\theta_{uv} = \delta(o_v, d_v)\,. \tag{5.3b}$$

Equation (5.3a) states that the valuation of user u is the difference between the distance of their own trip and the distance traveled (i.e., the negative of the detour user u has to incur from their shortest path to serve user v). Equation (5.3b) defines the valuation of user v simply as the driving distance of their trip. It is worth noting that θ_{uv} in (5.3b) is independent of u, which indicates that the valuation of rider v does not depend on the driver with whom they are sharing a ride. The converse, however, is not true. A driver's valuation is dependent upon who they serve, since different riders would require different detours.

5.3.1 Matching for P2P Ridesharing Systems with Role Flexibility

The goal of the ridesharing system is to find the maximum social welfare, i.e., the maximum weighted general matching in graph G. In what follows, we present a binary program to formulate the general matching problem. The decision variable x_{uv} is a binary variable that holds the value 1 if user u provides a ride to user v, and the value 0 otherwise. Note that the value of this variable indicates whether there is a match between users u and v and the order of users in the index indicates their roles. The objective function in (5.4a) maximizes the social welfare, which equates to maximizing the weighted sum of matches. Constraint (5.4b) ensures that each user can be matched with at most one other user. Constraints (5.4c) enforce the decision variables x_{uv} to take binary values.

$$\max \quad \sum_{(u,v)\in E} w_{uv}\, x_{uv} \tag{5.4a}$$

$$\text{s.t.} \quad \sum_{\substack{u\in V:\\ (v,u)\in E}} x_{vu} + \sum_{\substack{u\in V:\\ (u,v)\in E}} x_{uv} \leq 1\,, \qquad \forall\, v \in V\,, \tag{5.4b}$$

$$x_{uv} \in \{0, 1\}\,, \qquad \forall\, (u, v) \in E\,. \tag{5.4c}$$

Next, we present a proposition that will prove useful as it helps us reduce the number of edges in graph G.

Proposition 5.1. *In the optimal solution of the primal formulation (5.4), if* $(u, v) \in E$*,* $(v, u) \in E$*, and* $x_{uv} + x_{vu} = 1$*, then we have* $x^*_{uv} = 1$ *if* $w_{uv} > w_{vu}$ *and* $x^*_{vu} = 1$ *if* $w_{uv} < w_{vu}$*.*

Proof. We prove this proposition by contradiction. Let us assume $w_{uv} < w_{vu}$, but $x^*_{uv} = 1$ in the optimal solution. This means that user u gives a ride to user v. Given all other variables fixed at their optimal value, we switch the roles of u and v, i.e., let user u serve as a driver and user v serve as a rider which leads us to change $x^*_{uv} = 0$, $x^*_{vu} = 1$. Considering all these changes, it is easy to check that all constraints are still satisfied, indicating that the solution remains feasible. However, the objective function increases by the value of $(w_{vu} - w_{uv}) > 0$, which contradicts with the assumption of optimal solution. The same result follows if we assume $w_{uv} > w_{vu}$, but $x^*_{vu} = 1$ in the optimal solution. ♠

This proposition simply states that if users u and v are determined to share a ride together in the optimal solution and both can provide a ride to the other one, they select their roles in the most profitable way. Based on the result of Proposition 5.1, we can safely remove the edges with lower weights between every two vertices with two edges in set E, without compromising the optimal solution of the problem in model (5.4). Doing so, we can avoid the role assignments that are not part of the optimal solution.

From the mathematical formulation in (5.4), it follows that the market game in a P2P ridesharing system with flexible roles is a special case of stable matching in one-sided markets with side payments, widely known as the roommate problem with transferable utilities (TU). The P2P ridesharing market requires an additional step to select the roles of the agents after they are matched together. In the next subsection, we explain how this game can be applied to a P2P ridesharing system.

5.3.2 Market Game for P2P Ridesharing Systems with Role Flexibility

Let $p \in \mathbb{R}$ denote the payment transfer between users u and v upon forming a coalition in our proposed market; that is, we assume that user u gives a ride to user v, and user v pays p units of money to user u in return. We further suppose that λ_{uv} and γ_{uv} are the utility functions of driving and riding, respectively, defined as in the following:

$$\lambda_{uv}(p) = \eta_{uv} + p \,, \tag{5.5a}$$

$$\gamma_{uv}(p) = \theta_{uv} + p \,, \tag{5.5b}$$

where a positive value for p indicates receiving a monetary payment and a negative value for p indicates transferring a payment. Note that the unit of p is the same as that of valuations, which is vehicle miles in this chapter. However, one can use an appropriate constant, say \$1/mile, to convert all payments and valuations from vehicle miles to dollars. Following these definitions, a payment p from v to u is *individually rational* if the two following inequalities hold simultaneously:

$$\lambda_{uv}(p) \geq \kappa_u \,, \tag{5.6a}$$

$$\gamma_{uv}(-p) \geq \chi_v \,, \tag{5.6b}$$

where κ_u and χ_v represent the reservation utilities of user u as a rider and user v as driver, respectively. In the context of P2P ridesharing, these values can be translated to the utility of not sharing the ride, i.e., not being matched. We assume that a participant who is not being matched uses the next best (utility maximizing) option in their choice set. Without loss of generality, we assume that all participants have access to private vehicles, and normalize κ_u and χ_v to be equal to 0.

Based on the above definitions, the market game of P2P ridesharing with flexible roles can be fully represented by the four-tuple $(V, \Pi, \Lambda, \Gamma)$, where Π, Λ, and Γ symbolize the set of payments, driving utility functions, and riding utility functions, respectively. In the following steps, we provide some necessary definitions for presenting the stability of matching in this market game.

Definition 5.1. *An outcome is a pair of matching x and a vector of payoffs $\rho \in \mathbb{R}^{|V|}$, denoted by (x, ρ).*

Note that the payoff of user $v \in V$, denoted by ρ_v, represents the monetary profit of user v from participating in the ridesharing system.

Definition 5.2. *The outcome (x, ρ) is feasible if there is a set of payments $p_{uv} \in \Pi \subseteq \mathbb{R}^{|V| \times |V|}$, only defined if x is a feasible matching, such that:*

i) $x_{uv} = 1 \rightarrow \rho_u = \lambda(p_{uv})$, *and* $\rho_v = \gamma(-p_{uv})$,

ii) $\sum\limits_{\substack{u \in V: \\ (v,u) \in E}} x_{vu} + \sum\limits_{\substack{u \in V: \\ (u,v) \in E}} x_{uv} = 0 \rightarrow \rho_v = 0$,

iii) $\rho_v \geq 0$.

Now, we are prepared to define the stability for the P2P ridesharing systems with flexible roles.

Definition 5.3. *The feasible outcome (x, ρ) is stable if there is no pair $(u, v) \in E$ with $x_{uv} = 0$ and $p_{uv} \in \Pi$ such that $\lambda_{uv}(-p_{uv}) > \rho_u$ and $\gamma_{uv}(p_{uv}) > \rho_v$.*

Note that Definition 3 simply states that the feasible outcome is stable only if there is no blocking pair for which both u and v can improve their payoffs by leaving their current partners, and forming a coalition together. According to this definition, a feasible outcome is stable if it satisfies the following set of necessary conditions:

$$\rho_u + \rho_v = w_{uv}\,, \qquad \forall\,(u,v) \in E \text{ if } x_{uv} = 1\,, \tag{5.7a}$$

$$\rho_u + \rho_v \geq w_{uv}\,, \qquad \forall\,(u,v) \in E \text{ if } x_{uv} = 0\,, \tag{5.7b}$$

$$\rho_v \geq 0\,, \qquad \forall\, v \in V\,, \tag{5.7c}$$

where (5.7a) ensures that the sum of payoffs is equal to the potential of profit for matched partners, (5.7b) enforces that there is no unmatched pair that can dominate the current payoffs, and (5.7b) guarantees the individual rationality of the payoffs. This result is consistent with that of the roommate problem with transferable utilities, stated in Eriksson and Karlander (2001). They show that the extreme points of the core of the integral TU roommate games are half-integral, i.e., $x_{uv} \in \{0, 0.5, 1\}$, indicating that a fractional stable matching always exists in the core of such problems. They also concluded that an integral stable matching may not always exist. They further introduce a sufficient condition for having stable integral matches. The sufficient condition requires that odd cycles meet a certain criterion such that the graph can be reduced to a bipartite graph. Based on our experience with large-scale P2P ridesharing systems with random preferences, this condition is very unlikely to be satisfied in real-life scenarios.

5.3.3 Treatments for Unstable P2P Ridesharing Systems with Role Flexibility

As mentioned in the previous subsection, P2P ridesharing systems with role flexibility may not admit stable outcomes. Thus, we have to develop a solution method that yields near-stable or stable outcomes with the help of interventionist policies. In the literature of game theory, researchers have defined a number of alternative problems to find *as stable as possible* solutions for those instances of the roommate problem with side payments that do not have a stable solution. Biró et al. (2012) introduces two such problems as the "Minimum Blocking Pairs" problem and the "Minimum Blocking Value" problem. Before introducing these problems, we first define a blocking pair and its value for a feasible outcome in our market.

Definition 5.4. *An edge $(u,v) \in E$ is a blocking pair for feasible outcome (x,ρ) with $x_{uv} = 0$ if $\rho_u + \rho_v < w_{uv}$.*

Definition 5.5. *For a feasible outcome (x,ρ), the blocking value of any edge $(u,v) \in E$, denoted by b_{uv}, can be defined as* $\max\{0, w_{uv} - \rho_u - \rho_v\}$.

Note that a stable outcome has no blocking pairs, and thus, the total blocking values is equal to 0. Based on these definitions, Biró et al. (2012) introduces the Minimum Blocking Pairs as a problem that aims to find the outcome that yields the smallest number of blocking pairs. Whereas, the Minimum Blocking Value is defined as the problem of finding the outcome that results in the smallest sum of blocking values. They further prove that both problems are NP-complete. The main drawback of these problems is that one cannot control the amount of deviation for each block. In other words, there may exist some pairs whose blocking values are rather large, which indicates that the users in those pairs have large incentives to deviate from their current partners and form new coalitions (including the singleton coalitions).

In recent years, graph theorists have devised a number of graph stabilizer techniques that minimally modifies the structure of a general graph to make sure it yields a stable outcome (Bock et al., 2015; Ito et al., 2017; Ahmadian et al., 2018; Chandrasekaran et al., 2019). These modifications can be in the form of adding/deleting edges/vertices, and adding to edge weights. Among these treatments, adding weights to edges serves as a meaningful treatment in the context of our problem. More precisely, we can assume that the ridesharing system subsidizes some pairs by increasing their potential gain to make sure that no user deviates from proposed outcome. Chandrasekaran et al. (2019) show that this problem, hereinafter referred to as the "Minimum Subsidy" problem, is NP-hard, and devise an approximation algorithm for the special case of graphs with uniform edge weights. To the best of our knowledge, there is no algorithm in the literature to tackle the problem of stabilizing a graph with non-uniform edge weights through changing the weights. In the next section, we provide a mathematical formulation of this problem, and present a solution methodology based on a Lagrangian Relaxation decomposition algorithm. We further show the relationship between this problem and the Minimum Blocking Value problem. It is worth noting that the main shortcoming of this stabilizing a graph through introducing subsidies resides in the fact that the payment system is clearly not budget-balanced. In Section 5.5, we show through numerical studies that the minimum amount of subsidy required is very small compared to the stable social welfare.

5.4 Solution Methodology

In this section, we first formulate the Minimum Subsidy problem as a mixed integer program. Next, we devise a solution methodology to efficiently solve the large-scale instances of this problem. Next, we provide some important properties of this problem. We finalize this section by presenting a small example that helps us describe the concepts and methods discussed in this chapter.

5.4.1 Mathematical Formulation

As mentioned earlier, the goal of the Minimum Subsidy problem is to find a stable outcome by minimally increasing the weight of edges. Therefore, we define s_{uv} as the amount of external subsidy for coalition of users u and v, i.e., edge $(u, v) \in E$, that increases the gain of their coalition. In other words, we are looking for a vector $s \in \mathbb{R}^{|E|}$ such that the core of the graph $G = (V, E)$ with weights $(w + s)$ is nonempty. The mathematical formulation of this problem is presented in (5.8). This formulation is motivated by the fact that there must be no integrality gap between the primal problem and the dual linear relaxation of the maximum weighted matching in graph G with new weights (Eriksson and Karlander, 2001; Chandrasekaran, 2017). In this formulation, we have three sets of variables: (*i*) matching $x \in \{0, 1\}^{|E|}$, (*ii*) payoffs $\rho \in \mathbb{R}^{|V|}$, and (*iii*) subsidy value $s \in \mathbb{R}^{|E|}$. The objective in (5.8a) minimizes the total amount of subsidy over all edges in graph G. Constraint (5.8b) makes certain that x is a feasible matching. Constraint (5.8c) ensures that the set of payoffs, with the addition of subsidy to potential gains, does not result in a blocking pair. Constraint (5.8d) ensures that the sum of payoffs is equal to the sum of social welfare and total amount of subsidy. Constraint (5.8e) makes sure that an unmatched user will have a zero payoff. Note that the set of Constraints in (5.8c), (5.8d), and (5.8e) combined satisfy the necessary conditions for stability stated in (5.7). Finally, Constraints (5.8f), (5.8g), and (5.8h) are the integrality and non-negativity constraints.

$$
\begin{aligned}
\min \quad & \sum_{(u,v)\in E} s_{uv} && && (5.8a)\\
\text{s.t.} \quad & \sum_{\substack{u\in V:\\(u,v)\in E}} x_{uv} + \sum_{\substack{u\in V:\\(v,u)\in E}} x_{vu} \le 1\,, && \forall\, v \in V\,, && (5.8b)\\
& \rho_u + \rho_v \ge w_{uv} + s_{uv}\,, && \forall\, (u,v) \in E\,, && (5.8c)\\
& \sum_{v\in V} \rho_v = \sum_{(u,v)\in E} (w_{uv} + s_{uv})\, x_{uv}\,, && && (5.8d)\\
& \rho_v \Big(1 - \sum_{\substack{u\in V:\\(u,v)\in E}} x_{uv} + \sum_{\substack{u\in V:\\(v,u)\in E}} x_{vu}\Big) = 0\,, && \forall\, v \in V\,, && (5.8e)\\
& x_{uv} \in \{0,1\}\,, && \forall\, (u,v) \in E && (5.8f)\\
& \rho_v \ge 0\,, && \forall\, v \in V\,, && (5.8g)\\
& s_{uv} \ge 0\,, && \forall\, (u,v) \in E\,. && (5.8h)
\end{aligned}
$$

Unfortunately, this formulation involves nonlinear constraints, which make the problem very hard to solve. In what follows, we present a number of claims that help us reformulate the problem in (5.8) as a mixed integer program and further tighten the solution space.

Claim 5.1. *An unmatched edge receives no subsidy in the optimal solution.*

Proof. Let us denote the set of matched edges in the optimal solution by E_M, i.e., $E_M = \{(u, v) \in E | x^*_{uv} = 1\}$, and the set of vertices incident to these edges by V_M. From Constraint (5.8e), we know that $\rho^*_v = 0$ for all v in $V \setminus V_M$. Thus, Constraint (5.8d) can be rewritten as $\sum_{v \in V_M} \rho^*_v = \sum_{(u,v) \in E_M}\sum (w_{uv} + s^*_{uv})$, which implies that the subsidies of unmatched edges in $E \setminus E_M$ do not impact the optimal payoffs, and thus, the objective function enforces them to be zero. □

Claim 5.2. *The optimal payoff of a user cannot exceed the maximum weight of its incident edges in* G.

Proof. Let ω_u be the maximum weight of the edges that are incident to u in G. We want to show that ρ^*_u cannot be greater than ω_u. If user u is unmatched, Constraint (5.8e) enforces ρ^*_u to be zero, which is obviously not greater than ω_u. Now, suppose that user u is matched with user v, ρ^*_u is the optimal payoff of user u, and the payoff is greater than ω_u. Without loss of generality assume that u gives a ride to v. We prove by contradiction. We know that Constraints in (5.8c), (5.8d), and (5.8e) imply the necessary conditions in (5.7). Also, we know from Claim 5.1 that the subsidies on unmatched edges are zeros. As a result, the following conditions must hold for user u:

$$\rho^*_u + \rho^*_v = w_{uv} + s^*_{uv} \tag{5.9a}$$

$$\rho^*_u + \rho^*_{v'} \geq w_{uv'}\,, \qquad \forall\ (u, v') \in E\,, \tag{5.9b}$$

$$\rho^*_{v''} + \rho^*_u \geq w_{v''u}\,, \qquad \forall\ (v'', u) \in E\,. \tag{5.9c}$$

Suppose a payoff $\hat{\rho}_u$ such that $\omega_u < \hat{\rho}_u < \rho^*_u$. It is obvious that replacing ρ^*_u by $\hat{\rho}_u$ satisfies Constraints (5.9b) and (5.9c). However, replacing ρ^*_u by $r\hat{h}o_u$ in (5.9a) enforces s^*_{uv} to decrease. As a result, we were able to decrease the objective function, which contradicts the optimality of ρ^*_u and the result follows. ♠

Claim 5.3. *The following is a valid inequality for the problem in (5.8):*

$$s_{uv} \leq (\omega_u + \omega_v - w_{uv})\, x_{uv} \qquad \forall (u, v) \in E\,, \tag{5.10a}$$

where ω_u and ω_v the maximum weights of edges incident to u and v, respectively.

Proof. From Claim 5.1, if $x_{uv} = 0$ then s_{uv}, and if $x_{uv} = 1$ then based on Constraint (5.8c) and Claim 5.2, we have:

$$s_{uv} \leq \rho_u + \rho_v - w_{uv} \leq \omega_u + \omega_v - w_{uv}\,. \tag{5.11a}$$

□

Claim 5.4. *Constraint (5.8e) can be replaced by the valid inequality introduced in Claim 5.3 and the following valid inequality:*

$$\rho_v \leq \sum_{\substack{u \in V: \\ (u,v) \in E}} (w_{uv}\, x_{uv} + s_{uv}) + \sum_{\substack{u \in V: \\ (v,u) \in E}} (w_{vu} x_{vu} + s_{vu}) \qquad \forall\, v \in V\,. \tag{5.12a}$$

Proof. As mentioned earlier, if user u is matched with user v, then the sum of the payoffs for users u and v is equal to the sum of the gain of their match and the added subsidy. Thus, the statement in the right hand side with the help of valid inequalities in Claim 5.3 enforces the payoff of user v to be zero if unexposed by the matching, and thus, this inequality can replace Constraint (5.8e). In addition, the proposed inequality provides an upper bound on ρ_v if user v is matched with another user. □

Based on the claims above, the mixed integer program for the Minimum Subsidy problem can be presented as in (5.13). The conventional approach for solving such programs is the well-known method of Branch & Bound, which is computationally expensive, and cannot be applied in large-scale ridesharing systems. However, special structure of this problem enables us to propose an efficient Lagrangian Relaxation (LR) algorithm. In the next subsection, we elaborate on our proposed methodology.

$$\min \quad \sum_{(u,v) \in E} s_{uv} \tag{5.13a}$$

$$\text{s.t.} \quad \sum_{\substack{u \in V: \\ (u,v) \in E}} x_{uv} + \sum_{\substack{u \in V: \\ (v,u) \in E}} x_{vu} \leq 1\,, \qquad \forall\, v \in V\,, \tag{5.13b}$$

$$\rho_u + \rho_v \geq w_{uv} + s_{uv}\,, \qquad \forall\, (u,v) \in E\,, \tag{5.13c}$$

$$\sum_{(u,v) \in E} s_{uv} = \sum_{v \in V} \rho_v - \sum_{(u,v) \in E} w_{uv}\, x_{uv}\,, \tag{5.13d}$$

$$s_{uv} \leq (\omega_u + \omega_v - w_{uv})\, x_{uv}\,, \qquad \forall\, (u,v) \in E\,, \tag{5.13e}$$

$$\rho_v \leq \sum_{\substack{u \in V: \\ (u,v) \in E}} (w_{uv}\, x_{uv} + s_{uv}) + \sum_{\substack{u \in V: \\ (v,u) \in E}} (w_{vu}\, x_{vu} + s_{vu})\,, \qquad \forall\, v \in V\,, \tag{5.13f}$$

$$x_{uv} \in \{0,1\}\,, \qquad \forall\, (u,v) \in E\,, \tag{5.13g}$$

$$\rho_v \geq 0\,, \qquad \forall\, v \in V\,, \tag{5.13h}$$

$$s_{uv} \geq 0\,, \qquad \forall\, (u,v) \in E\,. \tag{5.13i}$$

5.4.2 Lagrangian Relaxation

In the literature of combinatorial optimization, LR is a well-known technique to provide high-quality lower bounds (in minimization problems) for large-scale mixed integer programs that cannot be solved either to optimality or ever within the available time. Due to the reasons that will become clear later, we relax constraints (5.13e) and (5.13f), and add their violations to the objective function along with Lagrange multipliers $\phi \in \mathbb{R}_+^E$, and $\psi \in \mathbb{R}_+^V$, respectively. Also, we replace the objective in (5.13a) with the right-hand-side term in Constraint (5.13d). As a result, the LR problem, denoted by $\mathsf{LR}(\phi, \psi)$, can be formulated as:

$$
\begin{aligned}
\mathsf{LR}(\phi,\psi): \quad \min \quad & \sum_{v\in V} \rho_v - \sum_{(u,v)\in E} w_{uv}\, x_{uv} + \sum_{(u,v)\in E} \phi_{uv}\Big(s_{uv} - (\omega_u + \omega_v - w_{uv})\, x_{uv}\Big) + \\
& \sum_{v\in V} \psi_v\Big(\rho_v - \sum_{\substack{u\in V:\\ (u,v)\in E}} (w_{uv}x_{vu} + s_{uv}) - \sum_{\substack{u\in V:\\ (v,u)\in E}} (w_{vu}x_{vu} + s_{vu})\Big) \qquad (5.14a) \\
\text{s.t.} \quad & (5.13b) - (5.13c) \text{ and } (5.13g) - (5.13i)
\end{aligned}
$$

Later we will show that this problem can be further decomposed into two independent sub-problems that can be solved efficiently for constant multipliers at each iteration. The algorithm starts by fixing the Lagrange multipliers to pre-determined values. Next, the LR problem will be solved, and the value of the multipliers will be updated using subgradient optimization. The LR problem provides a lower bound, whereas an upper bound is obtained by solving another problem that yields a feasible solution for the total amount of subsidy. The algorithm iterates between the solution of the sub-problems and the upper problem until either the relative gap between the upper and the lower bounds meets the convergence criterion, or the solution time reaches its maximum threshold.

The advantages of the proposed LR-based method are three-fold: (*i*) the sub-problems can be solved in polynomial time, (*ii*) at each iteration we can easily obtain a feasible solution, and therefore once a time threshold is reached, the best solution can be easily retrieved, and (*iii*), the solution quickly converges to the neighborhood of the global optimal solution, which is highly favorable in cases where we need to provide a fast, high quality, but not necessarily optimal, solution. In the next subsections, we elaborate on the sub-problems, the subgradient optimization, and the upper bound problem, respectively. The complete LR algorithm is outlined in Algorithm 5.1.

5.4.3 LR Sub-problems

By rewriting the objective function in (5.14), we can obtain the two following sub-problems:

$$\begin{aligned}
\mathsf{LR}_1(\phi,\psi): \quad & \max \quad \sum_{(u,v)\in E} \Big((1-\phi_{uv}+\psi_u+\psi_v)w_{uv} + \phi_{uv}(\omega_u+\omega_v)\Big)\, x_{uv} && (5.15\text{a})\\
& \text{s.t.} \quad (5.13b) \text{ and } (5.13g) \\
\mathsf{LR}_2(\phi,\psi): \quad & \min \quad \sum_{v\in V}(1+\psi_v)\rho_v + \sum\sum_{(u,v)\in E} (\phi_{uv}-\psi_u-\psi_v)\, s_{uv} && (5.15\text{b})\\
& \text{s.t.} \quad (5.13c), (5.13h) \text{ and } (5.13i)
\end{aligned}$$

The first sub-problem is a maximum weighted matching in graph G with weight of edge $(u, v) \in E$ defined as $w'_{uv} = (1-\phi_{uv}+\psi_u+\psi_v)w_{uv}+\phi_{uv}(\omega_u+\omega_v)$. In the literature, several exact methods have been proposed to find the optimal weighted matching in general graphs. The *Blossom* algorithm, proposed by Edmonds (1965), is one of the most well-known methods for solving such problems in polynomial time, with worst-case computational complexity of $O(|V|^2|E|)$. Using these algorithms, one can obtain the optimal matching reasonably fast for relatively large problems. The second sub-problem is a linear problem that can be solved quickly in polynomial time using the well-known ellipsoid method.

5.4.4 The Subgradient Optimization Problem

Let us define the Lagrangian Dual problem, denoted by LD, as follows:

$$\begin{aligned}
\max \quad & \mathsf{LR}(\phi,\psi) && && (5.16\text{a})\\
\text{s.t.} \quad & \phi_{uv} \geq 0\,, && \forall\ (u,v)\in E\,, && (5.16\text{b})\\
& \psi_v \geq 0\,, && \forall\ v \in V\,. && (5.16\text{c})
\end{aligned}$$

We adopt the subgradient optimization method to approximately solve this problem in the hope of getting high-quality lower bounds for the original problem in (5.13). The subgradient method is an iterative algorithm that starts with pre-determined values (ϕ^0, ψ^0) for multipliers. Next, it repeatedly solves the LR problem and then updates the Lagrange multipliers for the next iteration using the violation of relaxed constraints in the current LR problem. Let $(\bar{x}^k, \bar{\rho}^k, \bar{s}^k)$ be the optimal solution of $\mathsf{LR}(\phi^k, \psi^k)$ at iteration k ($k \geq 0$). The multiplier for iteration $(k+1)$ can be updated as:

$$\begin{aligned}
\phi_{uv}^{(k+1)} &= \max\{0, \phi_{uv}^k + \vartheta\ \Phi_{uv}^k\}\,, && \forall\ (u,v)\in E\,, && (5.17\text{a})\\
\psi_v^{(k+1)} &= \max\{0, \psi_v^k + \vartheta\ \Psi_v^k\}\,, && \forall\ v\in V\,, && (5.17\text{b})
\end{aligned}$$

where Φ_{uv}^k and Ψ_v^k are the slacks of the relaxed constraints that can be computed as:

$$\Phi_{uv}^k = \bar{s}_{uv}^k - (\omega_u + \omega_v - w_{uv})\ \bar{x}_{uv}^k\,, \qquad \forall\ (u,v) \in E\,, \tag{5.18a}$$

$$\Psi_v^k = \bar{\rho}_v^k - \sum_{\substack{u\in V:\\(u,v)\in E}} (w_{uv}\bar{x}_{vu}^k + \bar{s}_{uv}^k) - \sum_{\substack{u\in V:\\(v,u)\in E}} (w_{vu}\bar{x}_{vu}^k + \bar{s}_{vu}^k\overline{)}\,, \qquad \forall\ v \in V\,, \tag{5.18b}$$

and ϑ denotes the step size which can be found as:

$$\vartheta = \frac{\alpha(\mathsf{BUB} - \mathsf{LB}^k)}{\sum\limits_{(u,v)\in E} (\Phi_{uv}^k)^2 + \sum_{v\in V}(\Psi_v^k)^2}\,, \tag{5.19a}$$

where LB^k denotes the optimal objective of the LR problem at iteration k, and BUB represents the best upper bound of the original problem found so far (i.e., in the previous k subgradient iterations). We use BLB to refer to the best lower bound found so far. The parameter α is a scale parameter that will be initially set to a value in $(0, 2]$, and will be cut by half whenever BLB does not improve for a given number of consecutive, say m, iterations. Let T be the maximum time budget and ε be a positive small scalar. The algorithm terminates if the relative gap between the upper and the lower bounds is less than ε, or the running time reaches T.

5.4.5 The Upper bound Problem

Although the minimum subsidy problem is an NP-hard problem, it is easy to show that the restricted minimum subsidy problem, i.e., the minimum subsidy problem given a fixed matching, can be formulated as a linear problem, and hence can be solved efficiently in a timely manner. In our LR algorithm, we solve the upper bound problem to obtain a feasible solution that provides an upper bound on the original problem. The goal of this problem is to find a set of feasible payoffs $\bar{\bar{\rho}}$ and subsidies $\bar{\bar{s}}$ for a given fixed matching $\bar{x}$ such that the sum of subsidies are minimized.

In our problem, this matching is provided by the LR problem from the current iteration. The linear program in (5.20) represents the mathematical formulation of this problem. Let $E_M = \{(u,v) \in E : x_{uv} = 1\}$ be the set of matched edges and $V_M = \{v \in V : \sum_{u\in V:(v,u)\in E} x_{vu} + \sum_{u\in V:(u,v)\in E} x_{uv} = 1\}$ be the set of matched vertices. The objective in (5.20a) is to minimize the total amount of subsidy over the edges in E. Constraint (5.20b) ensures that the users' payoffs in a matched edge is equal to the sum of the gain from that match and the added subsidy, and thus there exist a feasible payment between them. The necessary condition for stability over unmatched edges is enforced by Constraint (5.20c). Constraint (5.20d) make sure that unmatched users have 0 payoffs. Finally, Constraint

(5.20e) is the individual rationality constraint for matched users.

$$
\begin{array}{llll}
\mathsf{UB}(\bar{x}): & \min & \displaystyle\sum_{(u,v)\in E_M} s_{uv} & \qquad (5.20a)\\
& \text{s.t.} & \rho_u + \rho_v = w_{uv} + s_{uv}\,, \quad \forall\ (u,v) \in M\,, & \qquad (5.20b)\\
& & \rho_u + \rho_v \geq w_{uv}\,, \quad \forall\ (u,v) \in E \setminus E_M\,, & \qquad (5.20c)\\
& & \rho_v = 0\,, \quad \forall\ v \in V \setminus V_m\,, & \qquad (5.20d)\\
& & \rho_v \geq 0\,, \quad \forall\ v \in V_M\,. & \qquad (5.20e)
\end{array}
$$

Algorithm 5.1: A LR algorithm for the P2P ridematching problem with flexible roles

```
Input: Initial values of Lagrange multipliers ($(\phi^0, \psi^0)$), initial value of scale parameter $\alpha$, an
       acceptable optimality gap $\varepsilon \geq 0$, the maximum number of iterations with no improvement $m$,
       and the maximum allowed time budget $T$.
Output: A stable outcome $(x^*, \rho^*)$, and optimal subsidies $s^*$.
1  Initialize $k \leftarrow 0$; $\mathsf{BLB} \leftarrow -\infty$; $\mathsf{BUB} \leftarrow +\infty$; $\ell \leftarrow 0$ ;
2  repeat
3      Solve sub-problems $\mathsf{LR}_1(\phi^k, \psi^k)$ and $\mathsf{LR}_2(\phi^k, \psi^k)$ to obtain $(\bar{x}^k, \bar{\rho}^k, \bar{s}^k)$ ;
4      $\mathsf{LB}^k \leftarrow$ optimal objective of $\mathsf{LR}_2(\phi^k, \psi^k)$ − optimal objective of $\mathsf{LR}_1(\phi^k, \psi^k)$ ;
5      if $\mathsf{LB}^k > \mathsf{BLB}$ then
6          Update best lower bound $\mathsf{BLB} \leftarrow \mathsf{LB}^k$ ;
7          $\ell \leftarrow 0$ ;
8          Update best multipliers $\bar{\phi} \leftarrow \phi^k$; $\bar{\psi} \leftarrow \psi^k$ ;
9      else if $\ell \geq m$ then
10         $\ell \leftarrow 0$ ;
11         Update the scale parameter $\alpha \leftarrow \frac{\alpha}{2}$ ;
12         Restore best multipliers $\phi^k \leftarrow \bar{\phi}$; $\psi^k \leftarrow \bar{\psi}$ ;
13         Go to line 3  ;
14     else
15         $\ell \leftarrow \ell + 1$ ;
16     Solve upper bound problem $\mathsf{UB}(\bar{x}^k)$ to find feasible payoffs $\bar{\bar{\rho}}^k$ and subsidies $\bar{\bar{s}}^k$ ;
17     $\mathsf{UB}^k \leftarrow$ optimal objective of $\mathsf{UB}(\bar{x}^k)$ ;
18     if $\mathsf{UB}^k < \mathsf{BUB}$ then
19         Update best upper bound $\mathsf{BUB} \leftarrow \mathsf{UB}^k$;
20         Update optimal solutions $x^* \leftarrow \bar{x}^k$; $\rho^* \leftarrow \bar{\bar{\rho}}^k$; $s^* \leftarrow \bar{\bar{s}}^k$ ;
21     Update Lagrange multipliers $(\psi^k, \psi^k)$ as in (5.17) ;
22 until ($\frac{\mathsf{BUB}-\mathsf{BLB}}{\mathsf{BUB}} \leq \varepsilon$) or (elapsed time > $T$)
```

5.4.6 Properties of the Minimum Subsidy Problem

In this section, we present two propositions that provide upper and lower bounds on the optimal solution of the minimum subsidy problem.

Proposition 5.2. *The integrality gap in (5.4) yields a lower bound for the minimum subsidy problem in (5.13).*

Proof. Let ρ_v denote the dual variable corresponding to Constraint (5.4b) in the linear relaxation of the problem in (5.4). The dual problem associated with the linear relaxation of the problem in (5.4) can be found as:

$$\min \quad \sum_{v \in V} \rho_v \tag{5.21a}$$

$$\text{s.t.} \quad \rho_u + \rho_v \geq w_{uv}\,, \qquad \forall\,(u,v) \in E\,. \tag{5.21b}$$

$$\rho_u \geq 0, \qquad \forall\, v \in V\,. \tag{5.21c}$$

Now, if we let both ϕ's and ψ's be zeros in the LR problem, the sub-problems LR_1 and LR_2 reduce to the problems in (5.4) and (5.21), respectively. Thus, the objective function of the LR problem in this case yields the integrality gap between the fractional and integral maximum weighted matching. Since the LR problem provides a lower bound for the original problem in (5.13), the result follows. □

The result of this proposition indicates that setting (ϕ^0, ψ^0) to $(0, 0)$ provides a good starting point for the subgradient algorithm in (5.1). More precisely, setting (ϕ^0, ψ^0) to $(0, 0)$ ensures that we find the stable outcome for yes-instances of our problem in the first iteration of the subgradient optimization. Moreover, it starts the algorithm with a positive lower bound for no-instances of the problem which is better than the obvious lower bound of zero.

Proposition 5.3. *The optimal minimum blocking value yields an upper bound for the minimum subsidy problem in (5.13).*

Proof. As mentioned earlier, the goal of the minimum blocking value problem is to find an outcome that yields the minimum sum of blocking values. This problem can be formulated as a mixed integer program presented in (D.1). Now, let (x^*, ρ^*, b^*) denote the optimal solution of the problem in (D.1). Further, let E_M and V_M be the set of matched edges and the set of vertices incident to these edges based on x^*. We first show that this solution yields a feasible solution for the problem in (5.13). From Constraints (D.1c)-(D.1e), we can infer the following results regarding ρ^*:

$$\rho^*_u + \rho^*_v = w_{uv}\,, \qquad \forall\,(u,v) \in E_M\,, \tag{5.22a}$$

$$\rho^*_u + \rho^*_v + b^*_{uv} \geq w_{uv}\,, \qquad \forall\,(u,v) \in E \setminus E_M\,, \tag{5.22b}$$

$$\rho^*_v = 0\,, \qquad \forall\, v \in V \setminus V_M\,. \tag{5.22c}$$

Now, let us emphasize that x^* is obviously a feasible matching for the problem in (5.13). However, ρ^* is not necessarily a feasible payoff vector for the problem in (5.13) unless b^* is a vector of zeros. We show how one can construct feasible vectors of $\hat{\rho}$ and $\hat{s}$ from ρ^* and b^*. To construct $\hat{\rho}$, initially set their values to ρ^*. Next, for each unmatched edge in (5.22b) whose blocking value is greater than zero, add the whole blocking value to $\hat{\rho}$ of one of the unexposed users of that block (it does not matter which one we pick as long as the user is unexposed). Note that at least one of the users incident to an edge is always unexposed by the matching x^*. This is due to the fact that otherwise we can always decrease the blocking value of an edge by matching the two unexposed users incident to it, which contradicts the optimality of solution (x^*, ρ^*, b^*). After doing this process for all edges with a positive blocking value, we have the following results regarding the new payoffs, $\hat{\rho}$:

$$\hat{\rho}_u + \hat{\rho}_v \geq w_{uv}\,, \qquad \forall\,(u,v) \in E_M\,, \tag{5.23a}$$

$$\hat{\rho}_u + \hat{\rho}_v \geq w_{uv}\,, \qquad \forall\,(u,v) \in E \setminus E_M\,, \tag{5.23b}$$

$$\hat{\rho}_v = 0\,, \qquad \forall\, v \in V_M\,. \tag{5.23c}$$

If we let $\hat{s}_{uv}$ be equal to $\hat{\rho_u} + \hat{\rho_v} - w_{uv}$ for edges in E_M, and zeros for the edges in $E \setminus E_M$, we can replace (5.23a) with the following equation:

$$\hat{\rho}_u + \hat{\rho}_v = w_{uv} + \hat{s}_{uv}\,, \qquad \forall\,(u,v) \in E_M\,. \tag{5.24}$$

Consequently, the Constraints in (5.24), (5.23b), and (5.23c) are equivalent to those of the upper bound problem in (5.20), and hence, the solution $(x^*, \hat{\rho}, \hat{s})$ is feasible for the problem in (5.13). Now, let us show that the optimal blocking value is an upper bound for the optimal total amount of subsidy. From the process described above, we infer that:

$$\sum_{v\in V} \hat{\rho}_v = \sum_{v\in V} \rho^*_v + \sum_{(u,v)\in E} b^*_{uv}\,. \tag{5.25}$$

Thus, based on the definition of the objective function in (5.13a), and Constraint (D.1d), we have:

$$\begin{aligned}\sum_{(u,v)\in E} s^*_{uv} &\leq \sum_{(u,v)\in E} \hat{s}_{uv} \\ &= \sum_{v\in V} \hat{\rho}_v - \sum_{(u,v)\in E} w_{uv} x^*_{uv}\end{aligned}$$

$$= \sum_{v\in V}\rho_v^* - \sum_{(u,v)\in E} w_{uv}x_{uv}^* + \sum_{(u,v)\in E} b_{uv}^* = \sum_{(u,v)\in E} b_{uv}^* \tag{5.26}$$

where s^* denotes the optimal subsidy for the problem in (5.13) and the result follows. □

5.4.7 A Descriptive Example

In this subsection, we use a small example to demonstrate our proposed model. Suppose a ridesharing system that operates on the well-known Nguyen-Dupuis transportation network. Figure 5.1 shows an example of this network with synthetic travel times and driving distances denoted on directed links.

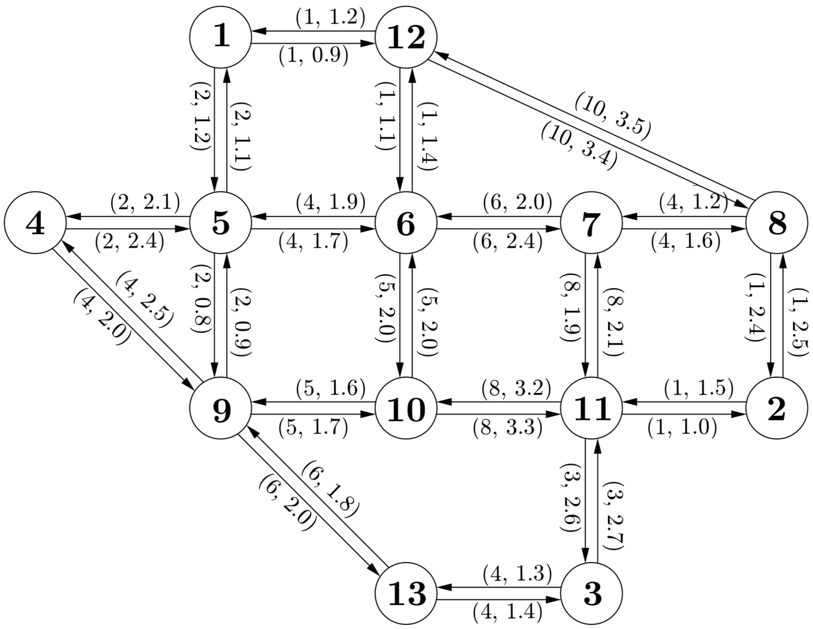


Figure 5.1: Nguyen-Dupuis Transportation Network. The tuple on each link shows the shortest-path travel time (in minutes) and driving distance (in miles) on that link.

Based on the information in Figure 5.1, the shortest-path travel time, τ, and driving distance, δ, between every two stations are computed in Tables 5.1 and 5.2, respectively.

For the sake of this example, we consider only 5 participants whose trip information are provided in Table 5.3. Without loss of generality, we assume that all users are open to both rider and driver roles.

By applying Equations (5.1), (5.2), and (5.3) to all combinations of users, we can generate the weighted directed graph G as shown in Figure 5.2(a). In this graph there exists a directed edge from user v to user u if (1) user u giving a rider to user v is spatio-temporally feasible, (2) both participants can obtain a positive utility. Let us consider users 3 and 4 as an instance. User 3 can start her trip

Table 5.1: The shortest-path travel times for Nguyen-Dupuis network

$o\backslash d$	1	2	3	4	5	6	7	8	9	10	11	12	13
1	0	12	14	4	2	6	12	11	4	11	20	1	10
2	12	0	4	18	15	11	5	1	14	9	1	11	8
3	14	4	0	14	12	17	11	5	10	11	3	15	4
4	4	18	14	0	2	6	12	15	4	9	17	5	10
5	2	14	12	2	0	4	10	13	2	9	18	3	8
6	6	11	17	6	4	0	6	10	6	5	14	1	12
7	12	5	11	12	10	6	0	4	12	11	8	7	15
8	11	1	5	16	14	10	4	0	15	10	2	10	9
9	4	14	10	4	2	6	12	15	0	5	13	5	6
10	11	9	11	9	9	5	11	15	5	0	8	6	11
11	13	1	3	17	18	14	8	2	13	8	0	12	7
12	1	11	18	5	3	1	7	10	5	6	15	0	11
13	10	8	4	10	8	12	15	9	6	11	7	11	0

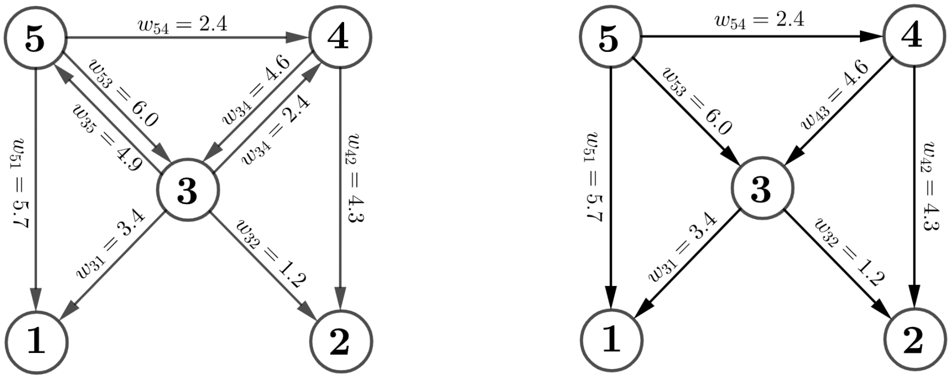
Table 5.2: The shortest-path driving distances for Nguyen-Dupuis network

$o\backslash d$	1	2	3	4	5	6	7	8	9	10	11	12	13
1	0	6.7	5.4	3.3	1.2	2.9	5.3	4.3	2.0	4.9	7.2	0.9	4.0
2	7.2	0.0	4.1	8.8	7.6	5.7	3.7	2.5	6.3	4.7	1.5	6.0	5.4
3	5.1	3.7	0.0	5.6	4.0	6.8	4.8	6.2	3.1	5.9	2.7	9.7	1.3
4	3.5	8.0	5.4	0.0	2.4	4.1	6.5	7.8	2.0	3.7	7.0	4.4	4.0
5	1.1	7.8	4.2	2.1	0.0	1.7	4.1	5.4	0.8	3.7	6.0	2.0	2.8
6	3.0	6.4	6.9	4.0	1.9	0.0	2.4	4.0	2.7	2.0	4.3	1.4	4.7
7	5.0	4.0	4.5	6.0	3.9	2.0	0.0	1.6	4.7	4.0	1.9	3.4	5.8
8	4.7	2.4	6.5	7.2	5.1	3.2	1.2	0.0	8.7	7.1	3.9	3.5	7.8
9	2.0	6.0	3.4	2.5	0.9	2.6	5.0	6.3	0.0	1.7	5.0	2.9	2.0
10	5.0	4.3	5.9	4.1	3.9	2.0	4.4	6.0	1.6	0.0	3.3	3.4	3.6
11	8.2	1.0	2.6	7.3	6.0	4.1	2.1	3.5	4.8	3.2	0.0	7.0	3.9
12	1.2	5.8	8.0	4.5	2.4	1.1	3.5	3.4	3.2	3.1	5.4	0.0	5.2
13	3.8	5.1	1.4	4.3	2.7	4.4	6.2	7.6	1.8	3.5	4.1	4.7	0.0

Table 5.3: The trip information for the participants in the small example

v	1	2	3	4	5
o	1	4	5	4	1
d	2	3	11	2	11
t	7:01	7:02	7:03	7:04	7:05
q	7:18	7:21	7:26	7:27	7:30

from station 5 at time 7:04 and pick up user 4 from station 4 at time 7:06 (i.e., $7{:}04 + \tau[5,\ 4]$). Then, she drops off user 4 at station 2 at time 7:24 (i.e., $7{:}06 + \tau[4,\ 2]$) and ends her trip in station 11 at time 7:25 (i.e., $7{:}24 + \tau[2,\ 11]$). Also, the mileage saving due to sharing rides in this case can be computed as $\delta[5,\ 11] - \delta[5,\ 4] - \delta[2,\ 11] = 6.0 - 2.1 - 1.5 = 2.4$ miles. Now, if user 4 gives a ride to user 3, still we can show that their time windows can be respected. However, in this case the mileage saving due to sharing rides can be computed as $\delta[4, 2] - \delta[4, 5] - \delta[11, 2] = 8.0 - 2.4 - 1.0 = 4.6$ miles. This implies that for users 3 and 4, the case of user 4 giving a ride to user 3 dominates the case of user 3 giving a ride to user 4. Therefore, based on proposition 5.1, we can remove the dominated edge (3 to 4 in this case). By removing all dominated edges, we can simplify graph *G* as shown in Figure 5.2(b).



(a) Original

(b) Reduced

Figure 5.2: Weighted directed graph *G*. The reduced graph can be obtained by removing the dominated edges between every two users.

Now, let us first show that graph *G* (without considering the direction of links) is a no-instance of the roommate problem with transferable utilities. Consider the sub-graph that includes users 3, 4, and 5. They form an odd-cycle of length 3 in which the weight of each edge is less than the sum of the weights of the other two edges. Hence, there exist no stable matching for graph *G* according to Eriksson and Karlander (2001). Note that the direction of links do not affect the stability, and they only help us find the roles of the users after finding the optimal matching.

Since the ridesharing system at hand is unstable, we solve the problem in (5.13) directly by a MIP solver to find the minimum amount of subsidy that makes this system stable. The optimal solution of this problem is presented in Figure 5.3. Based on the optimal solution, user 5 gives a ride to user 1 and from the mileage saving of 5.7, user 1 obtains a payoff of worth 1.55 miles and user 5 obtains the rest. Also, with the help of 1.55 miles worth of subsidy, user 4 gives a ride to user 3. As a result, their gain due to sharing rides is 4.6+1.55=6.15 miles from which 1.85 miles is assigned to user 3 and 4.3 miles is assigned to user 4. Finally, user 2 remains unmatched, and thus, receives no payoff.

Based on this result, one can further determine the optimal payments between users using Equations (5.3), (5.5) and condition (*i*) in Definition 5.2. For instance, consider users 3 and 4. User

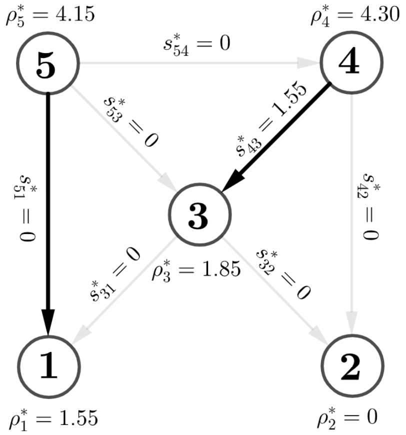


Figure 5.3: The optimal solution of the minimum subsidy problem for graph G. The solid edges determine the optimal matching. Also, ρ_v^* and s^* respectively denote the optimal payoff of user v and the optimal subsidy for a match (u, v).

3 is chosen to be a rider that will be served by user 4 as a driver. As a result, for user 4 we have:

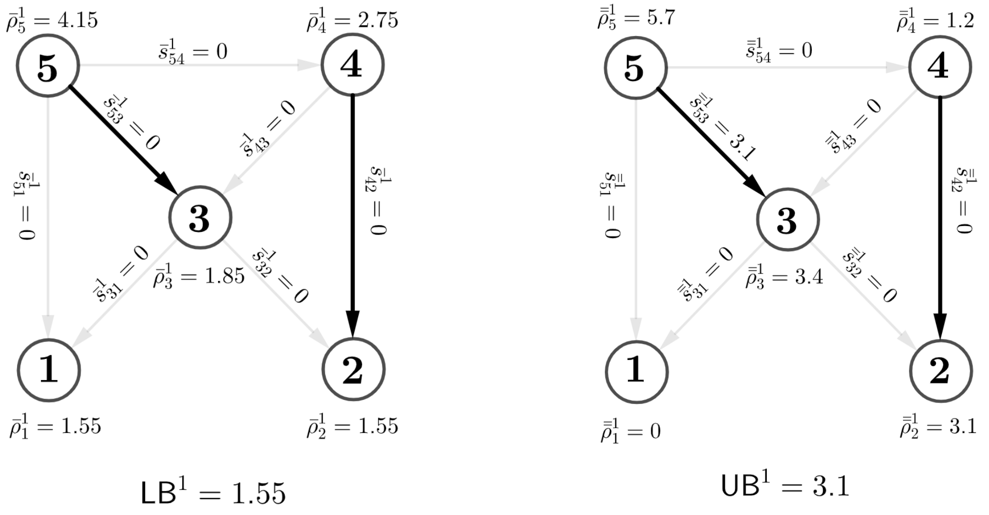
$$\begin{aligned} p_{43}^* &= \rho_4^* - \eta_{43} \\ &= \rho_4^* - (\delta[4,\ 2] - (\delta[4,\ 5] + \delta[5,\ 11] + \delta[11,\ 2]) \\ &= 4.3 - (8.0 - (2.4 + 6.0 + 1.0) \\ &= 5.70\ , \end{aligned}$$

and for user 3 we have:

$$\begin{aligned} p_{43}^* &= \rho_3^* - \theta_{43} \\ &= \rho_3^* - \delta[5,\ 11] \\ &= 1.85 - 6.0 \\ &= -4.15\ . \end{aligned}$$

This means that user 4 receive 5.70 miles worth of money and user 3 pays 4.15 miles worth of money. Note that since we subsidized their match, the payments do not cancel out and their difference is equal to the optimal subsidy allocated to that match, i.e., $s_{43}^* = 1.55$.

Finally, we describe the steps of the LR algorithm when applied to this small example. Our proposed LR algorithm converges in 2 iterations the solutions of which are presented in Figures 5.4 and 5.5. In the first iteration, we set the Lagrange multipliers to 0. As a result, subproblems LR_1 and LR_2 will be reduced to the max-weight general matching and the dual of the relaxed version of this problem, respectively. The difference between the optimal solutions to these problems yields a lower bound for the minimum subsidy problem, which is shown in Figure 5.4(a). This figure shows that user 5 serves user 3 and user 4 serves user 2, which yields the highest social welfare. Also, there is no subsidy allocated to any edge. This solution is clearly not feasible because user 1 is set to be unmatched, but receives a positive payoff. This implies that Constraint (5.13f) for user 1 is violated, and thus, multiplier ψ_1 will be inflated to further penalize this constraint in the LR objective. In order to obtain a feasible solution at this iteration, we solve the upper bound problem (5.20) given the matching M = {(5, 3), (4, 2)}. The optimal payoffs and subsidies given matching M are shown in Figure 5.4(b). This problem gives us an upper bound of 3.1. Since there is a large gap between the lower bound and upper, we continue the algorithm.

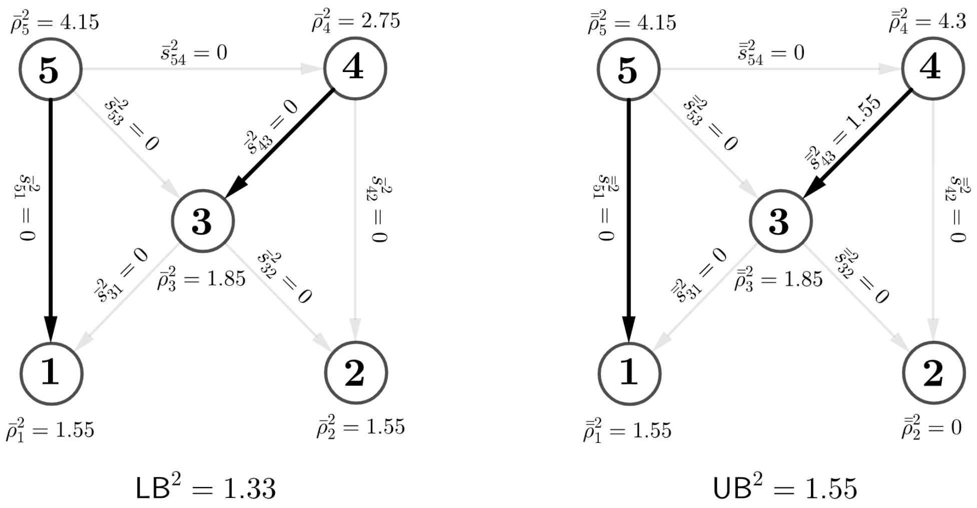

$\bar{\rho}_5^1 = 4.15$
$\bar{\rho}_4^1 = 2.75$
$\bar{s}_{54}^1 = 0$
$\bar{s}_{53}^1 = 0$
$\bar{s}_{43}^1 = 0$
$\bar{s}_{51}^1 = 0$
$\bar{s}_{42}^1 = 0$
$\bar{s}_{31}^1 = 0$
$\bar{\rho}_3^1 = 1.85$
$\bar{s}_{32}^1 = 0$
$\bar{\rho}_1^1 = 1.55$
$\bar{\rho}_2^1 = 1.55$
5
4
3
1
2
$\mathsf{LB}^1 = 1.55$
$\bar{\bar{\rho}}_5^1 = 5.7$
$\bar{\bar{\rho}}_4^1 = 1.2$
$\bar{\bar{s}}_{54}^1 = 0$
$\bar{\bar{s}}_{53}^1 = 3.1$
$\bar{\bar{s}}_{43}^1 = 0$
$\bar{\bar{s}}_{51}^1 = 0$
$\bar{\bar{s}}_{42}^1 = 0$
$\bar{\bar{s}}_{31}^1 = 0$
$\bar{\bar{\rho}}_3^1 = 3.4$
$\bar{\bar{s}}_{32}^1 = 0$
$\bar{\bar{\rho}}_1^1 = 0$
$\bar{\bar{\rho}}_2^1 = 3.1$
5
4
3
1
2
$\mathsf{UB}^1 = 3.1$

(a) Lower bound

(b) Reduced

Figure 5.4: The lower bound and upper solutions of the LR algorithm in iteration 1.

In the second iteration, the matching changes due to the change in the value of ψ_1. Figure 5.5(a) shows the optimal solution of the subproblems in the second iteration. In this iteration, user 5 is chosen to serve rider 1, and user 4 is chosen to serve user 3. Again, subsidies are all set to be zero. This solution, which yields a lower bound of 1.33, is clearly infeasible because this time user 2 is set to be unmatched and receives a positive payoff. Hence, we inflate ψ_2 appropriately to penalize violating Constraint (5.13f) for user 2. In order to obtain a feasible solution, again we solve the upper bound problem given matching M = {(4, 3), (5, 1)}. The optimal solution to the upper bound problem in the second iteration yields the optimal payoffs and subsidies, because it lowers the best upper bound to 1.55, which is the same as the best lower bound. Hence, we stop the LR algorithm and choose the solution in Figure 5.5(b) as the optimal solution to the minimum subsidy problem.


$\bar{\rho}_5^2 = 4.15$
$\bar{\rho}_4^2 = 2.75$
$\bar{s}_{54}^2 = 0$
$\bar{s}_{53}^2 = 0$
$\bar{s}_{43}^2 = 0$
$\bar{s}_{51}^2 = 0$
$\bar{s}_{42}^2 = 0$
$\bar{s}_{31}^2 = 0$
$\bar{\rho}_3^2 = 1.85$
$\bar{s}_{32}^2 = 0$
$\bar{\rho}_1^2 = 1.55$
$\bar{\rho}_2^2 = 1.55$
5
4
3
1
2
$\mathsf{LB}^2 = 1.33$
$\bar{\bar{\rho}}_5^2 = 4.15$
$\bar{\bar{\rho}}_4^2 = 4.3$
$\bar{\bar{s}}_{54}^2 = 0$
$\bar{\bar{s}}_{53}^2 = 0$
$\bar{\bar{s}}_{43}^2 = 1.55$
$\bar{\bar{s}}_{51}^2 = 0$
$\bar{\bar{s}}_{42}^2 = 0$
$\bar{\bar{s}}_{31}^2 = 0$
$\bar{\bar{\rho}}_3^2 = 1.85$
$\bar{\bar{s}}_{32}^2 = 0$
$\bar{\bar{\rho}}_1^2 = 1.55$
$\bar{\bar{\rho}}_2^2 = 0$
5
4
3
1
2
$\mathsf{UB}^2 = 1.55$

(a) Original

(b) Reduced

Figure 5.5: The lower bound and upper bound solutions of the LR algorithm in iteration 2.

5.5 Numerical Experiment

In this section, we conduct a numerical study to (*i*) evaluate the performance of our LR algorithm, (*ii*) showcase the advantage of including role flexibility in P2P ridesharing systems, (*iii*) present the advantage of subsidizing the system, and (*iv*) find the impact of changing different system parameters on a number of performance measures. All experiments are conducted on a 3.50 GHz Intel Xeon machine with a 64-bit version of the Windows 10 operating system with 128.0 GB RAM. The data preparation and the subgradient optimization algorithm are coded in `Python 3.7`, and all the optimization problems are solved using `GUROBI 9.0`. Next, we briefly introduce our dataset, followed by a discussion on the experiment setup of our ridesharing system along with parameter setting for our proposed Lagrangian Relaxation algorithm. Finally, the results of various experiments will be presented.

5.5.1 Dataset

In our numerical experiments, we use the New York City Taxi dataset[1] to obtain the users' trip information including the origin, destination, and earliest departure time. For large-scale implementation, we intend to use the data of trips that took place only in the Manhattan area during the peak hour of [18:00,19:00] on February 19, 2016. The transportation network of the Manhattan area, shown in Figure 5.6, is extracted from `OpenStreetMap` and contains about 4000 stations and 10000 transportation links. Without loss of generality, we assume that all trip requests are

[1] http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml

announced before the start time of the peak hour and should take place during this period. Also, we keep only those trips whose length is between 5 and 60 minutes in our pool of trips to make sure that our generated networks are dense enough. Doing so, we obtain a large pool of $N = 20,000$ trips. We introduce a "penetration" parameter, denoted by $\aleph$, that represents the percentage of trips that are registered in our ridesharing system. It is worth mentioning that we consider the shortest travel time path as the selected path for all trips. Shortest-path travel times and driving distances are extracted from Google API. In order to find the latest arrival time of users, we introduce a constant"time flexibility" parameter, denoted by β, for all users. As a result, the latest arrival time of user u can be computed based on the known information as $q_u = t_u + \tau(o_u, d_u) + \beta$.

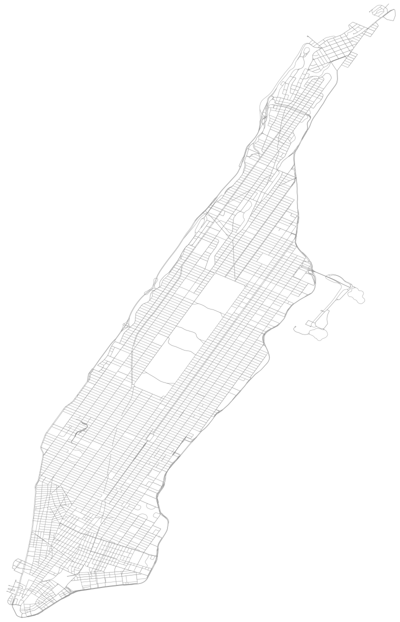


Figure 5.6: Transportation network of the Manhattan area

5.5.2 Experiment setup and Parameter Settings

We define 7 distinct scenarios of the ridesharing network with parameter settings listed in Table 5.4. In this table, the "role flexibility" parameter, denoted by ξ, represents the percentage of users with flexible roles. For each scenario, we generate 10 different instances by randomly picking $\aleph$% of trips from the pool of N available trips mentioned in the previous part, and further randomly assigning $(100 - \xi)/2$ (%) of participants to serve only as riders and $(100 - \xi)/2$ (%) of participants to serve only as drivers. Following this procedure, the instances of the base scenario (Scenario 1) are generated by randomly picking 25% × 20000 = 5000 users from which 1250 users are assigned as only riders, 1250 users are assigned as only drivers, and the rest can play both roles, and thus, they let the system choose the most profitable role for them. All other scenarios are built according to the base one by changing the value of a single parameter at a time. In Table 5.4, the value of the parameters in the base scenario and their changed values in other scenarios are written in bold.

Table 5.4: Parameter values for different scenarios. Scenario 1 is the base scenario. In Scenarios 2 to 7, we change one of the three parameters. The parameter changed in each scenario compared to the base scenario is shown in bold.

Scenario	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Penetration rate $\aleph$ (%)	25	25	25	25	25	**20**	**30**
Time flexibility β (min)	10	10	10	**5**	**15**	10	10
Role flexibility rate ξ (%)	50	**25**	**100**	50	50	50	50

5.5.3 Comparison Between Branch-and-Bound and Lagrangian Relaxation

In this subsection, we compare the performance of the proposed Lagrangian relaxation algorithm with a conventional LP-based branch-and-bound (B&B) algorithm. Table 5.5 presents the result of applying the two matching methods on the scenarios listed in Table 5.4. Note that for the sake of fair comparisons, the value of ε and T for both methods in the experiments are set to 0.01 and 300, respectively. Thus, we terminate the LR/B&B algorithm for each instance when the optimality gap is lower than 0.01, or after 5 minutes is passed.

Based on the result of Proposition 5.3, we set the initial values of the Lagrange multipliers to $(\phi^0, \psi^0) = (0, 0)$. We further let the initial value of the scale parameter α be 0.5. Finally, we set the parameter m to 8, i.e., we cut α by half after 8 consecutive iterations with no improvement in the value of BLB. The optimal solutions are obtained by solving the problem with a branch-and-cut (B&C) algorithm with time limit of 10000 seconds. To make sure that the optimal solution can be obtained within this time frame, we initialize the B&C method using the best implementable solution found by the LR algorithm. In this table, the solution quality metrics for the B&B and LR algorithms are computed as:

$$\begin{aligned}
&\text{algorithm absolute gap } (\Delta) = \mathsf{BUB} - \mathsf{BLB}\,,\\
&\text{actual absolute gap } (\Delta^*) = \mathsf{BUB} - \sum_{(u,v)\in E}\sum s^*_{uv}\,,\\
&\text{algorithm relative gap } (\delta) = 100 \times (\mathsf{BUB} - \mathsf{BLB})/\mathsf{BUB}\,,\\
&\text{actual relative gap } (\delta^*) = 100 \times \Big(\mathsf{BUB} - \sum_{(u,v)\in E}\sum s^*_{uv}\Big)\Big/\mathsf{BUB}\,,
\end{aligned}$$

where s^* denotes the optimal subsidy. Note that all the reported values are averaged over 10 instances. The dominant values between the two methods in each case are emboldened. This table clearly suggests that the proposed LR method significantly outperforms the B&B algorithm in all scenarios. More significantly, the B&B algorithm fails to find a feasible solution in the given time for scenarios 3, 5, and 7, while our LR algorithm yields fairly tight upper bounds. This observation implies that one cannot rely on the conventional solution approaches as the underlying graph of the ridesharing system grows larger and denser, and hence, the need for more efficient algorithms like the one proposed in this chapter is indispensable. Finally, it is worth mentioning that although the average relative gap of the LR is greater than 5% in all scenarios, the actual absolute gap is very small (< 0.1 miles) which indicates that the differences are negligible.

In order to showcase the convergence rate of both methods, Figure 5.7 plots the lower bounds, upper bounds and relative optimality gaps of the two methods over time for an arbitrary instance of the base scenario. From this figure, we can infer that the first iteration of the LR algorithm takes a long time due to starting from null matching. However, the subsequent iterations take shorter times to solve as we initialize the matching variables with the ones from the previous iterations. This figure clearly shows how the the LR bounds approach the optimal line in a small number of iterations, within a short period of time (i.e., in less than a minutes). However, the speed of convergence decreases rapidly with number of iterations. One can easily notice that the red lines denoting the LR bounds do not change significantly after 160 seconds. This figure also suggests that the choice of $(0, 0)$ for initial values of the Lagrange multipliers help us start the LR algorithm with a tight gap. Finally, this figure demonstrates that the B&B algorithm takes more than 180 seconds to find a feasible solution (an upper bound). Moreover, the lower bound of B&B starts from the obvious value of zero, and the rate of convergence of B &B is extremely slow to the point that there is almost no improvement after 100 seconds.

Table 5.5: The Comparison Between the B&B and Lagrangian Relaxation for different scenarios. Dominant results are shown in bold.

Scen.	**Branch-and-Bound**						**Lagrangian Relaxation**						**Optimal**
	BLB	BUB	Δ	δ	Δ^*	δ^*	BLB	BUB	Δ	δ	Δ^*	δ^*	
1	0.67	6.91	6.23	90.21	0.71	10.11	**5.83**	**6.29**	**0.46**	**7.33**	**0.09**	**1.47**	6.19
2	1.80	3.38	1.58	48.18	0.13	4.22	**3.03**	**3.27**	**0.24**	**6.93**	**0.03**	**0.70**	**3.25**
3	0.23	∞	∞	–	∞	–	**8.26**	**8.87**	**0.61**	**7.02**	**0.14**	**1.62**	8.73
4	4.70	6.05	1.35	22.08	0.09	1.54	**5.38**	**6.03**	**0.65**	**10.87**	**0.07**	**1.22**	5.96
5	0.07	∞	∞	–	∞	–	**5.10**	**5.58**	**0.49**	**8.76**	**0.13**	**2.35**	5.46
6	2.48	5.49	3.01	54.73	0.27	4.82	**4.92**	**5.28**	**0.36**	**6.49**	**0.06**	**0.94**	5.22
7	0.47	∞	∞	–	∞	–	**6.05**	**6.53**	**0.48**	**7.30**	**0.09**	**1.33**	6.44

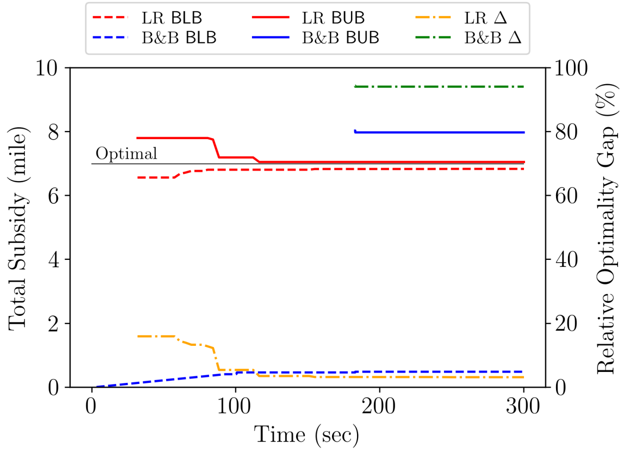


Figure 5.7: The comparison between the convergence rates of the Lagrangian relaxation and branch-and-bound methods for an arbitrary instance of the base scenario.

5.5.4 Social and Individual Advantages of Role Flexibility

In order to understand the impact of introducing role assignment variables in the performance of a P2P ridesharing market, we compare the social and individual characteristics of our proposed flexible-role system with alternative fixed-role systems over 10 random instances of the base scenario (scenario 1). To obtain alternative fixed-role systems for each instance, we generate 1000 samples of the problem with randomly assigned roles. Note that those users whose roles were originally fixed in the experiments will be assigned the same roles in all 1000 samples, and the difference between the 1000 alternative fixed-role systems is on the role-assignments of formerly flexible participants. For studying the social performance of the system, we consider the two following measures:

$$\text{matching rate} = 100 \times 2\Big(\sum_{(u,v)\in E}\sum x^*_{uv}\Big)\Big/|V|\,,$$

$$\text{social welfare} = \sum_{(u,v)\in E}\sum w_{uv}x^*_{uv}\,,$$

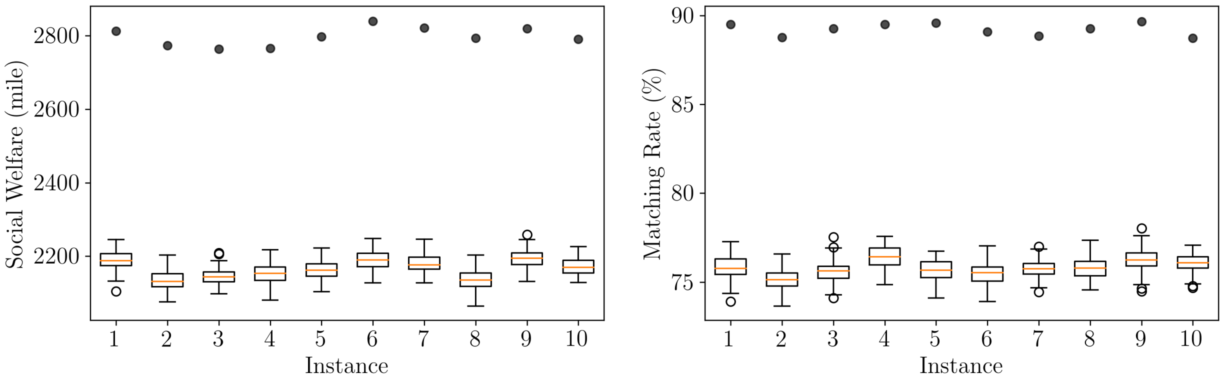
where x^* and ρ^* denote the optimal matching and payoffs for both system types. Note that since role assignment is not a decision variable in the 1000 randomly generated alternative fixed-role systems, x^* and ρ^* for these problems can be found as the primal and dual optimal solutions of a maximum weighted bipartite matching problem, respectively. Due to the stability of the assignment game, s^* is the vector of zeros for these problems. Figure 5.8(a) demonstrates the results for each individual instance separately (instances 1 to 10). The dot points in this figure show the result of including role assignment in the ridesharing model, and the box plots are the result of 1000 samples of the same configuration, but with arbitrarily pre-assigned roles. One can easily notice the advantage of letting the operator decide the role assignment of participants on both matching rate and social welfare. These results are not surprising, since treating role assignment as a decision variable provides higher levels of flexibility, leading to to higher matching rates as well as social welfare.

For showcasing the benefits of incorporating role flexibility in a P2P ridesharing system for each individual user, we consider the following measure:

$$\text{payoff difference for user } u = \rho^*_u - \Big(\sum_{i=1}^{1000} \bar{\rho}^{*i}_u\Big)\Big/1000\,, \qquad\qquad \forall\, u \in V\,,$$

where ρ^*_u is the payoff of user u in the original system with role flexibility, and $\bar{\rho}^{*i}_u$ is the payoff of user u in sample i of alternative fixed-role systems. In Figure 5.9, we show the distributions of payoff differences under two groups of participants for each individual instance of the base scenario.

In this figure, group 1 represents the set of users whose roles are fixed in the system with role flexibility, and group 2 represents the rest of users whose roles are chosen optimally by the system. This figure clearly suggests that the payoff difference of users in group 1 is normally distributed around 0 with majority of them being almost 0, which implies that they are not better off under a system with role flexibility. However, the distribution of payoff differences for the users in group 2 are highly skewed to right, with the majority of them having a positive payoff difference. Therefor, we can conclude that those users who let the operator determine their roles in a P2P ridesharing system with role flexibility can significantly benefit from such systems.



(a) Social Welfare

(b) Matching Rate

Figure 5.8: System-level advantages of role-flexibility. A system with role flexibility is represented by dot point, and 1000 fixed-role alternative systems are shown by the box plots.

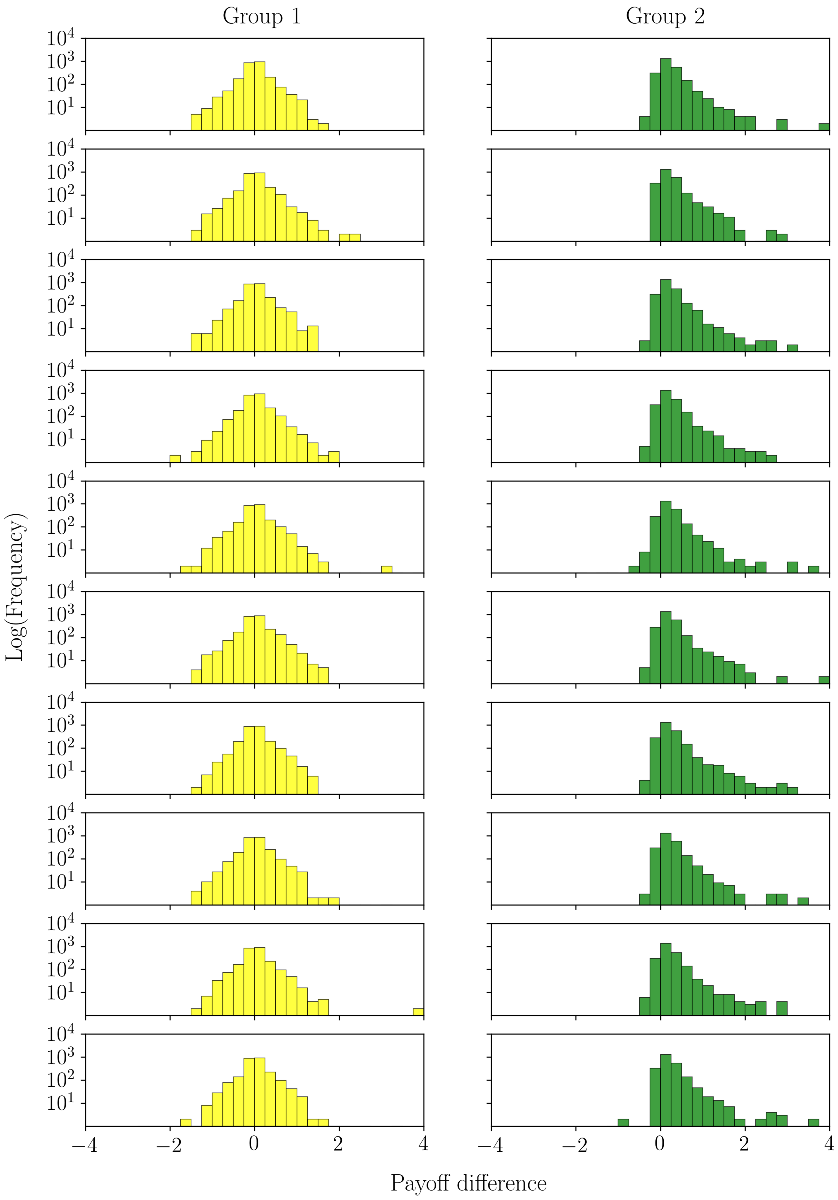


Figure 5.9: User-level advantages of role-flexibility. Histograms in each row belong to one instance of the base. Yellow histograms (column to the left) demonstrate the pay-off differences for users with fixed roles in the original problem, and green histograms (column to the right) show the pay-off differences for users with flexible roles in the original problem.

5.5.5 Advantages of Subsidizing Unstable P2P Ridesharing Systems with Role Flexibility

As mentioned earlier in Section 5.3, different approaches have been introduced in the literature for stabilizing an unstable graph. In this chapter, we choose to subsidize the unstable ridesharing system to make it stable. However, this comes at the price of having a payment mechanism that is not

Table 5.6: The comparison between the minimum subsidy and maximum stable subset problem for different scenarios.

Scen.	Minimum Subsidy			Maximum Stable Subset			Performance Measures		
	Total Subsidy	Social Welfare	Total Matched Users	Social Welfare		Total Matched Users Ratio	Added Social Welfare		Added Matched Users
				Ratio	Relative Gap		Worst	Best	
1	6.19	2797.81	4334.20	74.24	57.82	71.17	0.04	117.69	101.49
2	3.25	2509.86	4075.60	88.15	18.22	86.18	0.03	90.85	86.26
3	8.73	3229.86	4639.60	69.90	66.49	65.85	0.12	115.80	94.17
4	5.96	2063.41	3532.80	98.13	1.96	97.29	0.04	6.92	8.51
5	5.46	3017.43	4492.20	66.21	236.04	64.02	0.09	191.20	151.72
6	5.22	2145.08	3400.20	85.98	19.05	84.03	0.08	58.25	52.46
7	6.44	3461.34	5273.60	77.73	64.49	75.91	0.08	119.57	99.10

budget-balanced. Hence, other stabilizing approaches that involve adding/deleting edges/vertices may seem more appropriate as they do not compromise the budget-balancedness of the payments. In this subsection, we compare the performance of our proposed methodology with that of the vertex deletion approach that can be translated into finding (and serving) a subset of users for whom the outcome of the market is stable. The goal of this problem is to maximize the social welfare for a subset of users whose matching is stable. The mathematical formulation for this problem, which is hereinafter referred to as *maximum stable subset* problem, is presented in model (D.2) in Appendix D.2. A Branch-and-Cut algorithm with time limit of 10000 seconds is used to solve instances of this problem. The results of applying this method on all scenarios along with the results of the minimum subsidy problem are summarized in Table 5.6.

In this table, the reported ratios represent the social welfare and total matched users of the best feasible solution of the maximum stable subset problem as a percentage of corresponding optimal values in the minimum subsidy problem.

Based on the given time and the large relative gaps between the best upper and lower bounds of the maximum stable subset problem, we can conclude that this problem is significantly harder to solve when compared to the minimum subsidy problem. We further observe that the gap becomes larger as the number of nodes or edges in the underlying graph grows (i.e., scenarios 3, 5, and 7). This table also suggests that the best feasible social welfare of the maximum stable subset problem (i.e., the best lower bound) in all scenarios is significantly lower than optimal social welfare of the minimum subsidy problem by a significantly large degree. In order to see the impact of added subsidy more clearly, we have introduced the following performance measures per a unit of added subsidy (presented in the last three columns):

$$\text{worst-case added social welfare} = \Big(\sum_{(u,v)\in E}\sum w_{uv}x^*_{uv} - \widehat{\mathsf{BUB}}\Big)\Big/\sum_{(u,v)\in E}\sum s^*_{uv}\,,$$

$$\text{best-case added social welfare} = \Big(\sum_{(u,v)\in E}\sum w_{uv}x^*_{uv} - \widehat{\mathsf{BLB}}\Big)\Big/\sum_{(u,v)\in E}\sum s^*_{uv}$$

$$\text{added matched users} = \Big(\sum_{(u,v)\in E}\sum x^*_{uv} - \sum_{(u,v)\in E}\sum \hat{x}^*_{uv}\Big)\Big/\sum_{(u,v)\in E}\sum s^*_{uv}\,,$$

where x^*_{uv} and s^*_{uv} respectively represent the optimal matching and subsidies in the minimum subsidy problem. Also, $\hat{x}^*$, $\widehat{\mathsf{BLB}}$, and $\widehat{\mathsf{BUB}}$ respectively denote the best feasible matching, best lower bound, and the best upper bound in the maximum stable subset problem. The first two measures provide a range for average increase in social welfare per a unit of subsidy added. Table 5.6 suggests that the rate of increase in social welfare per one unit increase in the value of added subsidy is always positive and can be extraordinarily high. In addition, the rate of added matched users due to adding subsidy is very large, which clearly shows that by injecting small subsidies into the system we can incentivize a high number of commuters to participate and do not deviate from proposed outcomes by the system.

Overall, the results in this table indicate that the policy of injecting subsidies to the ridesharing system significantly outperforms the policy of excluding a subset of customers, especially in large-scale problems. (Note that the best added social welfare is associated with a feasible solution of the maximum stable subset problem; the small values of worst added social welfare do not correspond to a feasible solution.) Comparing the value of these measures between different scenarios, we observe that the policy of adding subsidy benefits those systems in which the users announce rather wide/tight time windows (scenario 5/scenario 2) the most/least. An explanation for this observation is that allowing for wider time windows give rise to many highly beneficial coalitions that are not temporally feasible otherwise. Thus, hindering a subset of users to participate in the system results in losing such coalitions.

Finally, two additional factors contribute to the higher value of the minimum subsidy problem compared to the maximum stable subset problem in practice: (1) preventing some users from participating in the system in a given day, as is the case in the maximum stable subset solution, can discourage them from participating in the future. This is a very important drawback, since the P2P ridesharing market lays on the foundation of peer travelers fulfilling the supply and demand roles, and a low participation rate can lead to a downward spiral that could lead to the demise of the market; (2) although the market is founded on recurring trips, new members may join and existing members may change their travel schedules every day. As such, matching solutions need to be obtained on a daily basis, if not more frequently, making the system sensitive to solution times.

Given the large optimality gaps of the minimum stable subset problem in 10,000 seconds, we may need several extra hours to close the gap (if enough memory is available). In addition, the solver took at least 30 minutes to find a positive feasible solution for this problem in all instances. On the contrary, the combination of Branch-and-Cut and the proposed Lagrangian relaxation algorithms yields the optimal solution for the minimum subsidy problem in no more than 20 minutes.

5.5.6 Sensitivity Analysis of the Ridesharing System Parameters

In this subsection, we study the impact of changing parameters $\aleph$, β, and ξ on a series of system performance measures, namely, social welfare, matching rate, total subsidy, relative social welfare, and relative subsidy, the last two of which can be defined as:

$$\text{relative social welfare} = 100 \times \sum_{(u,v)\in E}\sum w_{uv}x^{*}_{uv} \Big/ \sum_{(u,v)\in E}\sum w_{uv}x^{**}_{uv} \, ,$$

$$\text{relative subsidy} = 100 \times \sum_{(u,v)\in E}\sum s^{*}_{uv} \Big/ \sum_{(u,v)\in E}\sum w_{uv}x^{*}_{uv} \, ,$$

where x^{*} and s^{*} represent the matching and subsidies of the optimal solution of the minimum subsidy problem in (5.13), respectively, while x^{**} denotes the optimal solution of the maximum weighted matching in (5.4). Figures 5.10, 5.11, and 5.12 demonstrate the results for parameters ξ, β, and $\aleph$, respectively. In order to evaluate the statistical significance of observed trends, we run an ANOVA test and the Tukey's pairwise comparison tests for the data in each box plot. Table 5.7 summarizes the results of these tests by providing the corresponding p-values.

In the base scenario, we assume that 25% of commuters register their trips in the ridesharing system. In order to study the effect of the system's penetration rate $\aleph$ on different performance metrics, we change the percentage of participation to 20% and 30% in scenarios 6 and 7, respectively. The results are provided in Figure 5.10.

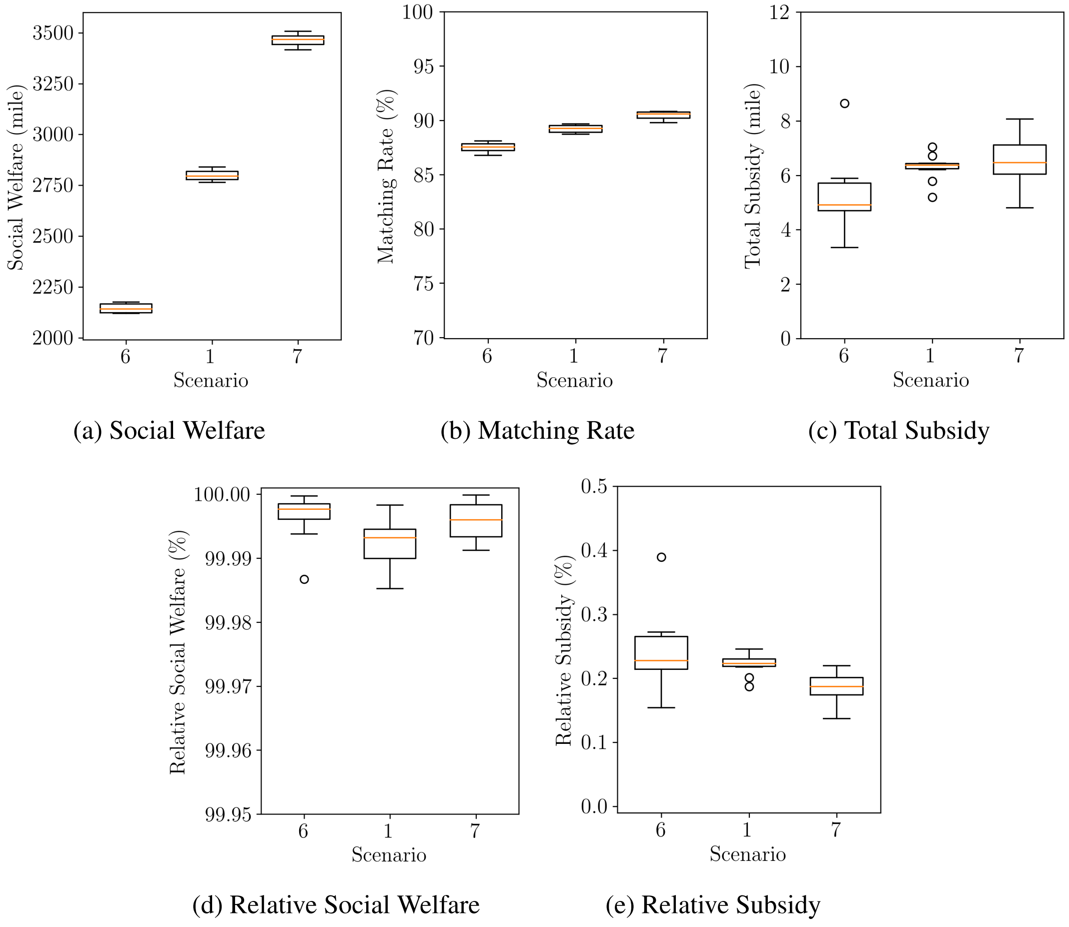
Figures 5.10(a) and 5.10(b) suggest that the social welfare and the matching rate increase with

Table 5.7: The p-values of ANOVA and Tukey's tests to evaluate the significance of parameters on the system performance metrics. The significant values under confidence level of 5% are written in bold.

System Quality Metrics	**Effect of ξ**				**Effect of β**				**Effect of $\aleph$**			
	ANOVA	Tukey			ANOVA	Tukey			ANOVA	Tukey		
		1&2	1&3	2&3		1&4	1&5	4&5		1&6	1&7	6&7
Social Welfare	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
Matching Rate	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.03**	**0.00**	**0.00**	**0.00**	**0.00**
Total Subsidy	**0.00**	**0.00**	**0.00**	**0.00**	0.32	0.83	0.30	0.60	**0.02**	0.08	0.84	**0.02**
Rel Social Welfare	**0.03**	0.06	0.90	**0.04**	**0.00**	**0.02**	0.31	**0.00**	0.06	0.07	0.13	0.90
Relative Subsidy	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.01**	0.15	**0.00**	**0.01**	0.44	0.13	**0.01**

the number of participants. As the system's penetration rate grows, the spatiotemporal proximity of trips increases. Consequently, users find more opportunities to share a ride with their peers (hence the higher matching rate), and higher quality matches (hence the higher social welfare). Also, the rate of increase in both matching and social welfare is linear, which indicates that the marginal value of an additional user remains the same as number of registered users increases. Table 5.7 indicates that these trends are statistically significant. Figure 5.10(c) suggests that the total amount of subsidy required for stabilizing the ridesharing system increases with the number of participants. However, this figure shows that the rate of increase in total subsidy decreases as $\aleph$ goes up. This is due to the fact that the chance of finding a partner increases as $\aleph$ increases. As a result, the number of unexposed users decreases. Since unexposed users usually play an important role in destabilizing a matching, we conclude that the level of instability in the system decreases with the number of participants.

Figure 5.10(d) indicates that the social welfare of the optimal solution of the minimum subsidy



(a) Social Welfare

(b) Matching Rate

(c) Total Subsidy

(d) Relative Social Welfare

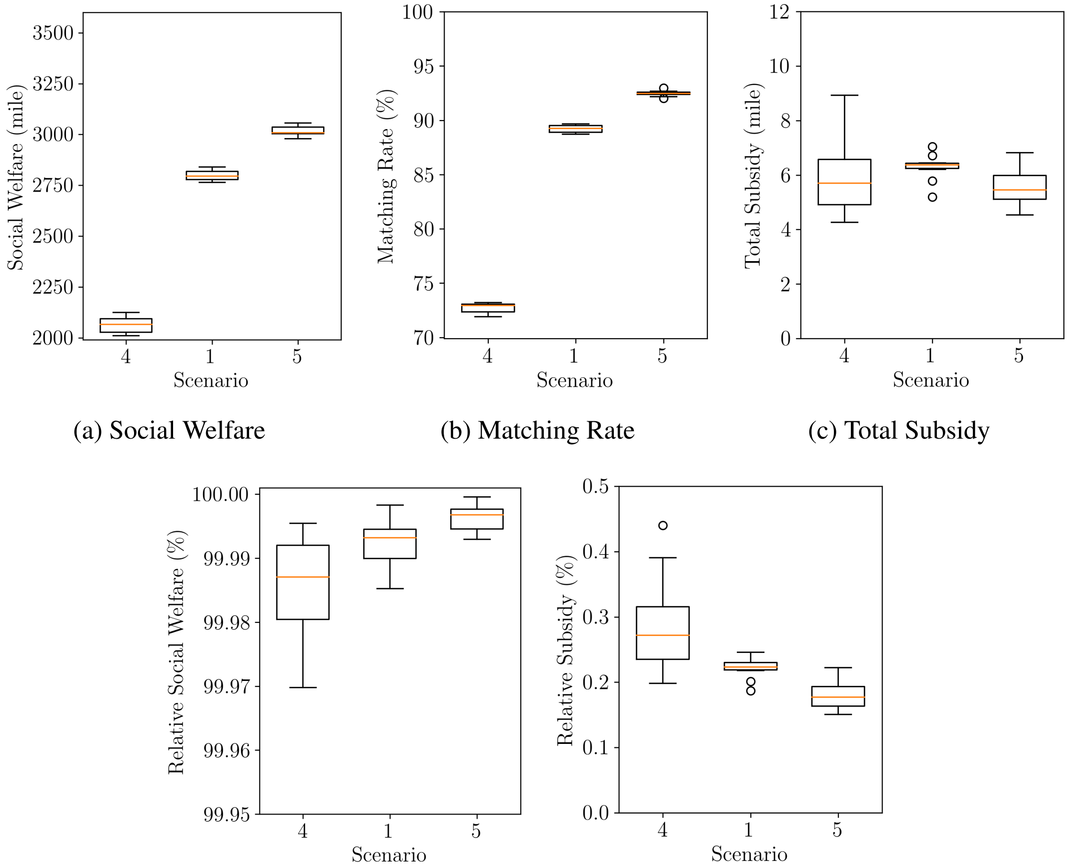
(e) Relative Subsidy

Figure 5.10: Impact of participation rate on the performance measures

problem is almost the same as the maximum possible social welfare. Comparing the p-values for this measure, there is no significant difference between its values under different number of participants. Also, Figure 5.10(e) clearly shows that the total amount of subsidy required for stabilizing the ridesharing systems is negligible when compared to the social welfare of stable outcomes. This figure also confirms that the level of instability in the system decreases with number of participants. From these two figures, we can infer that the shortcoming of adding subsidy, i.e., not being budget-balanced, starts to fade out as more users participate.

The base scenario considers a maximum allowable time flexibility (i.e., additional time budget) of 10 minutes. In scenarios 4 and 5, we investigate the impact of changing this value to 5 and 15 minutes, respectively. Figure 5.11 displays the system-level performance results of these scenarios. Figure 5.11(b) demonstrates that the matching rate increases as we increase the time budget. This observation is in line with intuition, since by increasing the time budget of participants we allow them to take longer detours, thereby providing more opportunities for drivers to serve riders. This figure also suggests that the rate of increase in the matching rate decreases with time budget. This is due to the fact that when all participants are fairly flexible, the matching rate becomes high, not leaving much more room for improvement. Figure 5.11(a) demonstrates that the average social welfare follows the same trend as the matching rate. Figure 5.11(c) does not show any significant difference between the average subsidy required for stabilizing the graphs under different levels of time budget. On the one hand, increasing the time budget results in an increase in the number of edges and consequently increasing the number of odd cycles. On the other hand, the number of unexposed users decreases due to having a higher matching rate. This figure suggests that these two effects cancel out each other and hence there is no substantial change in the total subsidy. Figure 5.11(d) clearly shows that the social welfare gained by subsidizing the system becomes closer to the maximum possible social welfare. The results in Figure 5.11(d) combined with those of Figure 5.11(e) suggests that stabilizing the system through adding subsidy becomes more beneficial as users announce wider time windows.

Finally, the base scenario assumes that half of participants are flexible in their role assignments. In this section, we investigate the impact of having a lower (scenario 2) or a higher (scenario 3) portion of participants enter the system with pre-determined roles. Figure 5.12 demonstrates how various performance metrics change as the percentage of participants with flexible roles, ξ, increases from 25% to 100%. Figure 5.12(b) shows that the matching rate increases sublinearly as we increase the value of ξ (Note that the difference between scenario 1 and 3 is twice the difference between 1 and 2). This observation is intuitive as by allowing the system to choose the roles, we increase the number of potential matches in the system, thereby increasing the matching rates. Figure 5.12(a) suggests that the social welfare experiences a similar trend. Figure 5.12(c) clearly shows a significant increase in the total amount of required subsidy when ξ changes from 25% to 100%.



(a) Social Welfare

(b) Matching Rate

(c) Total Subsidy

(d) Relative Social Welfare

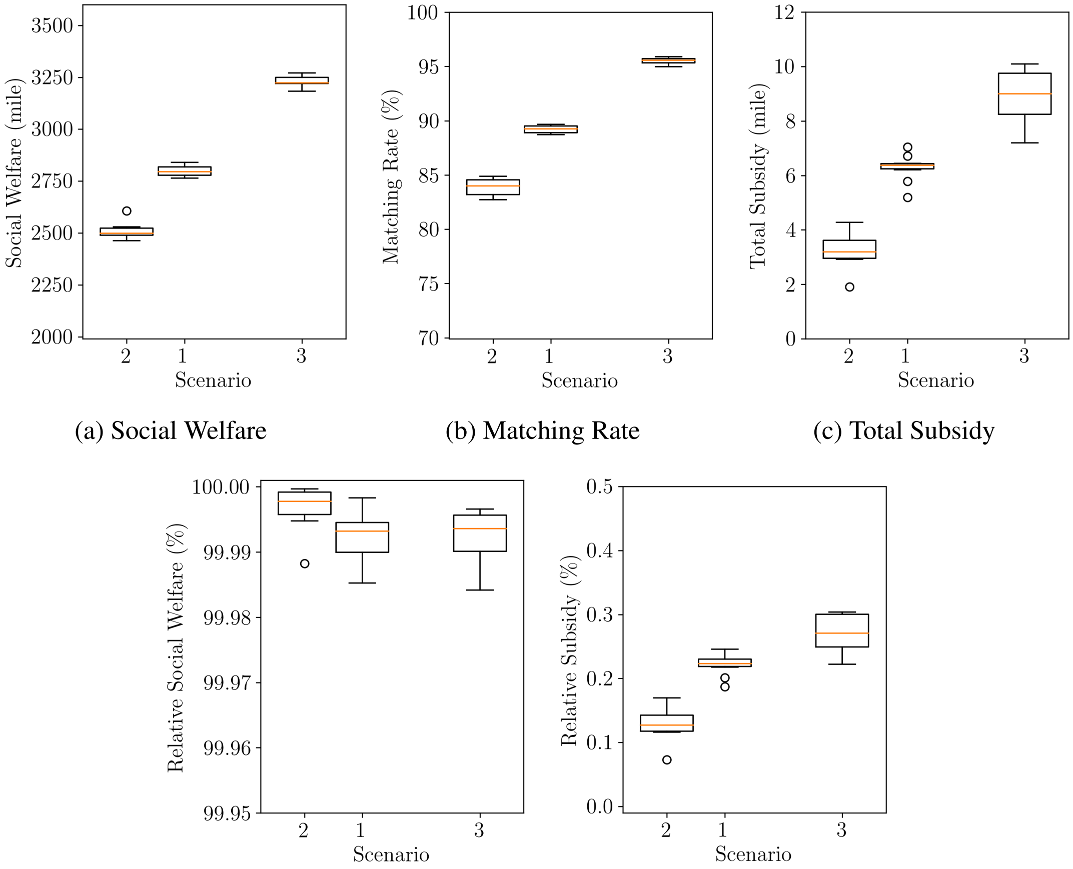
(e) Relative Subsidy

Figure 5.11: Impact of time flexibility on the performance measures

This is not surprising as we know that by increasing ξ the graph becomes highly non-bipartite, and hence, we expect that the level of instability in the system would rapidly increase as a result of that. Figure 5.12(d) indicates that by increasing the percentage of participants with role flexibility, we initially observe a small decrease in the relative social welfare due to stability; however, it fades out after a certain point. According to Figure 5.12(e), the rate of total subsidy required for stabilizing the system to gain a higher social welfare increases, which is not favorable. However, we argue that first this rate increases sublinearly, and second, the ratio in the most extreme case (i.e., $\xi = 100\%$) is still very small (lower than 0.3% of social welfare).

5.6 Conclusion

Recent developments in the communication technologies as well as the ubiquity of high-speed internet on the one hand, and the need for fast and reliable solutions for increasing demand for



(a) Social Welfare

(b) Matching Rate

(c) Total Subsidy

(d) Relative Social Welfare

(e) Relative Subsidy

Figure 5.12: Impact of flexible-role participants' rate on the performance measures

mobility on the other hand, have resulted in a movement toward sharing the available vacant seats on personal vehicles. A P2P ridesharing system is a platform in which individuals can be arranged to provide transportation services to their peers while completing their personal trips. Such systems can attract enough customers to be considered as a reliable alternative for major transportation modes.

Despite major benefits that P2P ridesharing systems have to offer, there are two, somewhat related, major questions that need to be addressed for these systems to be deployed successfully. First, how can we make these systems more profitable? And second, how can system users be compensated fairly for participating in the system such that they do not have any incentive to deviate from the matching and payments offered to them by the system? In this chapter, we provide answers to both of these questions.

A P2P ridesharing system is traditionally categorized as a two-side market, with two mutually exclusive sets of riders and drivers. In reality, some participants may own a personal vehicle and be willing to play either of these roles upon request, undermining the assumption of two disjoint sets. In fact, this study shows that bisecting the network into fixed-role agents will result in rather significant opportunity costs. Our results from a simulated network confirm that including the assignment role variables will increase the matching rate as well as the social welfare, making ridesharing system one step closer to becoming a viable option.

The second question pertains to the concept of stability that originates from the game theory literature. Since the trip information of commuters are common knowledge, they should be fairly compensated as they may otherwise stop to participate and start forming new coalitions that are more profitable to them. To meet the need for stability, we seek to form a matching market in which the payoffs of riders and drivers are beneficial to both parties, such that no two matched users are able to increase their utilities by joining other coalitions. In this chapter, we show that the P2P ridematching problem with role flexibility is a special form of roommate problem for which there might be no stable outcome. Thus, there might be no matching and payment that can satisfy all customers. In order to circumvent this issue, we suggest subsidizing the system to help the market stability. As such we introduce a mixed integer program that outputs stable matching, role assignment, and the corresponding payoffs by adding the minimum amount of subsidy. We further propose a decomposition algorithm based on Lagrangian Relaxation (LR) to solve the proposed formulation on large scale networks. Finally, we show the connection between our proposed treatment and another type of treatment that yields near-stable outcome without injecting subsidy to the system.

In our numerical experiments with the New York taxi dataset, we showcase the advantages of subsidizing the P2P ridesharing with role flexibility. Although our proposed methodology requires injecting subsidy into the system, we found that the total amount of subsidy is negligible when compare to the increase in matching rate and social welfare of stable outcomes. Furthermore, the proposed LR algorithm shows promising results in finding a high quality solution for the proposed problem in a timely manner. Finally, through conducting extensive numerical experiments, we evaluate the performance of our methodology and investigate the impact of various parameters on a number of system-level metrics including the matching rate, the social welfare, and the level of stability.

CHAPTER 6

A Truthful Subsidy Scheme for a Peer-to-Peer Ridesharing Market With Incomplete Information

6.1 Introduction

For decades, traffic congestion during peak hours has been among the most serious issues faced by the public in many large or mid-sized urban areas worldwide. Undoubtedly, commuting from home to work in the morning, and from work to home in the evening is one of the major factors behind this issue. Unfortunately, most commuting trips are completed by solo drivers. A recent article by the U.S. News & World Report reveals that more than 76% of American commuters prefer to drive alone to work (Puentes, 2017). In the past few decades, enabled by the recent advancements in communication technologies and mobile devices, various shared mobility services have been introduced to alleviate this issue.

Peer-to-peer (P2P) ridesharing is a shared mobility platform that encourages its commuting users with similar time windows and itineraries to share their rides together (Agatz et al., 2011). Through reducing the number of vehicles on roads, P2P ridesharing can effectively mitigate traffic congestion, and thereby remove several individual and societal adverse impacts attached to it. In spite of these benefits, however, ridesharing has not seen much attention as a viable mode of transportation by commuters. (Puentes, 2017) reports that only less than 9% of commuters in the U.S. share their rides with one or more other commuters. One possible explanation is that the traveling time windows of commuters are usually tight, preventing the platform from finding profitable matches for users, and thereby preventing the system from reaching the critical mass required to improve service quality and price (Stiglic et al., 2015). In order to ameliorate this drawback, the ridesharing system can incentivize users by providing monetary subsidies to extend their time windows, and thereby, increasing their chance of being matched (Xiong et al., 2020; Tafreshian and Masoud, NDa).

In recent years, many researchers have studied different forms of P2P ridesharing to enhance the performance of these services (see e.g., Tafreshian et al. (2020) for a recent review of these studies). However, the overwhelming majority of these studies assumed systems with complete information and focused merely on improving the operations (routing, scheduling, and matching) of these services. Only a few studies considered P2P ridesharing systems as a market in which the information of users' trips are unknown and need to be solicited from the participants upon registering their trips into the system. However, the allocation of subsidy and its impact on the system has been overlooked in the pricing methods proposed in these studies. More specifically, all these studies fail to provide a systemic way of allocating a set budget among their participants.

Exploiting the rich literature of mechanism design and auction theory, this chapter aims to fill this research gap by introducing a subsidy scheme for the P2P ridesharing market with incomplete information. More precisely, we assume that the ridesharing system possesses a fix capital budget to subsidize users to extend their time windows. As such, we design a mechanism that incentivizes users to announce the true value of their private information while making sure that voluntary participation and budget constraints are satisfied.

In the rest of this chapter, we first provide a review of the literature related to the general field of mechanism design and its applications to ridesharing systems in Section 6.2. In Section 6.3, we carefully define the problem at hand, its underlying assumptions, and the economic properties of a successful mechanism. Next, we introduce a mechanism for subsidizing a ridesharing system and show its economic properties in Section 6.4. In section 6.5, we showcase the performance of the proposed mechanism using various numerical experiments. Finally, we present a summary of our findings and a number of directions for future research in Section 6.6.

6.2 Literature Review

In this section, we first present a brief overview of mechanism design followed by a number of major studies that apply this concept to determine a pricing scheme for various forms of ridesharing systems. At the end of this section, we clearly identify the contribution of this chapter.

6.2.1 Mechanism Design

In a market game with incomplete information, players hold private information that enables them to take strategic actions in their interactions (buying or selling goods/services) with other players in the hope of maximizing their own utilities (see e.g., Nisan et al. (2007)). In such markets, the set of rules that determine the market outcome and the payments of the players are referred to as a "mechanism". Mechanism design is a branch of game theory and microeconomics that helps market mediators choose proper rules to govern the interactions between selfish agents (see e.g., Börgers and Krahmer (2015); Shoham and Leyton-Brown (2008)).

In the literature of the mechanism design, there are a few key economic properties that characterize the performance of a mechanism: (1) truthfulness or dominant-strategy incentive compatibility (DSIC), (2) individual rationality (IR), (3) budget-balance (BB), (4) economic efficiency (EE), and (5) tractability or computational efficiency (CE). In a truthful mechanism, no player has an incentive to lie about their private information, regardless of what other players announce. Also, individual rationality and budget-balance respectively ensure voluntary participation for the players and no deficit for the mechanism. An efficient mechanism chooses an outcome that maximizes the social welfare. Finally, a mechanism is tractable if its result can be computed in a polynomial time. In the next section, we provide more formal definitions of these properties.

A single-item auction is a simple example of markets with incomplete information, in which a set of buyers whose true values are unknown to one another place bids to collect a single item. In his seminal work, Vickrey (1961) introduces the second-price sealed-bid auction that assigns the item to the player with the highest bid and charges the winner with the second-highest bid. Later, Clarke (1971) and Groves (1973) generalize this idea and introduce the concept of the Vickery-Clark-Groves (VCG) mechanism that is always both truthful and efficient. With certain conditions, the VCG mechanism satisfies the other properties, too. However, based on the celebrated impossibility theorem of Myerson and Satterthwaite (1983), there exist no mechanism that is IR, (weakly) BB, DSIC, and EE in general.

A bilateral trade with a single buyer and a single seller is a simple two-sided market in which the VCG mechanism may fail to satisfy the BB property. Double auctions are a more general case of bilateral trade with multiple buyers and sellers interested in buying and selling items or services. McAfee (1992) consider double auctions in which every seller can trade one single item with every buyer, and present a novel method, known as trade reduction, that is IR, weakly BB, DSIC, CE, and asymptotically EE. Dütting et al. (2017) look at a more restricted case where there exist some feasibility constraints for the outcome of a mechanism. They devise a greedy algorithm based on the deferred-acceptance auction method proposed by Milgrom and Segal (2014), and show that it is IR, weakly BB, DSIC, CE, and approximately EE. Also, Segal-Halevi et al. (2018) introduce an IR, strongly BB, DSIC, and CE mechanism based on random sampling for double auctions with a single item type. In contrary to these double auctions that assume single-minded players, Yang et al. (2011) and Feng et al. (2012) study a combinatorial double auction with heterogeneous preferences, and develop a trade reduction-based algorithm that is IR, BB, DSIC, and CE. Our proposed method in Section 6.4 is based on a combination of the last three works described here.

6.2.2 Mechanism Design in Ridesharing Markets

In the literature, P2P ridesharing has been mostly modeled as a market with complete information, and thus, the focus has been on the stability of its outcome (see e.g., Thaithatkul et al. (2015, 2017); Wang et al. (2018a); Rasulkhani and Chow (2019); Tafreshian and Masoud (2020b); Chau et al. (2020); Peng et al. (2020)). Only a few studies have considered the P2P ridesharing market with incomplete information. Kleiner et al. (2011) were among the first to use mechanism design in P2P ridesharing. For a dynamic ridesharing system, they adopt a single-sided second-price auction mechanism that ensures IR, weakly BB, and DSIC. For a ridesharing system with flexible roles, Zhao et al. (2014) introduce a VCG mechanism and show that it results in a huge deficit. In order to control the deficit, they propose two-sided reserve prices to control the deficit and define the range of reserve prices that respects the IR and IC properties of the new mechanism. Zhao et al. (2015) consider a ridesharing system where there is an uncertainty about riders and drivers undertaking their trips. By introducing a reward and penalty scheme, they propose a so called commitment-based-pay mechanism and show that it satisfies ex-post incentive compatibility (being truthful is of a player's best interest if all other players are truthful).

In a dynamic ridesharing system with flexible time windows, Zhang et al. (2015) introduce a so called discounted trade reduction method as a variant of the trade reduction method proposed by McAfee (1992). They show that this method is IR, DSIC, weakly BB, and has a larger trading volume compared to the trade reduction method. Lloret-Batlle et al. (2017a) define a P2P ridesharing market with ride-back guarantee that exploits the HOV lanes to increase the users' travel time savings. For such a system, they introduce an IR and DSIC two-stage mechanism that first classifies users as riders and drivers based on multidimensional reserve prices, and then uses the VCG mechanism to determine the matching and pricing between riders and drivers. Masoud et al. (2017a) propose a bilateral trade for the matched riders of a one-to-many P2P ridesharing system in which they can exchange their rides with newly arrived riders. They formulate the ride exchange framework as a posted-price mechanism that is IR, strongly BB, DSIC, and has a theoretical bound on expected surplus. Finally, for a ridesharing system in which riders and drivers bid on their sensitivity to schedule displacement, Li et al. (2020) introduce a VCG mechanism and show that the mechanism is IR, DSIC, CE, and EE. By sacrificing truthfulness, they further introduce a one-sided reward pricing policy that satisfies weakly BB property.

6.2.3 Our Contributions

All the reviewed studies above either use VCG-based mechanisms with no bound on deficit or use inefficient auction-based approaches that ensures no deficit. None of these studies considers the possibility of the ridesharing platform utilizing a set budget to tolerate a limited amount of deficit.

As such, the contributions of this chapter are as follows:

- For a static ridesharing market with incomplete information in which the platform possesses a set capital budget, We propose a subsidy scheme that incentivizes a targeted set of users with the help of monetary subsidies.
- We show that the proposed subsidy scheme is IR, weakly BB (with respect to the available budget), DSIC, and CE.
- We conduct a comprehensive numerical study to evaluate the performance of our proposed subsidy scheme against that of the VCG mechanism.

6.3 Problem Statement

In this chapter, we consider a static one-to-one ridesharing system that operates in a region with a finite number of stations, denoted as $S = \{s_1, s_2, ..., s_{|S|}\}$, to match riders and drivers during peak-hour period T. Let us denote the shortest-path travel time and travel distance between two stations $s_i \in S$ and $s_j \in S$ by τ_{s_i,s_j} and ρ_{s_i,s_j}, respectively. We assume that the travel times and distances are known and stored in hash tables τ and ρ. The complete list of notations for this chapter is presented in Table E.1.

On a given day, a set of riders, denoted by $R = \{r_1, r_2, ..., r_{|R|}\}$, and a set of drivers, denoted by $D = \{d_1, d_2, ..., d_{|D|}\}$, register their trips in the system before the start of period T. Two sets of R and D are mutually exclusive and their union yields the set of all users, i.e., $R \cup D = N = \{n_1, n_2, ..., n_{|N|}\}$. Each user $n_\ell \in N$ provides the system with the following information regarding their trip:

- origin station: $\mathsf{I}_{n_\ell} \in S$
- destination station: $\mathsf{J}_{n_\ell} \in S$
- desired earliest departure time: $\mathsf{T}_{n_\ell} \in T$
- desired latest arrival time: $\mathsf{Q}_{n_\ell} \in T$

We assume that users are truthful about their trip information. We further assume that drivers are flexible about their departure times and arrival times, and thus, are willing to extend their desired time window by γ_{d_j} minutes if they are compensated properly by the ridesharing platform. Note that the platform determines the value of γ_{d_j} during the ride-matching process. In order to ensure a high quality of service, however, this value cannot exceed a pre-determined value denoted as Γ. Without loss of generality, we also assume that every rider has access to a personal vehicle. Therefore, in the case of not being matched to a driver, a rider will use their own personal vehicle to complete their trip.

Every user $n_\ell \in N$ also has a private type, denoted as θ_{n_ℓ}, which is only known to the user and must be elicited by the system upon trip registration. For rider $r_i \in R$, $\theta_{r_i} = \delta_{r_i} \in \Theta_{r_i}$ represents the value of every unit of distance (mile) driven. For driver $d_j \in D$, however, $\theta_{d_j} = (\delta_{d_j}, \vartheta_{d_j}) \in \Theta_{d_j}$

consists of two values, namely the value of every mile driven, δ_{d_j}, and the value of every minute of extension in their time window, ϑ_{d_j}. Since users belong to the same population, we assume that δ_{n_ℓ} and ϑ_{d_j} follow the same distribution with cumulative distribution functions F^δ and F^ϑ, respectively. Nonetheless, we assume that F^δ and F^ϑ are unknown to users and the system only knows the mean of these distributions, denoted by $\hat{\delta}$ and $\hat{\vartheta}$. Also, note that the θ'_{n_ℓ} values that the users announce to the system may be different from their true values.

Now, let us introduce mechanism $\mathcal{M} = (f, p_1, p_2, ..., p_{|N|})$ for our ridesharing market that consists of a decision function, denoted as f, and $|N|$ payment functions, denoted as p_{n_ℓ}. Decision function $f : \Theta \rightarrow \mathcal{X}$ takes the announced private information of all users $\theta' \in \Theta$ as input and outputs ride-matching outcome $(x, t, q, \gamma) \in \mathcal{X}$, which includes matching matrix $x \in \{0, 1\}^{|R| \times |D|}$, trip start time and end time vectors $t, q \in T^{|N|}$, and time window extension vector $\gamma \in \mathbb{R}^{|D|}_{\geq 0}$. Note that $\mathcal{X}$ represents the set of feasible outcomes as defined below:

Definition 6.1. *An outcome is feasible if it satisfies the following conditions:*

1. *Time window extensions cannot exceed* Γ.
2. *Trip start and end times must lie in the extended time windows.*
3. *Every rider can be served by at most one driver.*
4. *Every driver can serve at most one rider.*

Payment function $p_{n_\ell} : \Theta \rightarrow \mathbb{R}$ outputs the payment from user n_ℓ to the system given the announced private information of all users $\theta' \in \Theta$. A negative payment for a user implies that the user receives money from the system.

With the assumption of zero payment for users who do not get matched in the system and quasi-linear preferences, the utility of users from participating in the ridesharing system can be defined as:

$$u_{n_\ell}(\theta') = v_{n_\ell}(f(\theta'), \theta_{n_\ell}) - p_{n_\ell}(\theta') \,, \qquad \forall\, n_\ell \in N \tag{6.1}$$

where, v_{n_ℓ} denotes the valuation of user n_ℓ from participating in the system and is assumed to be the difference between the cost of driving alone and the cost of sharing rides with another user. Thus, the valuations of riders and drivers for participating in the ridesharing market can be computed as:

$$v_{r_i}(f(\theta'), \theta_{r_i}) = \begin{cases} \delta_{r_i}\, \rho_{\mathrm{I}_{r_i}, \mathrm{J}_{r_i}} = V_{r_i} \,, & \text{if } \sum_{d_j \in D} x_{r_i, d_j} = 1 \,, \\ 0 \,, & \text{otherwise} \,, \end{cases} \qquad \forall\, r_i \in R \,, \tag{6.2a}$$

$$v_{d_j}(f(\theta'), \theta_{d_j}) = \begin{cases} -\delta_{d_j}\, \Delta_{r_i, d_j} - \vartheta_{d_j}\, \gamma_{d_j} = -V^{r_i}_{d_j} \,, & \text{if } \exists\, r \in R \text{ s.t. } x_{r_i, d_j} = 1 \,, \\ 0 \,, & \text{otherwise} \,, \end{cases} \qquad \forall d_j \in D \,, \tag{6.2b}$$

where $\Delta_{r_i,d_j} = (\rho_{\mathsf{I}_{d_j},\mathsf{I}_{r_i}} + \rho_{\mathsf{I}_{r_i},\mathsf{J}_{r_i}} + \rho_{\mathsf{J}_{r_i},\mathsf{J}_{d_j}} - \rho_{\mathsf{I}_{d_j},\mathsf{J}_{d_j}})$ represents the detour incurred by driver d_j to serve rider r_i through rider r_i's shortest path. Equation (6.2a) indicates that the true valuation of rider r_i, denoted as V_{r_i}, is independent of driver d_j as it is the product of their value of distance and the shortest-path length of their trip. On the contrary, the true valuation of driver d_j, denoted as $-V_{d_j}^{r_i}$, in Equation (6.2b) depends on rider r_i as it is calculated as the sum of the costs of the detour and the time window extension incurred by driver d_j to serve rider r_i. In the economics literature, we refer to V_{r_i} and $-V_{d_j}^{r_i}$ as willingness to pay and willingness to ask, respectively.

Finally, we assume that the system has a capital budget of B dollars to incentivize drivers to extend their time windows. Before stating the goal of our study, let us first provide the definitions of some important economic properties. Note that θ'_{-n_ℓ} in the following definitions denotes the announced type of all users except user n_ℓ.

Definition 6.2. *Mechanism $\mathcal{M}$ is (weakly) dominant-strategy incentive-compatible (DSIC) if*

$$u_{n_\ell}(\theta_{n_\ell}, \theta'_{-n_\ell}) \geq u_{n_\ell}(\theta'_{n_\ell}, \theta'_{-n_\ell}), \qquad \forall\, n_\ell \in N, \forall\, \theta' \in \Theta. \tag{6.3}$$

Definition 6.3. *Mechanism $\mathcal{M}$ is ex-post individually rational (IR) if*

$$u_{n_\ell}(\theta') \geq 0, \qquad \forall\, n_\ell \in N, \forall\, \theta' \in \Theta. \tag{6.4}$$

Definition 6.4. *Given a capital budget of* B*, mechanism $\mathcal{M}$ is (weakly) budget-balanced (BB) if*

$$\sum_{n_\ell \in N} p_{n_\ell}(\theta') + \mathsf{B} \geq 0, \qquad \forall\, \theta' \in \Theta. \tag{6.5}$$

Definition 6.5. *Mechanism $\mathcal{M}$ is computationally efficient (CE) if decision function f and payment functions p_{n_ℓ} yield an outcome in polynomial time with respect to their inputs.*

Definition 6.6. *Mechanism $\mathcal{M}$ is (economically) efficient (EE) if*

$$\sum_{n_\ell \in N} v_{n_\ell}(f(\theta'), \theta_{n_\ell}) \geq \sum_{n_\ell \in N} v_{n_\ell}(f'(\theta'), \theta_{n_\ell}), \qquad \forall \theta' \in \Theta, \forall\, f' \in \mathcal{X}. \tag{6.6}$$

As mentioned in Section 6.2, it is impossible to design a mechanism that satisfies all these properties. By sacrificing the economic efficiency, in this study we aim to design a subsidy scheme for our ridesharing market that is implementable and economically feasible, i.e., a mechanism that satisfies the first four properties above.

6.4 Solution Methodology

In this section, we first introduce our proposed subsidy scheme for the ridesharing market under study followed by stating its economic properties. Next, we provide a descriptive example to illustrate different steps of our subsidy scheme.

6.4.1 The Proposed Subsidy Scheme

Based on the assumptions declared in Section 6.3, the ridesharing market in this chapter can be considered as a combinatorial double auction with $|R|$ buyers (riders) and $|D|$ sellers (drivers) where sellers have heterogeneous preferences towards buyers. Let B_i denote the bid submitted by the i^{th} buyer, r_i, whose true value is V_{r_i}. Also, let $A_j = (A_j^1, ..., A_j^i, ..., A_j^{|R|})$ be the vector of asks by j^{th} seller, d_j, whose true vector of values is $V_{d_j} = (V_{d_j}^{r_1}, ..., V_{d_j}^{r_i}, ..., V_{d_j}^{r_{|R|}})$. Since these bids and asks are a function of the chosen outcome, we first introduce a pre-processing procedure, outlined in Algorithm 6.1, to calculate the bids and asks based on the information provided by the users upon their trips registration. Furthermore, the trip start, end times and time window extension of a user depends on the matched user. As such, this procedure further helps us find the set of all potential outcomes.

Let E denote the set of all pairs of a rider and a driver that can potentially share their rides together. We initialize E to be an empty set. Next, for every combination of a rider and a driver, we solve the LP in (6.7). In this problem, the decision variables are $\tilde{t}_{r_i}$, $\tilde{t}_{d_j}$, $\tilde{\gamma}_{d_j}^-$, and $\tilde{\gamma}_{d_j}^+$. Given that r_i and d_j get matched together, the first two variables represent the trip start times of the rider and the driver, respectively, and the next two variables represent the amount of extension in the driver's time window from left and right, respectively. The objective function in (6.7a) represents the driver's time window extension that must be minimized. The constraints in (6.7b)-(6.7g) ensure that the trip start and end times of both the driver and the rider lie within their time windows. After retrieving the optimal solution, we check whether the driver's time window extension is less than the threshold Γ. If the threshold is not violated, we add the (r_i, d_j) pair to set E. We further store the optimal solution of the trip start times and time window extension in $\bar{t}_{r_i}^{d_j}, \bar{t}_{d_j}^{r_i}$, and $\bar{\gamma}_{d_j}^{r_i}$. In lines 10 and 11, we compute the trip end times for the rider and the driver, respectively. Next, in line 12, we compute the ask of driver d_j for rider r_i given the announced type of driver d_j, i.e., $\theta'_{d_j} = (\delta'_{d_j}, \vartheta'_{d_j})$. Note that we set the ask A_j^i to $+\infty$ if driver d_j cannot serve rider r_i. Finally, we set the bid of rider r_i, which only depends on their type $\theta'_{r_i} = \delta'_{r_i}$.

Our proposed subsidy scheme is a mechanism $\mathcal{M}$ that takes the vector of bids, B, and the matrix of asks, A, as its inputs and yields outcome (x, t, q, γ) and vector of payments p as its output. This mechanism, some of whose elements are inspired by the work of Yang et al. (2011), Feng et al. (2012), and Segal-Halevi et al. (2018) includes the following steps:

Algorithm 6.1: The pre-processing procedure for the subsidy scheme

```
Input: $R$, $D$, $\mathsf{I}$, $\mathsf{J}$, $\mathsf{T}$, $\mathsf{Q}$, $\theta'$ .
Output: $E$, $\bar{t}$, $\bar{q}$, $\bar{\gamma}$ .
Initialize $E \leftarrow \varnothing$ ;
for $r_i \in R$ do
    for $d_j \in D$ do
        Solve the following linear problem:
            $$\min \quad \zeta = \tilde{\gamma}^-_{d_j} + \tilde{\gamma}^+_{d_j} \qquad (6.7a)$$
            $$\text{s.t.} \quad \tilde{t}_{d_j} \geq \mathsf{T}_{d_j} - \tilde{\gamma}^-_{d_j} \ , \qquad (6.7b)$$
            $$\tilde{t}_{r_i} - \tilde{t}_{d_j} \geq \tau_{\mathsf{I}_{d_j},\mathsf{I}_{r_i}} \ , \qquad (6.7c)$$
            $$\tilde{t}_{r_i} \geq \mathsf{T}_{r_i} \ , \qquad (6.7d)$$
            $$\tilde{t}_{r_i} + \tau_{\mathsf{I}_{r_i},\mathsf{J}_{r_i}} \leq \mathsf{Q}_{r_i} \ , \qquad (6.7e)$$
            $$\tilde{t}_{r_i} + \tau_{\mathsf{I}_{r_i},\mathsf{J}_{r_i}} + \tau_{\mathsf{J}_{r_i},\mathsf{J}_{d_j}} \leq \mathsf{Q}_{d_j} + \tilde{\gamma}^+_{d_j} \ , \qquad (6.7f)$$
            $$\tilde{t}_{r_i}, \tilde{t}_{d_j}, \tilde{\gamma}^-_{d_j}, \tilde{\gamma}^+_{d_j} \geq 0 \ . \qquad (6.7g)$$
        Retrieve optimal solution $(\tilde{t}^*_{r_i}, \tilde{t}^*_{d_j}, \tilde{\gamma}^{-*}_{d_j}, \tilde{\gamma}^{+*}_{d_j})$ and objective $\zeta^*$ ;
        Let $z^*_1 = \tilde{\gamma}^{-*}_{d_j} + \tilde{\gamma}^{+*}_{d_j}$;
        if $\zeta^* \leq \Gamma$ then
            Update $E \leftarrow E \cup \{(r_i, d_j)\}$ ;
            Let $\bar{t}^{d_j}_{r_i}$, $\bar{t}^{r_i}_{d_j}$, $\bar{\gamma}^{r_i}_{d_j} \leftarrow \tilde{t}^*_{r_i}$, $\tilde{t}^*_{d_j}$, $z^*_1$ ;
            Let $\bar{q}^{d_j}_{r_i} \leftarrow \tilde{t}^*_{r_i} + \tau_{\mathsf{I}_{r_i},\mathsf{J}_{r_i}}$ ;
            Let $\bar{q}^{r_i}_{d_j} \leftarrow \bar{q}^{d_j}_{r_i} + \tau_{\mathsf{J}_{r_i},\mathsf{J}_{d_j}}$ ;
            Let $A^i_j \leftarrow \delta'_{d_j} \Delta_{r_i,d_j} + \vartheta'_{d_j} \bar{\gamma}^{r_j}_{d_j}$ ;
        else
            Let $A^i_j \leftarrow +\infty$ ;
        Let $B_i \leftarrow \delta'_{r_i} \rho_{\mathsf{I}_{r_i},\mathsf{J}_{r_i}}$ ;
```

1. Find a mapping between riders and drivers independent of all announced bids and asks.
2. Determine the winning matches out of matches obtained from step 1 in which the drivers do not qualify for receiving subsidy, and find the payments.
3. Determine the winning matches out of matches obtained from step 1 whose drivers qualify for receiving subsidy, and find the subsidized payments.

In the next three subsections, we elaborate on these three steps. It is worth noting that in the case of having a tie in any step of our subsidy scheme, we give priority to the user with the lowest index with the assumption that these indices represent their order of registration in the system.

6.4.1.1 Type-Independent Mapping

In this step, we aim to find a mapping σ between riders and drivers as outlined in Algorithm 6.2. Let $\sigma : D \to R$ be a mapping function that takes the index of a rider as its inputs and yields the index of the driver matched to that rider as its output. We further let $\sigma^{-1} : R \to D$ denote the inverse of function σ. In order to ensure the truthfulness of our mechanism, this mapping must be independent of the types of the users. Given the assumption that the platform knows the mean value of distance and time extension of the users, we choose to maximize the expected gain of matching rider r_i to driver d_j, which can be computed as:

$$w_{r_i,d_j} = \hat{\delta}_{r_i}\, \rho_{\mathsf{I}_{r_i},\mathsf{J}_{r_i}} - (\hat{\delta}_{d_j}\, \Delta_{r_i,d_j} - \hat{\vartheta}\, \gamma^{r_i}_{d_j})\,, \qquad \forall\, (r_i, d_j) \in E\ . \tag{6.8}$$

As shown in Equation (6.8), the expected gain does not depend on the type of users, and thus, can be used in our mechanism to find the mapping between riders and drivers. Upon computing the expected gain for each pair of a rider and a driver in E, we can solve a maximum weighted bipartite matching problem, as in (6.10) to determine σ. In this problem, $\tilde{x}_{r_i,d_j}$ is a binary decision variable that takes the value of 1 if rider r_i and driver d_j are matched together, and zero otherwise. Also, Constraints (6.10b)-(6.10d) ensure that every rider is matched with at most one driver and every driver is matched with at most one rider. After obtaining the optimal solution, we can set the value of sigma for each rider accordingly.

In the case of platform having an accurate estimate of these values, we expect that using this objective helps us find an allocation that does not degrade the social welfare significantly. However, for the cases in which the platform does not have any estimate of these values or have poor estimates, let us introduce two other alternatives for the objective of the problem in (6.10) that do not depend on the type of users. In the literature of peer-to-peer ridesharing, a large number of studies considered maximizing the total vehicle miles travelled saving (VMTS) as the objective of their ride-matching problems. Following the literature, we can adopt the same objective function. Let w_{r_i,d_j} denote the VMTS due to rider r_i and driver d_j sharing their rides together, computed as:

$$\begin{aligned} w_{r_i,d_j} &= (\rho_{\mathsf{I}_{r_i},\mathsf{J}_{r_i}} + \rho_{\mathsf{I}_{d_j},\mathsf{J}_{d_j}}) - (\rho_{\mathsf{I}_{d_j},\mathsf{I}_{r_j}} + \rho_{\mathsf{I}_{r_i},\mathsf{J}_{r_i}} + \rho_{\mathsf{J}_{r_j},\mathsf{J}_{d_j}}) \\ &= \rho_{\mathsf{I}_{d_j},\mathsf{J}_{d_j}} - \rho_{\mathsf{I}_{d_j},\mathsf{I}_{r_i}} - \rho_{\mathsf{J}_{r_i},\mathsf{J}_{d_j}}\,, \qquad \forall\, (r_i, d_j) \in E\ . \end{aligned} \tag{6.9}$$

Also, we can choose to simply maximize the matching rate, which can be done by letting w_{r_i,d_j} be 1 for all the pairs in E. Doing so turns the problem in (6.10) into a maximum cardinality bipartite matching problem. In Section 6.5, we compare the performance of these objective functions using simulated data.

Besides obtaining mapping σ, Algorithm 6.2 helps us split the matched pair into two groups.

Algorithm 6.2: The type-independent mapping for the subsidy scheme

```
Input: $R$, $D$, $E$, $w$, $\bar{\gamma}$ .
Output: $\sigma$, $R_1$, $D_1$, $R_2$, $D_2$ .
Solve the maximum weighted bipartite matching problem:
```

$$\max \quad z = \sum_{(r_i,d_j)\in E} w_{r_i,d_j}\, \tilde{x}_{r_i,d_j} \tag{6.10a}$$

$$\text{s.t.} \quad \sum_{r_i\in R} \tilde{x}_{r_i,d_j} \leq 1\,, \qquad \forall\, d_j \in D\,, \tag{6.10b}$$

$$\sum_{d_j\in D} \tilde{x}_{r_i,d_j} \leq 1\,, \qquad \forall\, r_i \in R\,, \tag{6.10c}$$

$$\tilde{x}_{r_i,d_j} \geq 0\,, \qquad \forall\, (r_i, d_j) \in E\,. \tag{6.10d}$$

```
Retrieve optimal solution $\tilde{x}^*$ and objective $z^*$ ;
for $(r_i, d_j) \in E$ do
    if $\tilde{x}^*_{r_i,d_j} = 1$ then
        Let $\sigma(j) \leftarrow i$ ;
        if $\bar{\gamma}^{r_j}_{d_j} = 0$ then
            Update $R_1 \leftarrow R_1 \cup \{r_i\}$ ;
            Update $D_1 \leftarrow D_1 \cup \{d_j\}$ ;
        else
            Update $R_2 \leftarrow R_2 \cup \{r_i\}$ ;
            Update $D_2 \leftarrow D_2 \cup \{d_j\}$ ;
```

The first group includes those pairs in which drivers are not required to extend their time windows, and thus, are not qualified to receive any subsidy. On the contrary, the drivers in the second group are required to extend their time windows to serve the riders assigned to them, which makes them eligible to receive subsidy. Let R_1 and D_1 denote the set of riders and drivers in group 1, respectively. R_2 and D_2 can be defined similarly for group 2.

6.4.1.2 Winner Determination and Pricing for the Matches Without Subsidy

Although we find a matching between riders and drivers in the previous step, not all of them will be finalized. In this step, we use the methodology in Yang et al. (2011) to find which matches in group 1 win the auction, and set the prices for the winning users. First, let us point out that the previous step enables us to reduce the matrix of asks A into vector of asks $A^{\sigma} = (A_1^{\sigma(1)}, ..., A_j^{\sigma(j)}, ..., A_{|D|}^{\sigma(|D|)})$. Therefore, our problem reduces to a double auction in which every buyer (rider) can trade with only its assigned seller (driver). Algorithm 6.3 describes the procedure for finding the set of winning matches X_1, rider price $p_1^{\mathcal{R}}$, and driver price $p_1^{\mathcal{D}}$.

Algorithm 6.3: Winner determination and pricing for users in group 1

```
   Input: $R_1$, $D_1$, $\sigma$, $B$, $A^{\sigma}$ .
   Output: $X_1$, $p_1^{\mathcal{R}}$, $p_1^{\mathcal{D}}$ .
 1 Let $\mathcal{R}_1 \leftarrow$ Sort riders in $R_1$ in descending order based on their bids in $B$ ;
 2 Let $\mathcal{D}_1 \leftarrow$ Sort drivers in $D_1$ in ascending order based on their asks in $A^{\sigma}$ ;
 3 Initialize $k$, $X_1 \leftarrow 0$, $\varnothing$ ;
 4 for $\ell = 1, ..., |\mathcal{R}_1|$ do
 5     if $B_{i_\ell} - A_{j_\ell}^{\sigma(j_\ell)} \geq 0$ then
 6         Update $k \leftarrow \ell$ ;
 7 if $k \geq 2$ then
 8     Initialize $\alpha$, $\beta \leftarrow k$, $k$ ;
 9     for $\ell = k, k+1, ..., |\mathcal{R}_1|$ do
10         if $B_{i_\ell} - A_{j_k}^{\sigma(j_k)} \geq 0$ then
11             Update $\alpha \leftarrow \ell$ ;
12         if $B_{i_k} - A_{j_\ell}^{\sigma(j_\ell)} \geq 0$ then
13             Update $\beta \leftarrow \ell$ ;
14     Initialize $X_1^{\alpha}$, $X_1^{\beta} \leftarrow \varnothing$, $\varnothing$ ;
15     for $d_j \in D_1$ do
16         if $(B_{\sigma(j)} \geq B_{i_\alpha}) \wedge (A_j^{\sigma(j)} \leq A_{j_k}^{\sigma(j_k)}) \wedge (\sigma(j) \neq i_\alpha) \wedge (j \neq j_k)$ then
17             Update $X_1^{\alpha} \leftarrow X_1^{\alpha} \cup \{(r_{\sigma(j)}, d_j)\}$ ;
18         if $(B_{\sigma(j)} \geq B_{i_k}) \wedge (A_j^{\sigma(j)} \leq A_{j_\beta}^{\sigma(j_\beta)}) \wedge (\sigma(j) \neq i_k) \wedge (j \neq j_\beta)$ then
19             Update $X_1^{\beta} \leftarrow X_1^{\beta} \cup \{(r_{\sigma(j)}, d_j)\}$ ;
20     if $|X_1^{\alpha}| \geq |X_1^{\beta}|$ then
21         Let $X_1$, $p_1^{\mathcal{R}}$, $p_1^{\mathcal{D}} \leftarrow X_1^{\alpha}$, $B_{i_\alpha}$, $A_{j_k}^{\sigma(j_k)}$ ;
22     else
23         Let $X_1$, $p_1^{\mathcal{R}}$, $p_1^{\mathcal{D}} \leftarrow X_1^{\beta}$, $B_{i_k}$, $A_{j_\beta}^{\sigma(j_\beta)}$ ;
```

The algorithm starts with defining two sorted sequences of $\mathcal{R}_1 = \langle r_{i_1}, ..., r_{i_k}, ..., r_{i_{|R_1|}} \rangle$ and $\mathcal{D}_1 = \langle d_{j_1}, ..., d_{i_k}, ..., d_{j_{|D_1|}} \rangle$ for the riders and drivers based on their announced bids and asks. Thus, $r_{i_1} \in \mathcal{R}_1$ represents the rider with the largest bid, while $d_{j_1} \in \mathcal{D}_1$ represents the driver with the smallest ask. Next, in lines 3-6, we find the largest k such that the bid of the k^{th} rider in $\mathcal{R}_1$ is greater than or equal to the ask of the k^{th} driver in $\mathcal{D}_1$. In the case of having $k < 2$, we finalize no match in group 1 and end the algorithm. Otherwise, we find the largest $\alpha \geq k$ such that the bid of the α^{th} rider in $\mathcal{R}_1$ is greater than or equal to the ask of the k^{th} driver in $\mathcal{D}_1$. Also, we find the largest $\beta \geq k$ such that the bid of the k^{th} rider in $\mathcal{R}_1$ is greater than or equal to the ask of the β^{th} driver in $\mathcal{D}_1$. Next, in lines 14-19, we compute two set of matches X_1^{α} and X_1^{β}. X_1^{α} includes all the pairs of a rider and a driver such that the bid of the rider is among the first $(\alpha - 1)$ bids of $\mathcal{R}_1$, and the ask of the driver is among the first $(k - 1)$ asks of $\mathcal{D}_1$. Also, X^{β} includes all the pairs of a rider and a driver such that the bid of the rider is among the first $(k - 1)$ bids of $\mathcal{R}_1$, and the ask of the driver is among the first $(\beta - 1)$ asks of $\mathcal{D}_1$. Finally, we compare the size of X_1^{α} and X_1^{β}. If $|X_1^{\alpha}| \geq |X_1^{\beta}|$, we finalize the matches in X_1^{α}, require the winning riders to pay the bid of the α^{th} rider in $\mathcal{R}_1$ to the system, and pay the ask of the k^{th} driver in $\mathcal{D}_1$ to the winning drivers. Otherwise, we finalize the matches in X_1^{β}, require the winning riders to pay the bid of the k^{th} rider in $\mathcal{R}_1$ to the system, and pay the ask of the β^{th} driver in $\mathcal{D}_1$ to the winning drivers.

6.4.1.3 Winner Determination and Pricing for Subsidized Matches

In this step, we aim to find the set of winning matches in group 2, and set the rider and driver prices given the assumption that the system has B dollars to subsidize the payment to the drivers. Since the drivers have two sources of payment, we adopt the method of random market halving proposed by Segal-Halevi et al. (2016, 2018) to devise a two-stage process that determines the prices, subsidies and winning matches for the users in group 2 (see Algorithms 6.4 and 6.5).

Algorithm 6.4 starts with randomly dividing the riders in group 2 into two submarkets of equal size: the left market, denoted by L, and the right market, denoted by R. We denote the set of riders in the left market by R_2^{L}, and in the right market by R_2^{R}. Also, we let $\mathcal{R}_2^{\mathsf{L}}$ denote the sequence of riders in the left market in descending order of their bids. Now, let us define the trade price in the right market, denoted by p_2^{R}, as the payment of every winning rider in this submarket to their assigned driver. In lines 3-13 of Algorithm 6.4, we determine this trade price using only the bids of the riders in the *left* market and asks of the drivers assigned to them. More specifically, let G denote the social welfare of the users in the left market and initialize it to zero. Next, for every rider r_{i_m} in $\mathcal{R}_2^{\mathsf{L}}$, we find the riders in R_2^{L} whose bids are at least as high as the bid of rider r_{i_m}. Let D_2' be the set of all drivers matched with these riders, and $\mathcal{D}_2'$ be the sorted sequence of these drivers in ascending order of their asks. In lines 7-13, we find the largest k such that subsidizing all the first k drivers in $\mathcal{D}_2'$ by the difference between the ask of the driver and the bid of i_m^{th} rider does not violate half of the budget. Finally, we compute the social welfare of the first k drivers in $\mathcal{D}_2'$ and their assigned riders. If the computed value is at least as high as G, we update G to the computed social welfare and update the trade price of the right market to be the bid of rider r_{i_m}. In lines 14-25, we follow the same procedure in the right market to determine the trade price in the left market, denoted by p_2^{L}.

Given the trade prices found in the previous stage, we use the procedure described in Algorithm 6.5 to find the winning matches in group 2 and the subsidy paid by the system. Let D_2^{L} denote the set of drivers whose assigned riders belong to the left market and announced a bid at least as high as the trade price in that market, p_2^{L}. We sort these drivers in ascending order of their asks and store the result in $\mathcal{D}_2^{\mathsf{L}}$. Let s_2^{L} denote the subsidy paid by the system to a winning driver in the left market. In

Algorithm 6.4: First stage of winner determination and pricing for users in group 2

```
   Input: $R_2$, $D_2$, $\sigma$, $B$, $A^{\sigma}$ .
   Output: $R_2^{\mathsf{L}}$, $R_2^{\mathsf{R}}$, $p_2^{\mathsf{L}}$, $p_2^{\mathsf{R}}$ .
1  $R_2^{\mathsf{L}}, R_2^{\mathsf{R}} \leftarrow$ Randomly divide riders in $R_2$ into two submarkets of left ($\mathsf{L}$) and right ($\mathsf{R}$) ;
2  $\mathcal{R}_2^{\mathsf{L}} \leftarrow$ Sort riders in $R_2^{\mathsf{L}}$ in descending order based on their bids in $B$ ;
3  Initialize $G, p_2^{\mathsf{R}} \leftarrow 0,\ 0$ ;
4  for $m = 1, ..., |\mathcal{R}_2^{\mathsf{L}}|$ do
5      Let $D_2' \leftarrow \{d_j : (r_{\sigma(j)} \in R_2^{\mathsf{L}}) \wedge (B_{\sigma(j)} \geq B_{i_m})\}$ ;
6      Let $\mathcal{D}_2' \leftarrow$ Sort drivers in $D'$ in ascending order based on their asks in $A^{\sigma}$ ;
7      Initialize $k \leftarrow 1$ ;
8      for $\ell = 1, ..., |\mathcal{D}_2'|$ do
9          if $\ell\,(A_{j_\ell}^{\sigma(j_\ell)} - B_{i_m}) \leq \frac{\mathsf{B}}{2}$ then
10             Update $k \leftarrow \ell$ ;
11     if $\sum_{\ell=1}^{k} (B_{\sigma(j_\ell)} - A_{j_\ell}^{\sigma(j_\ell)}) \geq G$ then
12         Update $G \leftarrow \sum_{\ell=1}^{k} (B_{\sigma(j_\ell)} - A_{j_\ell}^{\sigma(j_\ell)})$ ;
13         Update $p_2^{\mathsf{R}} \leftarrow B_{i_m}$ ;
14 $\mathcal{R}_2^{\mathsf{R}} \leftarrow$ Sort riders in $R_2^{\mathsf{R}}$ in descending order based on their bids in $B$ ;
15 Initialize $G, p_2^{\mathsf{L}} \leftarrow 0,\ 0$ ;
16 for $m = 1, ..., |\mathcal{R}_2^{\mathsf{R}}|$ do
17     Let $D_2' \leftarrow \{d_j : (r_{\sigma(j)} \in R_2^{\mathsf{R}}) \wedge (B_{\sigma(j)} \geq B_{i_m})\}$ ;
18     Let $\mathcal{D}_2' \leftarrow$ Sort drivers in $D'$ in ascending order based on their asks in $A^{\sigma}$ ;
19     Initialize $k \leftarrow 1$ ;
20     for $\ell = 1, ..., |\mathcal{D}_2'|$ do
21         if $\ell\,(A_{j_\ell}^{\sigma(j_\ell)} - B_{i_m}) \leq \frac{\mathsf{B}}{2}$ then
22             Update $k \leftarrow \ell$ ;
23     if $\sum_{\ell=1}^{k} (B_{\sigma(j_\ell)} - A_{j_\ell}^{\sigma(j_\ell)}) \geq G$ then
24         Update $G \leftarrow \sum_{\ell=1}^{k} (B_{\sigma(j_\ell)} - A_{j_\ell}^{\sigma(j_\ell)})$ ;
25         Update $p_2^{\mathsf{L}} \leftarrow B_{i_m}$ ;
```

lines 4-6, we check if we can subsidize all the drivers in $\mathcal{D}_2^{\mathsf{L}}$ equally with half of the budget. If yes, we let s_2^{L} be $\frac{\mathsf{B}}{2\,|\mathcal{D}_2^{\mathsf{L}}|}$. Also, we let k be $|\mathcal{D}_2^{\mathsf{L}}|$ and go to line 12. Otherwise, if the size of D_2^{L} is greater than 2, we find the largest k such that subsidizing the first k drivers in $\mathcal{D}_2^{\mathsf{L}}$ by the difference between the ask of the $(k+1)^{th}$ driver and the trade price does not violate half of the budget. We also let the subsidy for the drivers in the left market be the difference between the ask of the $(k+1)^{th}$ driver in $\mathcal{D}_2^{\mathsf{L}}$ and the left market trade price if positive, and zero otherwise. In line 12, we define X^{L} as the set

Algorithm 6.5: Second stage of winner determination and pricing for users in group 2

```
Input: $R_2$, $D_2$, $\sigma$, $B$, $A^{\sigma}$, $R_2^{\mathsf{L}}$, $R_2^{\mathsf{R}}$, $p_2^{\mathsf{L}}$, $p_2^{\mathsf{R}}$ .
Output: $X_2$, $s_2^{\mathsf{L}}$, $s_2^{\mathsf{R}}$ .
Let $D_2^{\mathsf{L}} \leftarrow \{d_j : (r_{\sigma(j)} \in R_2^{\mathsf{L}}) \wedge (B_{\sigma(j)} \geq p_2^{\mathsf{L}})\}$ ;
Let $\mathcal{D}_2^{\mathsf{L}} \leftarrow$ Sort drivers in $D_2^{\mathsf{L}}$ in ascending order based on their asks in $A^{\sigma}$ ;
Initialize $k,\ s_2^{\mathsf{L}} \leftarrow 0,\ 0$ ;
if $(A_{j_{|\mathcal{D}_2^{\mathsf{L}}|}}^{\sigma(j_{|\mathcal{D}_2^{\mathsf{L}}|})} - p_2^{\mathsf{L}}) \leq \frac{\mathsf{B}}{2\,|\mathcal{D}_2^{\mathsf{L}}|}$ then
    Update $k \leftarrow |\mathcal{D}_2^{\mathsf{L}}|$ ;
    Update $s_2^{\mathsf{L}} \leftarrow \frac{\mathsf{B}}{2\,|\mathcal{D}_2^{\mathsf{L}}|}$ ;
else if $|\mathcal{D}_2^{\mathsf{L}}| \geq 2$ then
    for $\ell = 1, ..., |\mathcal{D}_2^{\mathsf{L}}| - 1$ do
        if $\ell\,(A_{j_{\ell+1}}^{\sigma(j_{\ell+1})} - p_2^{\mathsf{L}}) \leq \frac{\mathsf{B}}{2}$ then
            Update $k \leftarrow \ell$ ;
    Update $s_2^{\mathsf{L}} \leftarrow \max\{0, A_{j_k}^{\sigma(j_k)} - p_2^{\mathsf{L}}\}$ ;
Initialize $X_2^{\mathsf{L}} \leftarrow \varnothing$ ;
if $k \geq 1$ then
    for $\ell = 1, ..., k$ do
        Update $X_2^{\mathsf{L}} \leftarrow X_2^{\mathsf{L}} \cup \{(r_{\sigma(j_\ell)}, d_{j_\ell})\}$ ;
Let $D_2^{\mathsf{R}} \leftarrow \{d_j : (r_{\sigma(j)} \in R_2^{\mathsf{R}}) \wedge (B_{\sigma(j)} \geq p_2^{\mathsf{R}})\}$ ;
Let $\mathcal{D}_2^{\mathsf{R}} \leftarrow$ Sort drivers in $D_2^{\mathsf{R}}$ in ascending order based on their asks in $A^{\sigma}$ ;
Initialize $k,\ s_2^{\mathsf{R}} \leftarrow 0,\ 0$;
if $(A_{j_{|\mathcal{D}_2^{\mathsf{R}}|}}^{\sigma(j_{|\mathcal{D}_2^{\mathsf{R}}|})} - p_2^{\mathsf{R}}) \leq \frac{\mathsf{B}}{2\,|\mathcal{D}_2^{\mathsf{R}}|}$ then
    Update $k \leftarrow |\mathcal{D}_2^{\mathsf{R}}|$ ;
    Update $s_2^{\mathsf{R}} \leftarrow \frac{\mathsf{B}}{2\,|\mathcal{D}_2^{\mathsf{R}}|}$ ;
else if $|\mathcal{D}_2^{\mathsf{R}}| \geq 2$ then
    for $\ell = 1, ..., |\mathcal{D}_2^{\mathsf{R}}| - 1$ do
        if $\ell\,(A_{j_{\ell+1}}^{\sigma(j_{\ell+1})} - p_2^{\mathsf{R}}) \leq \frac{\mathsf{B}}{2}$ then
            Update $k \leftarrow \ell$ ;
    Update $s_2^{\mathsf{R}} \leftarrow \max\{0, A_{j_k}^{\sigma(j_k)} - p_2^{\mathsf{R}}\}$ ;
Initialize $X_2^{\mathsf{R}} \leftarrow \varnothing$ ;
if $k \geq 1$ then
    for $\ell = 1, ..., k$ do
        Update $X_2^{\mathsf{R}} \leftarrow X_2^{\mathsf{R}} \cup \{(r_{\sigma(j_\ell)}, d_{j_\ell})\}$ ;
Let $X_2 \leftarrow X_2^{\mathsf{L}} \cup X_2^{\mathsf{R}}$ ;
```

of winning matches in left market and initialize it as an empty set. If k is not positive, we do not finalize any match. Otherwise, the first k drivers in $\mathcal{D}_2^{\mathsf{L}}$ and their assigned riders will be the winning users. In lines 16-30 of this algorithm, we follow the same procedure to find X^{R} and s^{R} for the right market. Finally, the union of X^{L} and X^{R} gives us the set of winning matches in group 2, denoted by X_2.

6.4.2 Economic Properties of the Subsidy Scheme

In this subsection, we present the economic properties of the proposed subsidy scheme.

Proposition 6.1. *The proposed subsidy scheme is individually rational.*

Proof. For each winning pair (r_i, d_j) in group 1, the rider pays $p_1^{\mathcal{R}} \leq B_i$ and the driver receives $p_1^{\mathcal{D}} \geq A_j^i$. Also, the losing users pay nothing. As a result, their utilities are always non-negative. Also, for each winning pair (r_i, d_j) in the left market of group 2, the rider pays $p_2^{\mathsf{L}} < B_i$ and the driver receives $p_2^{\mathsf{L}} + s^{\mathsf{L}} > A_j^i$ from the rider and the system. Similarly, for each winning pair (r_i, d_j) in the right market of group 2, the rider pays $p_2^{\mathsf{R}} < B_i$ and the driver receives $p_2^{\mathsf{R}} + s^{\mathsf{R}} > A_j^i$ from the rider and the system, and the losing pairs riders and drivers pay nothing. In both markets, the losing users pay nothing. Thus, the utility of all users in group 2 are also non-negative, and the result follows. □

Proposition 6.2. *Given any budget* B*, the proposed subsidy scheme is weakly budget-balanced.*

Proof. In group 1, all winning riders pay $p_1^{\mathcal{R}}$, which is always greater than or equal to the amount paid to every driver, i.e., $p_1^{\mathcal{R}} \geq p_1^{\mathcal{D}}$. Thus, no winning pair incurs any deficit. In group 2, the sum of the deficits of all pairs in each market is bounded by half of the budget. Therefore, the total deficit of winning pairs in group 2 cannot exceed B, and the result follows. □

Before presenting the next proposition and its associated lemmas, let us point out that the mapping in Algorithm 6.2 is type-independent, and hence, the rider/driver assigned to every driver/rider as well as the group to which they belong would not change if their bids or asks were different.

Lemma 6.1. *If a rider wins with bid B_i, the rider can also win with a higher bid $\tilde{B}_i > B_i$, and if a driver wins with ask $A_j^{\sigma(j)}$, the rider can also win with a lower ask $\tilde{A}_j^{\sigma(j)} < A_j^{\sigma(j)}$.*

Proof. In both groups, a rider wins if their bid is greater than a threshold, which is equal to the highest bid of the losing riders in group 1, and the trade price of the other market in group 2. Therefore, if a winning rider had announced a higher bid, their bid would have still been greater than the threshold, and thus, they would have still won the auction. Similarly, a driver wins if their ask is lower than a threshold, which is equal to the lowest ask of the losing drivers in group 1, and the difference between the lowest ask of the losing drivers and the market trade price. Therefore, if a winning driver had announced a lower ask, their ask would have still been lower than the threshold, and thus, they would have won the auction. This result is true for any random sampling presented in Algorithm 6.4. □

Lemma 6.2. *If a rider wins with both bids of B_i and $\tilde{B}_i \neq B_i$, the rider pays the same amount, and if a driver wins with both asks of $A_j^{\sigma(j)}$ and $\tilde{A}_j^{\sigma(j)} \neq A_j^{\sigma(j)}$, the payment to the driver is the same.*

Proof. As mentioned in the proof of Lemma 6.1, the winning users in group 1 pay or receive money based on a threshold that only depends on the highest bid or the lowest ask of losing users. Thus, the only way for a user to change the payment is to bid or ask beyond the threshold, in which case the user will no longer be a winner. The same reasoning applies to the drivers in group 2. Also, for the riders in group 2, there is no way to change their payment as it comes from the bids and asks of the users in the opposite market. Note that this result holds for any random sampling used for obtaining the left and right market. □

Lemma 6.3. *The proposed subsidy scheme is truthful for riders.*

Proof. Let us assume rider r_i who has the true value of V_i bids $B_i \neq V_i$. In what follows we consider different possible cases of bidding different from the true value, and show that in all cases the utility of the rider will not exceed the utility of telling the truth denoted by $u_i(V_i)$. These cases are summarized in Table 6.1.

1. If the rider announces a higher bid, there are 3 possible subcases based on Lemma 6.1:
   (a) If the rider wins with both B_i and V_i, then based on Lemma 6.2, the rider pays the same amount. Thus, the utility does not change, no matter to which group the rider belongs.
   (b) If the rider wins with B_i but loses with V_i, the rider pays the bid of a rider whose bid is at least as high as the true value of rider (V_i). Therefore, the utility of rider cannot be positive, no matter to which group the rider belongs.
   (c) If the rider loses with both B_i and V_i, then the utilities are the same and equal to zero.
2. If the rider announces a lower bid, there are 3 possible subcases based on Lemma 6.1:
   (a) If the rider wins with both B_i and V_i, then based on Lemma 6.2, the rider pays the same amount. Therefore, the utility of a rider in group 1 or group 2 does not change.
   (b) If the rider wins with V_i but loses with B_i, the rider's utility with B_i will be zero while the utility of being truthful will be non-negative. This is the case for both group 1 and group 2.
   (c) If the rider loses with both B_i and V_i, then again the utilities are the same and equal to zero.

Table 6.1: Analysis of rider truthfulness for different possible cases

Cases		**Results**	
		$r_i \in R_1$	$r_i \in R_2$
2. $B_i > V_i$	(a) B_i wins, V_i wins	$0 \leq u_i(B_i) = u_i(V_i)$	$0 \leq u_i(B_i) = u_i(V_i)$
	(b) B_i wins, V_i loses	$u_i(B_i) \leq u_i(V_i) = 0$	$u_i(B_i) \leq u_i(V_i) = 0$
	(c) B_i loses, V_i loses	$u_i(B_i) = u_i(V_i) = 0$	$u_i(B_i) = u_i(V_i) = 0$
3. $B_i < V_i$	(a) B_i wins, V_i wins	$0 \leq u_i(B_i) = u_i(V_i)$	$0 \leq u_i(B_i) = u_i(V_i)$
	(b) B_i loses, V_i wins	$0 = u_i(B_i) \leq u_i(V_i)$	$0 = u_i(B_i) \leq u_i(V_i)$
	(c) B_i loses, V_i loses	$u_i(B_i) = u_i(V_i) = 0$	$u_i(B_i) = u_i(V_i) = 0$

□

Lemma 6.4. *The proposed subsidy scheme is truthful for drivers.*

Proof. Let us assume driver d_j who has the true value of $V_j^{\sigma(j)}$ asks $A_j^{\sigma(j)} \neq V_j^{\sigma(j)}$. Table 6.2 summarizes the impact of lying on the utility of the driver for different possible cases. As this table suggests, in all cases the utility of the driver will not exceed the utility of telling the truth denoted by $u_j(V_j^{\sigma(j)})$. Note that due to the similarity between the analyses of a rider and a driver, we have omitted the explanation. □

Table 6.2: Analysis of driver truthfulness for different possible cases

Cases		**Results**	
		$r_{\sigma(j)} \in R_1$	$r_{\sigma(j)} \in R_2$
2. $A_j^{\sigma(j)} > V_j^{\sigma(j)}$	(a) $A_j^{\sigma(j)}$ wins, $V_j^{\sigma(j)}$ wins	$0 \leq u_j(A_j^{\sigma(j)}) = u_j(V_j^{\sigma(j)})$	$0 \leq u_j(A_j^{\sigma(j)}) = u_j(V_j^{\sigma(j)})$
	(b) $A_j^{\sigma(j)}$ loses, $V_j^{\sigma(j)}$ wins	$0 = u_j(A_j^{\sigma(j)}) \leq u_j(V_j^{\sigma(j)})$	$0 = u_j(A_j^{\sigma(j)}) \leq u_j(V_j^{\sigma(j)})$
	(c) $A_j^{\sigma(j)}$ loses, $V_j^{\sigma(j)}$ loses	$u_j(A_j^{\sigma(j)}) = u_j(V_j^{\sigma(j)}) = 0$	$u_j(A_j^{\sigma(j)}) = u_j(V_j^{\sigma(j)}) = 0$
3. $A_j^{\sigma(j)} < V_j^{\sigma(j)}$	(a) $A_j^{\sigma(j)}$ wins, $V_j^{\sigma(j)}$ wins	$0 \leq u_j(A_j^{\sigma(j)}) = u_j(A_j^{\sigma(j)})$	$0 \leq u_j(A_j^{\sigma(j)}) = u_j(V_j^{\sigma(j)})$
	(b) $A_j^{\sigma(j)}$ wins, $V_j^{\sigma(j)}$ loses	$u_j(A_j^{\sigma(j)}) \leq u_j(V_j^{\sigma(j)}) = 0$	$u_j(A_j^{\sigma(j)}) \leq u_j(V_j^{\sigma(j)}) = 0$
	(c) $A_j^{\sigma(j)}$ loses, $V_j^{\sigma(j)}$ loses	$u_j(A_j^{\sigma(j)}) = u_j(V_j^{\sigma(j)}) = 0$	$u_j(A_j^{\sigma(j)}) = u_j(V_j^{\sigma(j)}) = 0$

Proposition 6.3. *The proposed subsidy scheme is weakly dominant-strategy incentive compatible.*

Proof. Based on Lemmas 6.3 and 6.4, truthfulness is the dominant strategy for both riders and drivers, and thus, the proposed mechanism is weakly dominant-strategy incentive compatible. □

Proposition 6.4. *The proposed subsidy scheme is computationally efficient.*

Proof. The pre-processing procedure in Algorithm 6.1 can be performed in $\mathcal{O}(|R||D|)$. Also, we can use the Fredman-Tarjan algorithm to solve the maximum weighted bipartite matching problem in Algorithm 6.2, which is known to have a running time complexity of $\mathcal{O}(|N||E| + |N|^2 \log |N|)$. In Algorithm 6.3, sorting riders and drivers takes $\mathcal{O}(|R_1| \log |R_1|)$ and $\mathcal{O}(|D_1| \log |D_1|)$ running time. The rest of the algorithm runs in $\mathcal{O}(|R_1|)$. Algorithm 6.4 runs in $\mathcal{O}(|R_2||D_2| \log |D_2|)$. Finally, Algorithm 6.5 solves in $\mathcal{O}(|D_2| \log |D_2|)$. Combining all these, the worst-case running time complexity of our proposed algorithm is equivalent to that of the maximum weighted bipartite matching which runs in polynomial time, and the result follows. □

6.4.3 A Descriptive Example

In this subsection, we showcase the different steps of our proposed subsidy scheme using a small example. Let us consider a ridesharing system that operates on the well-known Nguyen-Dupuis transportation network. In Figure 6.1, we show an example of this network with arbitrary travel times and driving distances for every directed link. The shortest-path travel time, τ, and driving distance, δ, between every two stations are presented in Tables E.2 and E.3, respectively.

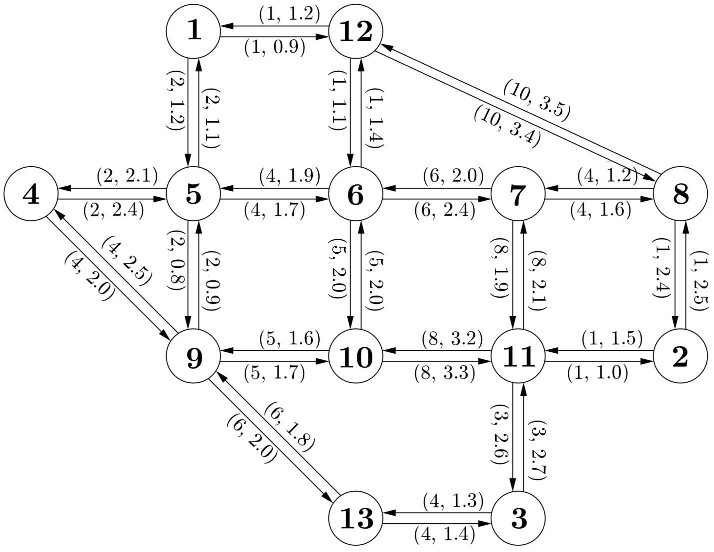


Figure 6.1: The Nguyen-Dupuis Transportation Network. The tuple on each link shows the shortest-path travel time (in minutes) and driving distance (in miles) on that link.

For the sake of this example, we assume that 14 riders and 13 drivers register their trips in a given day. The trip information and announced private values are provided in Tables 6.3 and 6.4 for the riders and the drivers, respectively. In this example, the maximum allowed time window extension for a driver is assumed to be 5 minutes. We also assume that the ridesharing system has 10 dollars to subsidize the users.

Table 6.3: The trip information for the riders in the small example

	r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	r_9	r_{10}	r_{11}	r_{12}	r_{13}	r_{14}
I	1	1	4	12	5	4	7	9	5	8	10	6	5	6
J	2	13	2	13	2	9	3	7	7	6	12	10	10	3
T	7:00	7:00	7:00	7:00	7:00	7:05	7:05	7:05	7:05	7:10	7:10	7:10	7:10	7:10
Q	7:16	7:14.0	7:21	7:15	7:16	7:13	7:19	7:20	7:19	7:24	7:19	7:18	7:32	7:20
δ	1.5	2.0	2.5	2.0	3.5	3.0	2.0	3.5	2.0	3.0	2.5	3.0	2.5	3.0

Table 6.4: The trip information for the drivers in the small example

	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}	d_{11}	d_{12}	d_{13}
I	1	4	1	1	4	4	1	6	2	11	1	12	9
J	13	3	11	3	8	8	11	9	6	1	11	3	10
T	7:00	7:00	7:05	7:05	7:05	7:05	7:05	7:10	7:10	7:10	7:10	7:10	7:10
Q	7:13	7:18	7:27	7:22	7:24	7:23	7:29	7:20	7:24	7:26	7:33	7:32	7:21
δ	1.5	3.0	3.5	2.5	2.5	3.0	3.0	3.5	2.0	1.0	3.0	3.0	2.0
ϑ	0.4	0.8	0.5	0.6	0.8	0.7	0.4	0.1	0.2	0.4	0.8	0.3	0.6

As a result of applying Algorithm 6.1 for all combinations of a rider and a driver, we can obtain the optimal time window extension of every driver conditional on sharing rides with any of the riders as shown in Table 6.5. Note that for the combinations of a rider and a driver that cannot be made spatio-temporally feasible, we do not have a value. Upon this pre-processing procedure, we can compute the bid vector of all the riders and the ask matrix of all the drivers as shown in Tables 6.6 and 6.7.

Table 6.5: The amount of time window extension for drivers conditional on their assigned rider

	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}	d_{11}	d_{12}	d_{13}
r_1	–	2	1	1	5	5	1	–	–	–	–	–	–
r_2	0	0	1	1	5	5	1	–	–	–	–	–	–
r_3	–	4	–	–	2	2	–	–	–	–	–	–	–
r_4	0	–	2	2	–	–	2	–	–	–	–	–	–
r_5	–	2	5	5	5	5	5	–	–	–	–	–	–
r_6	2	1	0	1	0	1	0	–	–	–	5	–	5
r_7	–	–	–	–	–	–	–	–	–	–	–	–	–
r_8	–	–	2	–	1	2	1	–	–	–	–	–	–
r_9	–	–	0	–	0	0	0	–	–	–	3	4	–
r_{10}	–	–	–	–	–	–	–	–	0	2	–	–	–
r_{11}	–	–	–	–	–	–	–	–	–	5	–	–	–
r_{12}	–	–	0	5	–	–	0	0	–	–	3	0	3
r_{13}	–	–	0	2	5	–	0	2	–	–	0	1	0
r_{14}	–	–	4	–	–	–	2	–	–	–	4	0	–

Before showing the steps of our subsidy scheme, first let us discuss the implementation of the VCG mechanism on this example, which yields the optimal social welfare. We use the VCG results

Table 6.6: Bid vector of riders, B, in the small example

r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	r_9	r_{10}	r_{11}	r_{12}	r_{13}	r_{14}
13.4	8.00	20.00	10.40	27.30	6.00	9.00	17.50	8.20	9.60	8.50	6.00	9.25	20.70

Table 6.7: Ask matrix of drivers A in the small example

	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}	d_{11}	d_{12}	d_{13}
r_1	∞	24.40	5.75	10.85	19.00	21.50	4.90	∞	∞	∞	∞	∞	∞
r_2	0.00	14.70	14.85	4.10	36.80	31.25	12.70	∞	∞	∞	∞	∞	∞
r_3	∞	15.50	∞	∞	7.85	8.9	∞	∞	∞	∞	∞	∞	∞
r_4	1.35	∞	18.50	6.95	∞	∞	15.80	∞	∞	∞	∞	∞	∞
r_5	∞	21.10	11.95	16.25	16.25	18.20	10.10	∞	∞	∞	∞	∞	∞
r_6	8.75	11.00	11.00	17.35	15.75	19.60	24.90	∞	∞	∞	28.90	∞	11.40
r_7	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
r_8	∞	∞	14.65	∞	9.80	12.20	12.10	∞	∞	∞	∞	∞	∞
r_9	∞	∞	10.85	∞	10.00	12.00	9.30	9.30	∞	∞	11.70	21.90	∞
r_{10}	∞	∞	∞	∞	∞	∞	∞	∞	7.5	7.3	∞	∞	∞
r_{11}	∞	∞	∞	∞	∞	∞	∞	∞	∞	6.4	∞	∞	∞
r_{12}	∞	∞	21.70	25.00	∞	∞	18.60	5.60	∞	∞	21.00	21.00	7
r_{13}	∞	∞	15.75	∞	∞	∞	13.50	14.20	∞	∞	13.50	25.20	1.80
r_{14}	∞	∞	21.60	∞	∞	∞	17.60	∞	∞	∞	20.00	3.30	∞

as a benchmark against which our proposed mechanism can be compared. For finding the VCG outcome, we solve the maximum weighted bipartite matching problem (6.10) in Algorithm 6.2 with w_{r_i,d_j} being the welfare of every pair of a rider and a driver computed as

$$w_{r_i,d_j} = B_i - A^i_j\,, \qquad \forall\,(r_i, d_j) \in E\,. \tag{6.11}$$

The optimal matching for the small example is demonstrated in Figure 6.2. Let SW^1 denote the objective of this problem. The optimal social welfare SW^{1*} is equal to 86.05 in this example. In order to find the payment of every rider r_i and every driver d_j, we calculate the Clark pivot as follows:

$$p_{r_i} = \mathrm{SW}^{1*}_{-r_i} - \mathrm{SW}^{1*} + B_i \sum_{d_j \in D} \tilde{x}^*_{r_i,d_j}\,, \qquad \forall\, r_i \in R\,, \tag{6.12a}$$

$$p_{d_j} = \mathrm{SW}^{1*}_{-d_j} - \mathrm{SW}^{1*} + \sum_{r_i \in R} A^i_j\, \tilde{x}^*_{r_i,d_j}\,, \qquad \forall\, d_j \in D\,, \tag{6.12b}$$

where $\mathrm{SW}^{1*}_{-n_\ell}$ represents the optimal social welfare of the system excluding user n_ℓ. Also, $\tilde{x}^*_{r_i,d_j}$ represents the optimal matching solution between rider r_i and driver d_j. The VCG payments of the users in the small example are presented in Table 6.8. Adding these payments together and negating the result reveals that the system incurs a payment deficit of $\mathrm{PD}^1 = 66.75 > 10 = \mathsf{B}$ dollars to implement the VCG outcome. Note that this value is more than 75% of the optimal social welfare.

Therefore, although the VCG mechanism satisfies the IR, DSIC, EE, and CE properties, it fails to satisfy the BB property, which indicates that the VCG mechanism is infeasible for a system with a capital budget of maximum 10 dollars.

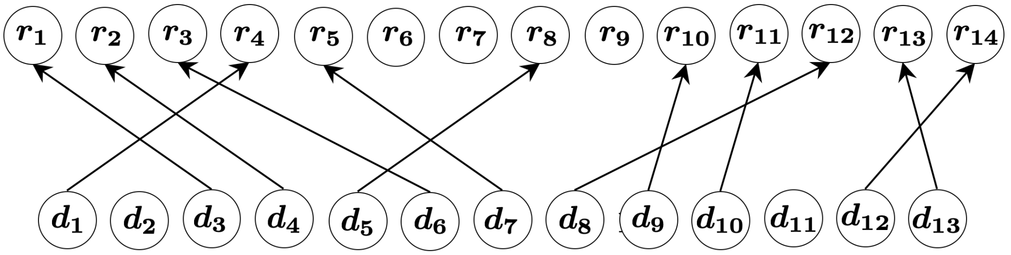


Figure 6.2: The optimal social welfare matching of the small example. Every matched pair of users is linked with a direct arc from the driver to the assigned rider.

Table 6.8: The VCG payments of the users in the small example

r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	r_9	r_{10}	r_{11}	r_{12}	r_{13}	r_{14}
5.75	4.1	8.9	5.45	10.95	0.0	0.0	10.85	0.0	7.5	6.6	5.6	1.8	3.3

d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}	d_{11}	d_{12}	d_{13}
-10.4	0.0	-13.4	-8.0	-17.5	-15.5	-19.6	-6.0	-9.4	-8.5	0.0	-20.0	-9.25

Now, let us discuss the steps of our proposed subsidy scheme. In the first step, we apply Algorithm 6.2 to obtain a matching between the riders and drivers independent of their asks and bids. For the sake of this example, we consider a mapping that maximizes the total VMTS, whose optimal solution is demonstrated in Figure 6.3. This step yields the mapping σ between the riders and the drivers. This mapping enables us to reduce the ask matrix of the drivers into an ask vector A^{σ} as shown in Table 6.9. Finally, combined with the result of the pre-processing procedure, we can separate the pairs of users into group 1, shown in Figure 6.4, and group 2, shown in Figure 6.5. More specifically, drivers d_1, d_8, d_9, d_{12}, and d_{13} do not need to extend their time windows to serve riders r_2, r_{12}, r_{10}, r_{14}, and r_{13}, respectively. Thus, they are not qualified to receive subsidy from the system and belong to group 1. The rest of the matched drivers in Figure 6.3 are expected to extend their time windows to serve their assigned riders, and thus, they belong to group 2.

Table 6.9: The ask vector of the drivers A^{σ} given the maximum VMTS mapping σ in the small example

d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}	d_{11}	d_{12}	d_{13}
0.0	11.0	5.75	6.95	9.8	8.9	10.1	5.6	7.5	6.4	11.7	3.3	1.8

In the second step, we sort the riders of group 1 in descending order of their bids and the drivers of group 1 in ascending order of their asks, as shown in Figure 6.4. As a result, the sequence of riders

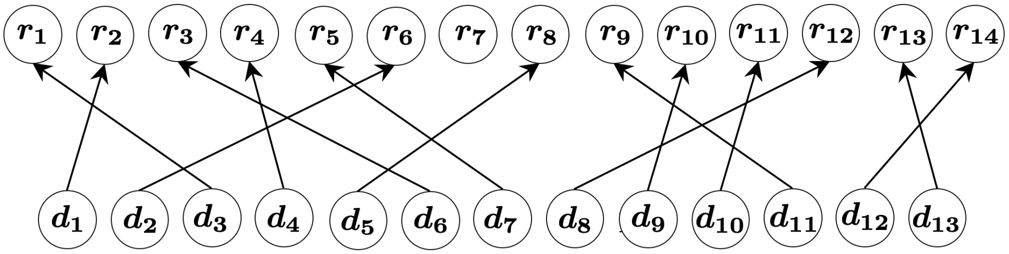


Figure 6.3: The optimal VMTS mapping of the small example. Every matched pair of users are linked with a direct arc from the driver to the assigned rider.

and drivers can be found as $\mathcal{R}_1 = \langle r_{14}, r_{10}, r_2, r_{13}, r_{12}\rangle$ $\mathcal{D}_1 = \langle d_1, d_{13}, d_{12}, d_8, d_9\rangle$. Next, we find the largest k such that the bid of k^{th} rider in $\mathcal{R}_1$ is at least as high as the ask of the k^{th} driver in $\mathcal{D}_1$. In the small example, k is 4 (7.0 > 5.6 and 6.0 < 7.5). Next, we find indices $\alpha = 5$ and $\beta = 4$ as instructed in Algorithm 6.3, i.e., $\alpha = 5$ since 6.0 > 5.6, and $\beta = 4$ since 7.5 > 7.0. As a result, we can find the candidate sets of matches $X_1^{\alpha} = \{(r_2, d_1), (r_{13}, d_{13}), (r_{14}, d_{12})\}$ and $X_1^{\beta} = \{(r_2, d_1), (r_{14}, d_{12})\}$. Since $|X_1^{\alpha}| = 3 > 2 = |X_1^{\beta}|$, we let X_1^{α} be the set of finalized matches in group 1. We also require riders r_2, r_{13}, and r_{14} to pay $p_1^{\mathcal{R}} = 6.0$ dollars to the system, and pay $p_1^{\mathcal{D}} = 5.6$ dollars to drivers d_1, d_{12}, and d_{13}.

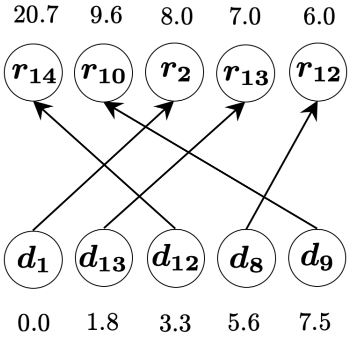


Figure 6.4: The pairs of a rider and a driver in group 1. The values on top of the riders are their bids and those under the drivers are their asks. The riders and drivers are sorted based on their bids and asks.

In the third step of our subsidy scheme, we first randomly divide the riders into two submarkets of equal size. Figure 6.5 shows an instance of the possible left and right markets denoted by L and R, respectively. More specifically, $\mathcal{R}_2^{\mathsf{L}} = \{r_4, r_5, r_8, r_9\}$ and $\mathcal{R}_2^{\mathsf{R}} = \{r_1, r_3, r_6, r_{11}\}$. Next, we find the trade prices p_2^{L} and p_2^{R}. For finding trade price p_2^{R}, we only use the ask and bids of the users in the left market. As such, we first sort the riders in the left market in descending order of their bids and store them in sequence $\mathcal{R}_2^{\mathsf{L}} = \langle r_5, r_8, r_4, r_9\rangle$. Next, we initialize the social welfare of the left market, G, to zero. Then, we assume the trade price is the bid of rider r_5. Let D_2' be the set of drivers whose assigned rider bids at least as high as $B_5 = 27.3$, which results in

$D'_2 = \{d_7\}$ and $\mathcal{D}'_2 = \langle d_7 \rangle$. Since, $10.1 - 27.3 \leq \frac{10}{2}$, we let k be 1. The social welfare of the only pair (r_5, d_7) is $27.3 - 10.1 = 17.2 \geq 0 = G$. Thus, we update G to 17.2 and $p^{\mathsf{R}} = 27.3$. Next, we assume the trade price is $B_8 = 17.5$. In this case, $D'_2 = \{d_7, d_5\}$ and $\mathcal{D}'_2 = \langle d_5, d_7 \rangle$. Since, $2 \times (10.1 - 17.5) \leq \frac{10}{2}$, we let k be 2. The social welfare of pairs (r_5, d_7) and (r_8, d_5) is $(27.3 - 10.1) + (17.5 - 9.8) = 24.9 \geq 17.2 = G$. Thus, we update G to 24.9 and $p^{\mathsf{R}} = 17.5$. Next, we assume the trade price is $B_4 = 10.4$. In this case, $D'_2 = \{d_7, d_5, d_4\}$ and $\mathcal{D}'_2 = \langle d_4, d_5, d_7 \rangle$. Since, $3 \times (10.1 - 10.4) \leq \frac{10}{2}$, we let k be 3. The social welfare of pairs (r_5, d_7), (r_8, d_5), and (r_4, d_4) is $(23.7 - 10.01) + (17.5 - 9.8) + (10.4 - 6.95) = 28.35 \geq 24.9 = G$. Thus, we update G to 28.35 and $p^{\mathsf{R}} = 10.4$. Finally, we assume the trade price is $B_9 = 8.2$. In this case, $D'_2 = \{d_7, d_5, d_4, d_{11}\}$ and $\mathcal{D}'_2 = \langle d_4, d_5, d_7, d_{11} \rangle$. Since $4 \times (11.7 - 8.2) \nleq \frac{10}{2}$, we cannot pass all the drivers. Also, $3 \times (10.1 - 8.2) \nleq \frac{10}{2}$, which indicates that we cannot let k be 3. However, $2 \times (9.8 - 8.2) \leq \frac{10}{2}$ which indicates that we can let k be 2. Nonetheless, the social welfare of pairs (r_8, d_5), and (r_4, d_4) is $(17.5 - 9.8) + (10.4 - 6.95) = 11.15 \ngeq 28.35 = G$. Thus, neither G nor p^{R}_2 will be updated in this case. We follow the same procedure for finding the trade price in the left market, p^{L}_2, using the bids and asks of the users in the right market. By doing so, we obtain p^{L}_2 as the bid of rider r_{11}, i.e., $p^{\mathsf{L}}_2 = B_{11} = 8.5$.

Having found these trade prices, we use the procedure in Algorithm 6.5 to find the subsidies and winning users in group 2 of the small example. More precisely, let D^{L}_2 be the set of the drivers in the left market whose assigned rider bids at least as high as $p^{\mathsf{L}}_2 = B_{11} = 8.5$. In the small example, D^{L}_2 can be found as $\{d_4, d_5, d_7\}$. We sort these drivers in ascending order of their asks and store it in sequence $\mathcal{D}^{\mathsf{L}}_2 = \langle d_4, d_5, d_7 \rangle$. Next, we check if we can subsidize all these users. Since $3 \times (10.1 - 8.5) < \frac{10}{2}$, we let s^{L} be $\frac{10}{2*3} = 1.67$. We also let $X^{\mathsf{L}}_2 = \{(r_5, d_7), (r_8, d_5), (r_4, d_74)\}$ as the set of finalized matches in the left market. Doing the same procedure for the right market with trade price $p^{\mathsf{R}}_2 = 10.4$ yields $s^{\mathsf{R}}_2 = 2.5$ and $X^{\mathsf{R}}_2 = \{(r_3, r_6), (r_1, d_3)\}$.

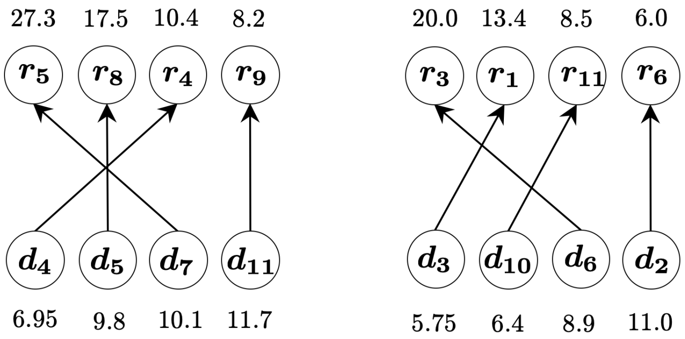
Combining the results of these steps provides us with the matching shown in Figure 6.6. The social welfare of this outcome is $\mathrm{SW}^2 = 79.95$, which is 93% of the optimal social welfare found by the VCG mechanism. Also, the payments made by the users in our subsidy scheme for the small example are presented in Table 6.10. Summing the negative of these payments gives us the required subsidy for our subsidy scheme, $\mathrm{PD}^2 = 8.8$, which is lower than the total budget of 10 dollars.

Table 6.10: The payments of the users by the subsidy scheme in the small example

r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	r_9	r_{10}	r_{11}	r_{12}	r_{13}	r_{14}
10.4	6.0	10.4	8.5	8.5	0.0	0.0	8.5	0.0	0.0	0.0	0.0	6.0	6.0

d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}	d_{11}	d_{12}	d_{13}
-5.6	0.0	-12.9	-10.17	-10.17	-12.9	-10.17	0.0	0.0	0.0	0.0	-5.6	-5.6

Before finalizing this section, let us provide the results of our subsidy scheme for the small



(a) Left market (L)

(b) Right market (R)

Figure 6.5: The pairs of a rider and a driver in group 2. The values on top of the riders are their bids and those under the drivers are their asks. The riders and drivers in each submarket are sorted based on their bids and asks.

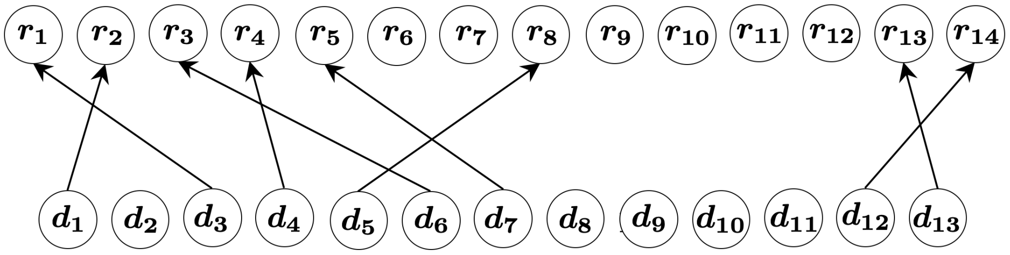


Figure 6.6: The matching of the subsidy scheme for the small example. Every matched pair of users are linked with a direct arc from the driver to the assigned rider.

example when there is no budget available for subsidizing the drivers. Setting B to zero does not change the result of first step. However, in the second step the set of finalized pairs changes as $X^{\mathsf{L}}_2 = \{(r_4, d_4)\}$, $X^{\mathsf{R}}_2 = \{(r_1, d_{14}), (r_3, d_6)\}$ although the trade prices of $p^{\mathsf{L}}_2 = 8.5$ and $p^{\mathsf{R}}_2 = 10.4$ remain the same. Also, $s^{\mathsf{L}}_2 = s^{\mathsf{R}} = 0$. The social welfare associated with this outcome is $\mathrm{SW}^3 = 55.05$ (64% of the optimal social welfare), while its payment deficit is $\mathrm{PD}^3 = -1.2$, which indicates that the required subsidy by the system is obviously zero and instead the system has a revenue of 1.2 dollars.

6.5 Numerical Experiment

In this section, we perform a set of simulation experiments using the New York City taxi dataset[1] to assess the quality of our proposed subsidy scheme. In the following subsection, we carefully define the parameter settings of our numerical experiments, and introduce a number of performance metrics to evaluate the performance of different mechanisms against each other. Afterwards, we

[1] http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml

present the result of various experiments.

All the experiments in this study are performed on a 3.50 GHz Intel Xeon machine with a 64-bit version of the Windows 10 operating system with 128.0 GB RAM. The data preparation, and the algorithms provided in 6.4 are coded in `Python 3.9`, and all the optimization problems introduced in this chapter are solved using `GUROBI 9.2`. Finally, the road network of the study region is extracted from `Open Street Map`, and the shortest-path travel time and driving distance between every pair of stations are obtained from the `Google Map API`.

6.5.1 Experiment setup and Parameter Settings

For the sake of this study, we focus only on the trips that originate and end in the Manhattan area of the New York City during the morning peak hour of [7:00,10:00]. For finding the trip information of the users in 10 different days (instances), we use the historical trip data in the New York City taxi dataset from Feb 1, 2016 to Feb 10, 2016 subject to the following assumptions and modifications:

- The origin and destination stations as well as the desired earliest departure time of every user is obtained directly from the dataset.
- In order to obtain the desired latest arrival time of every user, we generate the random parameter Υ_{n_ℓ} as a user's time flexibility, and add it to the sum of the desired earliest departure time and the shortest-path travel time between the origin and destination stations of that user. i.e., $\mathsf{J}_{n_\ell} = \mathsf{I}_{n_\ell} + \tau_{\mathsf{I}_{n_\ell},\mathsf{J}_{n_\ell}} + \Upsilon_{n_\ell}$. We further assume that this parameter follows a normal distribution with mean $\mu(\Upsilon)$ and standard deviation $\mu(\Upsilon)/5$,
- We only consider trips that are longer than 10 minutes.
- Half of the users are randomly selected to be riders and the rest are chosen to be drivers.
- A rider's value of distance and a driver's value of distance and time window extension are randomly generated following Normal distributions with means $\mu^R(\delta)$, $\mu^D(\delta)$, and $\mu^D(\vartheta)$ and standard deviations $\mu^R(\delta)/5$, $\mu^D(\delta)/5$, and $\mu^D(\vartheta)/5$, respectively.

We consider a base scenario of the ridesharing system with parameter settings listed in Table 6.11. The parameters in this table can be defined as:

- $|N|$: the total number of riders and drivers in the system,
- B : the total available budget for subsidy,
- $\mu^R(\delta)$: the mean of rider's value of distance,
- $\mu^D(\delta)$: the mean of driver's value of distance,
- $\mu(\vartheta)$: the mean of driver's value of time extension,
- $\mu(\Upsilon)$: the mean of user's time window flexibility, and finally,
- Γ : the threshold for extending a user's time window.

In our numerical experiments, we compare the performance of our subsidy scheme, denoted

Table 6.11: The parameter setting for the base scenario.

$\lvert N\rvert$	B	$\mu^R(\delta)$	$\mu^D(\delta)$	$\mu^D(\vartheta)$	$\mu(\Upsilon)$	Γ
	(\$)	(\$/mile)	(\$/mile)	(\$/min.)	(min.)	(min.)
4,000	500	2.0	0.2	2.0	5	5

by $\mathcal{M}_2$, with that of the VCG mechanism, denoted by $\mathcal{M}_1$, and the no-deficit variant of our subsidy scheme, denoted by $\mathcal{M}_3$. We introduce the following performance metrics to quantify the performance of these mechanisms:

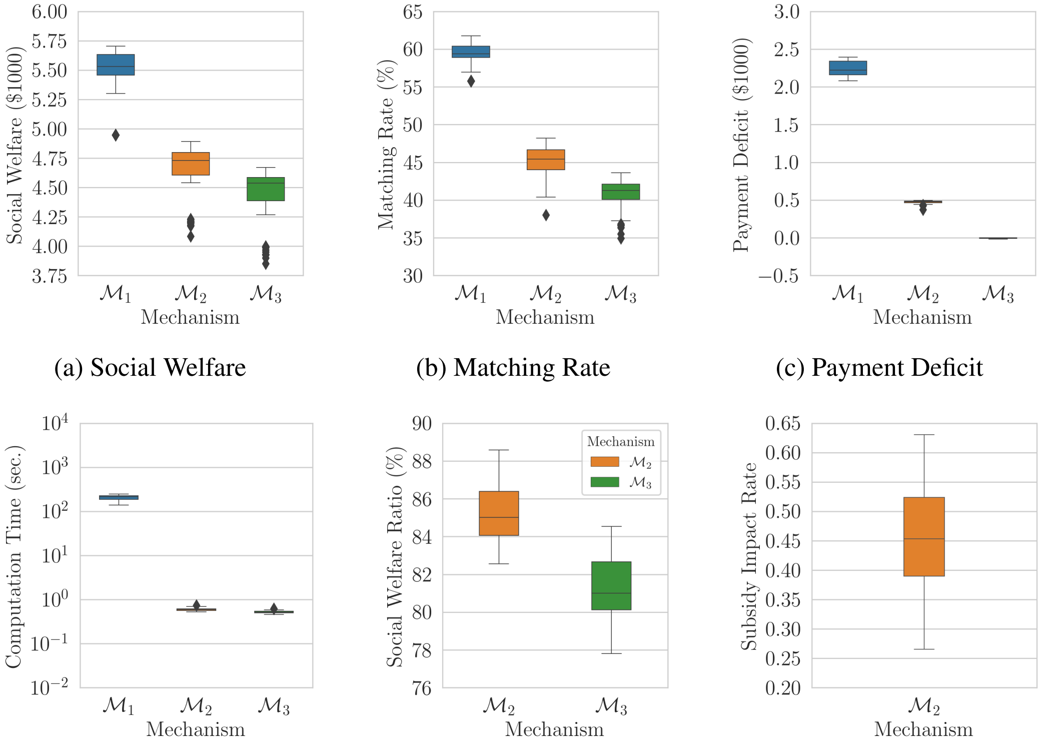
- Social Welfare of $\mathcal{M}_i$ (SW^i) = $\sum_{(r_i,d_j)\in X}(B_i - A^i_j)$, for $i = 1, 2, 3$,
- Matching Rate of $\mathcal{M}_i$ (MR^i) = $100 \times 2\,|X| \big/ |N|$, for $i = 1, 2, 3$,
- Payment Deficit of $\mathcal{M}_i$ (PD^i) = $-\sum_{n_\ell\in N} p_{n_\ell}$, for $i = 1, 2, 3$,
- Computation Time of $\mathcal{M}_i$ (CT^i) = Implementation time of mechanism $\mathcal{M}_i$, for $i = 1, 2, 3$,
- Subsidy Impact Rate of $\mathcal{M}_i$ (SIR^i) = $(\text{SW}^i - \text{SW}^3)\big/\text{PD}^i$ for $i = 2$,
- Social Welfare Ratio of $\mathcal{M}_i$ (SWR^i) = $100 \times \text{SW}^i\big/\text{SW}^1$ for $i = 2, 3$.

Finally, we assume that the ridesharing has an estimate of \$2/mile for the mean value of distance, $\hat{\delta}$, and an estimate of \$0.2 /min for the mean value of time extension, $\hat{\vartheta}$. Therefore, unless stated otherwise, the results of mechanisms $\mathcal{M}_2$ and $\mathcal{M}_3$ are based on the mapping objective function presented in (6.8).

6.5.2 The Performance of the Proposed Subsidy Scheme

In this subsection, we show the performance of our proposed subsidy scheme in comparison with an efficient mechanism and a no-deficit mechanism, when applied to a ridesharing system with the base scenario of parameter settings. Figures 6.7(a)-(f) show the box plots of different performance metrics over 10 instances of the base scenario. As expected, Figure 6.7(a) suggests that the VCG mechanism generates the highest amount of social welfare. Additionally, Figure 6.7(b) indicates that it also matches a higher number of pairs compared to the other two mechanism. However, VCG achieves these positive results at the cost of running a budget deficit as high as almost half of the generated social welfare (see Figure 6.7(c)), and a significant computational burden in orders of magnitude (see Figure 6.7(d)). Therefore, given a set budget of \$500 as used in the base scenario, the VCG mechanism seems to be inapplicable for the ridesharing system.

The proposed subsidy scheme, however, can help the system yield about 85% of the maximum possible social welfare, as shown in Figure 6.7(e), while respecting the budget constraint. Also, Figure 6.7(f) shows that every dollar spent on subsidy increases the social welfare by \$0.45 on average. Moreover, using the available budget enables our subsidy scheme to increase the matching



(a) Social Welfare

(b) Matching Rate

(c) Payment Deficit

(d) Computation Time

(e) Social Welfare Ratio

(f) Subsidy Impact Rate

Figure 6.7: The comparison between the performance of mechanisms $\mathcal{M}_1$, $\mathcal{M}_2$, and $\mathcal{M}_3$.

rate by 5% on average compared to a no-deficit mechanism. In order to see why $\mathcal{M}_2$ generates a higher social welfare and matching rate compared to $\mathcal{M}_3$, we present the box plots of the prices found by Algorithms 6.3-6.5 in both mechanisms in Figure 6.8. As expected, the prices of both riders and drivers in group 1 are the same, because subsidy has no impact on this group. However, M_2 has lower trade prices in the left and right markets of group 2 compared to M_3, which indicates

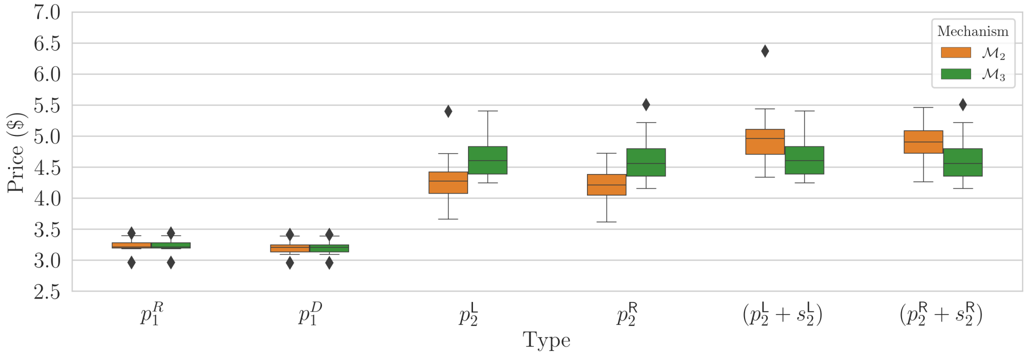


Figure 6.8: Comparison between the prices of mechanism $\mathcal{M}_2$ and $\mathcal{M}_3$.

that M_2 charges the riders in group 2 less, which ultimately leads to a higher number of profitable matches being finalized that would not have been finalized in $\mathcal{M}_3$, and subsequently a higher social welfare. Furthermore, this figure also suggests that the sum of trade prices and subsidies are higher in M_2, which points out larger payments to the drivers. Therefore, although the system directly targets and compensates drivers for time window extension, both riders and drivers benefit from the subsidy scheme in the form of lower and higher prices, respectively.

6.5.3 The Impact of the Mapping Objective and Random Sampling

In Section 6.4.1.1, we introduced three objective functions to find an initial matching between the riders and drivers in the first step of our proposed subsidy scheme. Let us call these objectives MEAN (i.e., maximizing social welfare based on an estimate of private parameter means), VMTS (i.e., maximizing vehicle miles traveled savings), and MM (i.e., maximum matching). In Figure 6.9, we demonstrate social welfare ratios of $\mathcal{M}_2$ and $\mathcal{M}_3$ under these three objectives over 10 instances of the base scenario. This figure shows that the MEAN method obviously generates the highest social welfare. In the case of having no estimates of mean values, VMTS enables us to obtain 80% of social welfare. This loss in social welfare is due to the fact that VMTS does not exploit the information regarding the sensitivity of users to time window extension. As such, it may initially match pairs in which drivers incur a large cost of time window extension. Consequently, we will not be able to finalize such pairs in the third step, and thus, lose a part of social welfare. Also, MM generates the lowest amount of social welfare among these objectives. This is not surprising because on top of the issue mentioned for VMTS, MM also completely ignores the gain of a match, which may result in finalizing the matches whose welfare is negative.

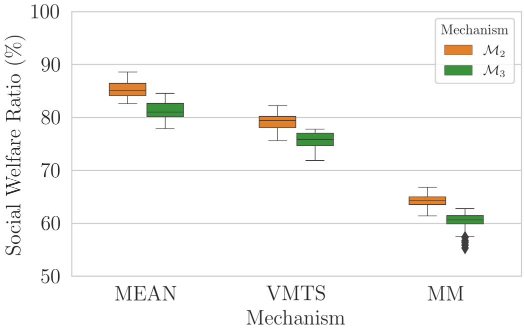
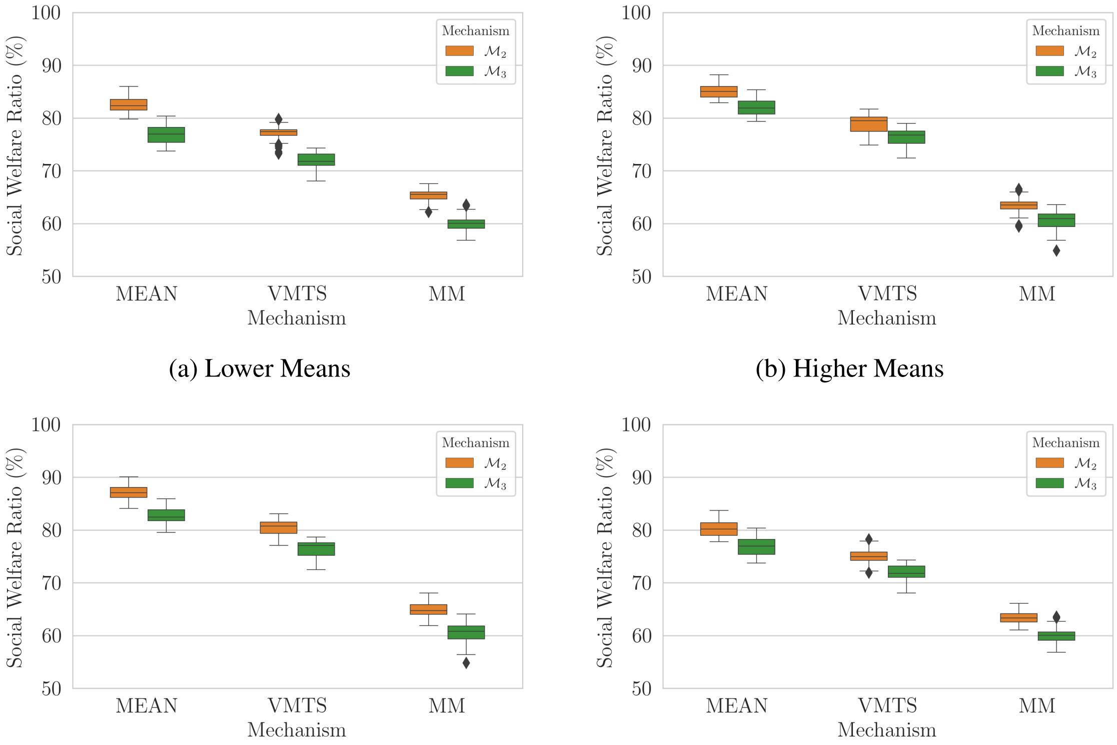


Figure 6.9: The social welfare ratio of mechanisms $\mathcal{M}_2$ and $\mathcal{M}_3$ with different mapping objective functions for the base scenario.

Note that in the base scenario, the mean values of distance and time window extension of the users match the estimates of the ridesharing system. In order to have a fair comparison between these objectives and also investigate the robustness of the MEAN objective with respect to the precision of the estimates, we consider four additional scenarios by changing the mean or standard deviation of the users' values. More precisely, in the "Lower Means" scenario, we decrease the means of δ and ϑ to \$1/mile and \$0.1/min., respectively, and in the "Higher Means" scenario, we increase them to \$3/mile and \$0.3/min. Also, in the "Lower Variances" scenario, we decrease the standard deviations of δ and ϑ to \$2/mile and \$0.02/min., respectively, and in the "Higher Variances" scenario, we increase them to \$8/mile and \$0.8/min. Figures 6.10(a)-(d) show the results of social welfare ratio for these additional scenarios. These figures suggest that the performance of the MEAN method slightly degrades in all scenarios, especially in the Higher Variance scenario. However, the magnitude of decrease in social welfare ratio is less than 5% in the worst case. Interestingly, the performance of VMTS slightly increases in the Low Variances scenario and decreases in the High Variance scenario. Also, MM does not show any significant sensitivity to these values. Despite all changes, the performance of MEAN is better than that of VMTS and MM under all variations.



(a) Lower Means

(b) Higher Means

(c) Lower Variances

(d) Higher Variances

Figure 6.10: The social welfare ratio of mechanisms $\mathcal{M}_2$ and $\mathcal{M}_3$ with different mapping objective functions under different scenarios of population values.

Finally, in Figure 6.11, we show the social welfare ratio of $\mathcal{M}_2$ and $\mathcal{M}_3$ for different random seeds used to create the right and left markets for pairs in group 2, in each instance of the base scenario. This figure suggests that the social welfare of our subsidy scheme and its no-deficit variant might change slightly as a result of the random sampling method in step 3 of our proposed method. However, the magnitude of the change does not go beyond 3% in the worst case. Also, the performance of the mechanism with budget is always better than that of the mechanism without it.

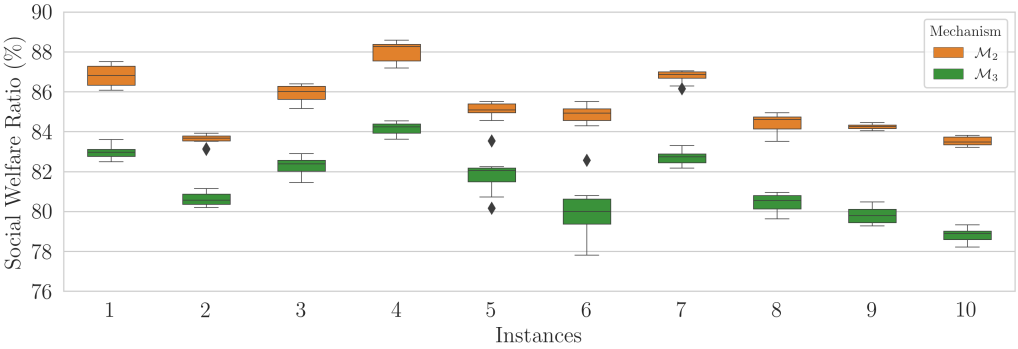


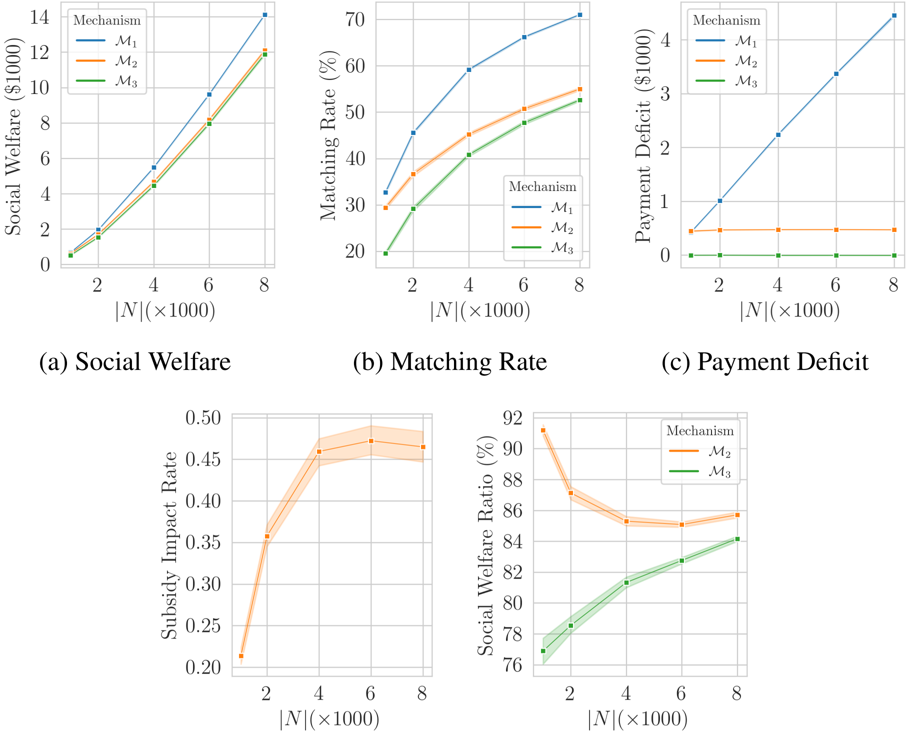
Figure 6.11: The impact of the random sampling on the social welfare ratio of mechanism $\mathcal{M}_2$ and $\mathcal{M}_3$ across different instances of the base scenario.

6.5.4 Sensitivity Analysis

In order to analyze the impact of parameter values on the performance metrics, in this subsection we consider the result of different scenarios of parameter setting. These scenarios are built by changing one parameter of the base scenario at a time, and their results for mechanisms $\mathcal{M}_1$, $\mathcal{M}_2$, and $\mathcal{M}_3$ are plotted in Figures 6.12, 6.13, E.1, E.2, E.3, E.4, and E.5. Every plot shows the average of the corresponding performance metric over 10 instances, and the shaded region demonstrates the 95% confidence intervals of the average values. In what follows, we present the analysis of Figures 6.12 and 6.13. The rest of Figures and their interpretations are presented in Appendix E.3.

In the base scenario, we assume that 4000 users (2000 drivers and 2000 riders) participate in the ridesharing system every day. To analyze the impact of this parameter, we change it to 1000, 2000, 6000, and 8000. The results are presented in Figure 6.12. Figures 6.12(a) and 6.12(b) indicate that the social welfare of all the mechanisms increases super-linearly, and their matching rates increase sub-linearly as the number of users increases. This is due to the fact that the system is not saturated and adding more users increases the possibility of matching, and thereby, finding more profitable matches. Also, Figure 6.12(c) suggests that the payment deficit of VCG increases linearly with the number of users. Figure 6.12(d) indicates that given a set budget of $500, the impact of every dollar spent on subsidy increases sub-linearly with the number of users until it converges to a maximum value. This is due to the fact the matching rate increases by adding more users, and thus, a given budget allows the subsidy scheme to add more to the social welfare. Despite this impact, Figure

6.12(e) shows that the difference between the social welfare ratio of $\mathcal{M}_2$ and $\mathcal{M}_3$ decreases as $|N|$ increases. This is because with a higher $|N$, the necessity for extending time windows decreases, and thus, the drivers' asks can be satisfied by the bid of riders without the help of subsidy. These figures suggest that as the penetration rate of the system goes up, we can rely on the users to generate a high social welfare (85% of the maximum possible) even without the help of any external capital budget.



(a) Social Welfare

(b) Matching Rate

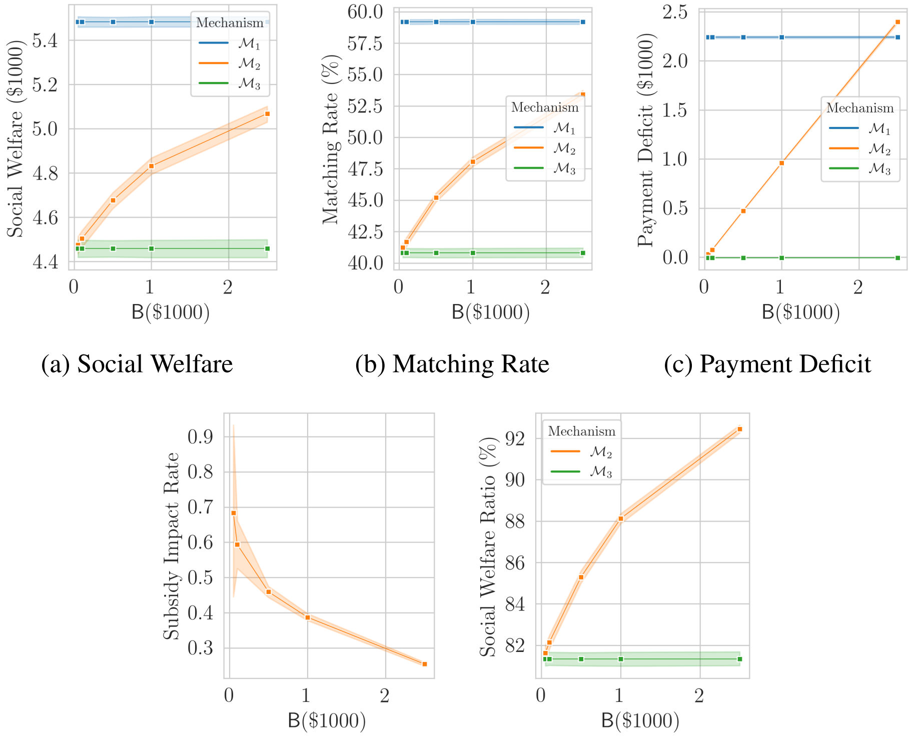
(c) Payment Deficit

(d) Subsidy Impact Rate

(e) Social Welfare Ratio

Figure 6.12: Impact of number of participants

Figure 6.13 demonstrates the results of changing the available budget for subsidy from $500 in the base scenario to $100 $250, $1,000 and $2,500. Note that only the results of $\mathcal{M}_2$ changes in this case, as the two other benchmarks do not utilize a budget. Figures 6.13(a) and 6.13(b) show that the social welfare and the matching rate of the subsidy scheme increase sub-linearly with the value of B. Also, the payment deficit increases linearly from about $100 to about $2500, suggesting that almost all budget will be used in each scenario to increase the social welfare. As Figure 6.12(e) suggests, this increase in budget can increase social welfare ratio up to 92% of the maximum social welfare. However, Figure 6.12(d) indicates that the impact of every dollar spent on subsidy decreases with the available budget. This is due to the fact that by adding more budget, we also increase the number of so-called "unprofitable" matches in which the ask of drivers is greater than the bid of riders. Therefore, although we increase the matching rate and even social welfare, the ratio of finalized unprofitable matches increases such that they adversely influence the subsidy impact rate.



(a) Social Welfare

(b) Matching Rate

(c) Payment Deficit

(d) Subsidy Impact Rate

(e) Social Welfare Ratio

Figure 6.13: Impact of available capital budget

6.6 Conclusion and Future Work

In this chapter, we consider a static one-to-one ridesharing system that uses a limited amount of budget to motivate drivers to extend their time windows, and thus, increase the possibility of matching with riders. With the assumption of having strategic users that hold their values of distance and time window extension as private information, we show that the ride-matching and pricing in this market can be modeled as a combinatorial double auction with heterogeneous items. Applying the celebrated VCG mechanism usually requires a significant and unbounded amount of budget that makes it inappropriate for finding a feasible outcome with limited budget deficit. As such, we propose a multi-step subsidy scheme that satisfies truthfulness, voluntary participation, budget-balance, and computational efficiency. Using the New York City taxi dataset, we evaluate different aspects of our proposed subsidy scheme in comparison with the VCG mechanism and a no-deficit variant of the scheme. From the results of these experiments, we infer that:

1. For large-scale systems, the proposed subsidy scheme can yield about 85% of the maximum possible social welfare on average, while ensuring that payment deficit does not violate the budget constraint.
2. The choice of the objective function for finding the initial mapping between riders and drivers significantly impacts the performance of the proposed subsidy scheme. In the case of having an estimate of users' private values, we suggest maximizing the expected gain, and in the absence of any estimate, we suggest maximizing the vehicle miles travelled saving.
3. The random sampling to find left and right markets in the mechanism does not significantly change the social welfare of our subsidy scheme.
4. The proposed subsidy scheme generally results in higher social welfare, when compared to its no-deficit variant. However, it is not advised to subsidize systems with high penetration rate or very flexible users.

In this study, we assume that the ridesharing system is a benevolent market maker that possesses an available capital budget to inject to the system. One possible direction to extend the work of this study is to consider the ridesharing system as a player with a utility function. In this case, the ridesharing system will not give out the monetary incentive unless doing so increases the utility of the system.

CHAPTER 7

Conclusion and Discussions

7.1 Research Summary and Findings

The advent of advanced communication technologies and high-performance computing have enabled various shared mobility systems, including P2P ridesharing and ride-pooling services, which can turn the rise in population growth into opportunities–in densely-populated areas, high demand for travel makes shared services profitable, giving rise to business models that not only can meet the growing travel demand, but can do so while reducing the total vehicles-miles-travelled (VMT) in the transportation network. In spite of several individual and societal benefits that ridesharing services offer, there are various operational and/or economic challenges that highly impact their performance and impede them from becoming a viable and reliable mode of transportation in the long run. This dissertation attempts to address some of these challenges by proposing high quality solutions that are developed based on well-known techniques from the fields of operations research, data mining, and game theory. The methodologies presented in this dissertation can be roughly divided into two major parts.

7.1.1 Fast And Reliable Dynamic Operations

The first part of the dissertation looks at ridesharing services from the prospective of operators, and devises new techniques to efficiently serve customers while taking system-wide considerations into account. For this purpose, Chapters 2 and 3 introduce two major challenges that P2P ridesharing and ride-pooling systems face in real time, respectively. The high computational complexity of the underlying ride-matching problems immensely undermines the real-time application of ridesharing services. As such, Chapter 2 introduces a distributed optimization technique based on graph partitioning to facilitate the implementation of dynamic ridesharing systems in densely populated metropolitan areas. The result of numerical experiments shows that this technique can effectively reduce the response time of the platform in solving various forms of ride-matching problems while providing high-quality solutions.

In Chapter 3, we combine this partitioning technique with predictions from historical demand data and an innovative local search algorithm to design a framework that dispatches the fleet of vehicles proactively to serve on-demand requests. This framework not only helps the system substantially reduce the search space of the feasible matches, and subsequently, decrease the response time, but also enables the system to react properly to unexpected demand. Numerical experiments reveal that the proposed framework results in cutting down the passenger waiting time and improving their LoS, while at the same time, increasing the utilization rate of vehicles, when compared to myopic state-of-the art methods. These results imply that using historical data allows the system to improve its ride-matching policies to obtain higher quality solutions with respect to all (i.e., both the realized and forthcoming) trips.

7.1.2 Subsidy-Enabled Economic Benefits

The second part of this dissertation aims to couple the expectation of operators with those of customers. However, in most cases, these expectations do not coincide. As subsidy plays an important role in the sustainability of any major public or private transport service, Chapters 4, 5, and 6 introduce various applications of subsidy to close the gap between these expectations in different forms of ridesharing services. Chapter 4 introduces an incentive program that uses monetary subsidy to change travel behavior of daily commuters by extending their time windows in the morning and/or evening and compensating them accordingly. The result of experiments indicates that this strategy increases the system social welfare substantially. More importantly, the distribution of the allocated subsidy in these experiments suggests that this program help the system promote transportation equity around the study region.

In Chapter 5, we introduce another implication of subsidy in ridesharing services. More specifically, we show that a ridesharing service with role-flexibility can be modeled as a roommate game that may not assume a stable outcome. Hence, there may be no combination of matching, role assignment, and pricing that includes no pair of users that would rather leave their current partners in order to share their rides together. In fact, we show that the optimal solution of the primal problem, a maximum weighted matching problem, represents the expectation of an operator, while the optimal solution to the dual problem attributes to that of users. In order to close the gap between the two problems, we introduce a mixed integer problem that finds the minimum amount of subsidy added to the weight of the blocking pairs such that the system becomes stable. The result of numerical experiments suggests that the required subsidy to stabilize the ridesharing market is infinitesimally small. Also, the proposed treatment is superior to excluding blocking pairs from the system, both economically and computationally.

Finally, Chapter 6 considers a ridesharing market with incomplete information and limited budget, in which drivers are offered to extend their time windows in exchange for monetary incentives. For this market, we show that the well-known VCG mechanism is inadequate in that its budget deficit may violate the available budget. In fact, our numerical experiments show that the payment deficit can be as large as half of the generated social welfare. In order to tackle this issue, we introduce a truthful mechanism with bounded payment deficit, referred to as a subsidy scheme. Although this mechanism does not guarantee a bounded efficiency theoretically, the experiments show that this scheme can secure 85% of the maximum social welfare on average. Also, the subsidy scheme enables the system to generate a higher social welfare with the help of subsidy, when compared to its no-deficit variant. However, the impact of subsidy in this chapter is much lower than that of the previous two chapters. This is mainly due to the fact that the introduction of private information and the necessity for satisfying truthfulness in this chapter requires the system to treat all drivers in need of subsidy equally regardless of their incurred disutility, which results in wasting a significant portion of capital budget.

7.2 Directions for Future Work

At the end of Chapters 2-6, we present the potential directions to further extend the methodologies presented therein. In what follows, we provide additional topics related to the content of this dissertation that are worth exploring in further detail.

7.2.1 Optimal Policy for the Rolling-Horizon Framework

In Chapters 2 and 4, we adopt a rolling-horizon approach to incorporate static algorithms into dynamic systems. With a larger rolling horizon, the system can collect a larger pool of participants, hence producing a higher matching rate. On the other hand, a longer horizon also implies higher waiting times for users (both riders and drivers) to learn about the status of their requests, thereby a lower LoS. In highly dynamic systems where the request arrival times are very close to the latest desired departure times of users, longer horizons could even lead to lost demand. As such, there is an obvious dependency between the operational efficiency of the system and the LoS of users, which is partly affected by the length of the optimization horizon. This inter-dependency calls for incorporating behavioral models into the routing, scheduling, and pricing optimization problems, and capturing the interaction between the supply and demand curves.

7.2.2 Truthful And Collusion-Free Mechanisms

In Chapters 5 and 6, we consider the ridesharing market as a cooperative and non-cooperative game with incomplete information, respectively. However, it is yet to be explored how one can design a ridesharing market as a cooperative game with incomplete information. In this case, it is crucial to design a mechanism that ensures both truthfulness and matching stability. In the literature of game theory and mechanism design, Milgrom and Segal (2014) introduces the differed acceptance (DA) auction that satisfies both properties. Also, Dütting et al. (2017) devise approximate greedy algorithms to apply the DA auction method in the context of matching and knapsack problems. However, their proposed algorithms cannot be directly applied to ridesharing markets in which the drivers (and sometimes also riders) are not single-minded agents, i.e., they have heterogeneous preferences (valuations) toward different matches.

7.2.3 Integrating Ridesharing into Mobility-as-a-Service (Maas)

With mobility-as-a-service (Maas) becoming a more popular mode of transportation, an interesting research direction for extending the work in this dissertation is to investigate integrating ridesharing services into MaaS operations, and study the implications of such integration on the MaaS LoS, as well as the empty miles imposed on the transportation infrastructure (Qi and Shen, 2019). More specifically, the following two research questions are of interest regarding the operations of MaaS services: (1) the implications of different P2P matching settings (i.e., one-to-one, one-to-many, many-to-one, and many-to-many) on the empty miles driven of MaaS, and (2) regulations and policies specific to each matching setting to curb the MaaS empty miles driven. Moreover, the integration of different forms of shared mobility services with possibly conflicting interests gives rise to intriguing questions regarding the pricing mechanisms adopted by MaaS services: (1) the cooperation between competing transport agencies and its impact on the pricing, (2) the distribution of available subsidy among involved transport providers.

7.2.4 Electrification and Autonomy

Experts believe that autonomous vehicles will largely be electrified for a number of technical reasons that revolve around the structural compatibility of autonomous driving with electric propulsion systems (Ersal et al., 2020). It is also believed that autonomous vehicles will be deployed by shared mobility providers due to their larger price tags, their ability to reposition themselves, and environmental concerns (Masoud and Jayakrishnan, 2017a, 2016). As such, ridesharing services provide a suitable platform to host electric autonomous vehicles. However, the integration of autonomy and electrification into ridesharing systems gives rise to a number of technical, operational, and economic challenges in these systems. Specifically, matching, routing, scheduling, and pricing in these systems should take into account the level of charge of vehicles and the distribution of charging stations in the network, adding to the complexity of these problems (MirHassani and Ebrazi, 2013; Schneider et al., 2014; Desaulniers et al., 2016; Sweda et al., 2017; Fazeli et al., 2020a,b).

The prospect of a future electrified transportation system has resulted in a number of alternative charging strategies that enable vehicles to be charged while driving. One such strategy is providing dedicated charging lanes that would enable electric vehicles to charge their batteries from the electric grid while driving (Li and Mi, 2014; Parvez et al., 2014; Stüdli et al., 2014; Alonso et al., 2014; Zhang et al., 2021). The possibility of charging a vehicle while traveling on a selected set of links changes the structure of the ride-matching problems, and calls for customized routing algorithms. Vehicle-to-vehicle power transfer is another recently proposed technology that enables an electric car to charge its battery en-route by connecting to another electric vehicle (Sanchez-Martin et al., 2012; Wang et al., 2016a; Abdolmaleki et al., 2019). The ride-matching optimization problems that arise in this context grow even more complex than the case with dedicated charging lanes. This is due to the fact that the ride-matching problems have to model scenarios in which not only can a vehicle receive electric power while traveling on a link, but the supply of electricity needs to be provided by other electric vehicles that need to be routed so as to share (part of) their paths with the recipient vehicles. This charging strategy would necessitate developing not only new algorithms, but also pricing mechanisms based on utility functions of the suppliers and recipients of power in this exchange. Such a charging platform provides the opportunity for a shared mobility provider to use its own fleet as power suppliers, giving rise to more interesting, but complex problems.

APPENDIX A

Supporting Materials of Chapter 2

A.1 Some Useful Propositions

Proposition A.1. *The coefficient matrix in model (2.8) is totally unimodular.*

Proof. The coefficient matrix in model (2.8) can be presented as the following matrix.

$$
\begin{array}{r|ccccccccccccccc|}
 & q_{1,n_1} & q_{2,n_1} & \dots & q_{|R|,n_1} & q_{|R|+1,n_1} & \dots & q_{|R|+|D|,n_1} & \dots & q_{1,n_K} & q_{2K} & \dots & q_{|R|,n_K} & q_{|R|+1,n_K} & \dots & q_{|R|+|D|,n_K} \\
 & 1 & 0 & \cdots & 0 & 0 & \cdots & 0 & \cdots & 1 & 0 & \cdots & 0 & 0 & \cdots & 0 \\
 & 0 & 1 & \cdots & 0 & 0 & \cdots & 0 & \cdots & 0 & 1 & \cdots & 0 & 0 & \cdots & 0 \\
 & \vdots & \vdots & \ddots & \vdots & \vdots & \cdots & \vdots & \cdots & \vdots & \vdots & \ddots & \vdots & \vdots & \cdots & \vdots \\
\text{Cons. (2.8b)} & 0 & 0 & \cdots & 1 & 0 & \cdots & 0 & \cdots & 0 & 0 & \cdots & 1 & 0 & \cdots & 0 \\
 & 0 & 0 & \cdots & 0 & 1 & \cdots & 0 & \cdots & 0 & 0 & \cdots & 0 & 1 & \cdots & 0 \\
 & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
 & 0 & 0 & \cdots & 0 & 0 & \cdots & 1 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 1 \\
\hline
 & 1 & 1 & \cdots & 1 & 0 & \cdots & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\
 & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\
\text{Cons. (2.8c)} & \vdots & \vdots & \cdots & \vdots & \vdots & \cdots & \vdots & \cdots & \vdots & \vdots & \cdots & \vdots & \vdots & \cdots & \vdots \\
 & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & \cdots & 1 & 1 & \cdots & 1 & 0 & \cdots & 0 \\
\hline
 & 0 & 0 & \cdots & 0 & 1 & \cdots & 1 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\
 & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\
\text{Cons. (2.8d)} & \vdots & \vdots & \cdots & \vdots & \vdots & \cdots & \vdots & \cdots & \vdots & \vdots & \cdots & \vdots & \vdots & \cdots & \vdots \\
 & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & \cdots & 0 & 0 & \cdots & 0 & 1 & \cdots & 1
\end{array}
$$

Since total unimodularity is invariant to multiplying a row by a constant, we can multiply the sub-matrices associated with constraints (2.8c) and (2.8d) by -1. As a result, the coefficient matrix becomes equivalent to a node-edge incidence matrix of a graph, where every column has exactly a single 1 and a single -1, and all other entries are zeros. As all node-edge incidence matrices are totally unimodular, the result follows. □

Proposition A.2. *The problem in model (2.8) can be transformed to an unbalanced Hitchcock Transportation problem with* $N + 1$ *source nodes and* $2 \times K$ *sink nodes.*

Proof. Let us define 2 disjoint sets of $\mathcal{U} = \{1, \ldots, N+1\}$ and $\mathcal{V} = \{1, \ldots, 2K\}$ to represent the set of sources and sinks, respectively. Without loss of generality, we assume that the first $|R|$ elements in $\mathcal{U}$ are rider trips, each with a supply of 1, followed by $|D|$ elements of driver trips, each with a supply of 1. The last element in $\mathcal{U}$ denotes a dummy node with a supply of $N - K(\lceil(1+\varepsilon)\frac{|R|}{K}\rceil + \lceil(1+\varepsilon)\frac{|D|}{K}\rceil)$. We further assume that the first K elements in $\mathcal{V}$ represent the riders in clusters 1 to K, each with a demand of $\lceil(1+\varepsilon)\frac{|R|}{K}\rceil$, followed by another K elements to represent the drivers in cluster 1 to K, each with a demand of $\lceil(1+\varepsilon)\frac{|D|}{K}\rceil$. The supply and demand vectors are presented by $\mathcal{S}$ and $\mathcal{D}$, respectively. The arc set can be defined by the set $\mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2 \cup \mathcal{A}_3$, where $\mathcal{A}_1 = \{1, \ldots, |R|\} \times \{1, \ldots, K\}$, $\mathcal{A}_2 = \{|R|+1, \ldots, N\} \times \{K+1, \ldots, 2K\}$, and $\mathcal{A}_3 = \{N+1\} \times \{1, \ldots, 2K\}$. The cost of each arc, denoted by C_{ij}, can be determined from the dissimilarity matrix C. More specifically, C_{ij} for an arc in set $\mathcal{A}_1$ is equal to c_{ik} where i is a rider trip and k is the representative trip of cluster j. Similarly, C_{ij} for an arc in set $\mathcal{A}_2$ is equal to c_{ik} where i is a driver trip and k is the representative trip of cluster $j - K$. Finally, the arcs in set $\mathcal{A}_3$ have a zero cost. As a result, the problem can be formulated as in model (A.1).

$$\min \sum_{(i,j)\in\mathcal{A}} C_{ij} f_{ij} \tag{A.1a}$$

$$\text{s.t.} \sum_{\substack{j\in\mathcal{V}:\\(i,j)\in\mathcal{A}}} f_{ij} = \mathcal{S}_i,, \qquad \forall\, i \in \mathcal{U}, \tag{A.1b}$$

$$\sum_{\substack{i\in\mathcal{U}:\\(i,j)\in\mathcal{A}}} f_{ij} = \mathcal{D}_j, \qquad \forall\, j \in \mathcal{V} \tag{A.1c}$$

$$f_{ij} \geq 0. \tag{A.1d}$$

In this formulation, the decision variable is the flow from source i to sink j denoted by f_{ij}. The model in (A.1) is the mathematical formulation of a Hitchcock Transportation problem with unbalanced number of source and sink nodes (i.e. $2k \ll N + 1$), and the result follows. □

A.2 Formulation of One-to-one Ride-matching Problem with Role Flexibility

The decision variable $u_{pp'}$ is a binary variable that holds value 1 if user p as a rider is matched with user p' as driver, and value 0 otherwise. The objective function in Statement (A.2a) maximizes the total VMTS. Constraint (A.2b) ensures that each participant is matched with at most one other participant.

$$\max \quad \sum_{p \in P} \sum_{\substack{p' \in P: \\ (p,p') \in \Lambda}} w_{pp'} u_{pp'} \tag{A.2a}$$

$$\text{s.t.} \quad \sum_{\substack{p' \in P: \\ (p',p) \in L}} u_{p'p} + \sum_{\substack{p' \in P: \\ (p,p') \in \Lambda}} u_{pp'} \leq 1\,, \qquad \forall\, p \in P\,, \tag{A.2b}$$

$$u_{pp'} \in \{0, 1\}\,. \tag{A.2c}$$

A.3 Formulation of One-to-many Ride-matching Problem

In a ridesharing system that is discretized in time and space, a trip can be represented by a sequence of spatio-temporal links, denoted as $\pi = (t_i, s_i, t_j, s_j) \in T \times S \times T \times S$, where where T is the set of time intervals during the study time horizon. Let us denote the set of links by Π. When operating across a large transportation network over a long period of time, it can be easily shown that solving the original formulation requires a significant amount of time and computational power. To mitigate this issue, Masoud and Jayakrishnan (2017b) proposed a fast algorithm, known as the Ellipsoid Spatio-Temporal Accessibility Method (ESTAM), to eliminate the infeasible spatio-temporal links for every trip. This algorithm serves as a pre-processing step before formulating the optimization problem, analogous to forming the connectivity matrix in the bipartite matching case. ESTAM can significantly reduce the size of the feasible region without compromising the optimality of the solution found by the optimization problem.

As an output, ESTAM limits the link set to links that are accessible by participant p, denoted by Π_p. We can consequently obtain $\Pi_{rd} = \Pi_r \cap \Pi_d$, which represents the set of links on which driver d can provide a ride to rider r. This algorithm further provides the set of time steps in which participant p can accomplish their trip, denoted as T_p. After implementing this pre-processing step, the mathematical formulation in (A.3) can be used to model a one-to-many ride-matching problem. This model contains four sets of binary decision variables: (1) x_d^π, which takes value 1 if driver d travels on link π, and value 0 otherwise; (2) y_{rd}^π, which takes value 1 if rider r is served by driver d on link π, (3) u_{rd}, which takes value 1 if rider r is served by driver d, and value 0 otherwise; and (4) z_r, which takes value 1 if rider r is matched, and value 0 otherwise.

Similar to the ono-to-one ride-matching problem, the objective function in (A.3a) maximizes the vehicle miles travelled savings. Constraints (A.3b) and (A.3c) make sure that drivers start their itineraries from their origin stations, and end their itineraries at their destination stations, respectively. The flow conservation of drivers is modeled by Constraint (A.3d). Using this set of constraints, we ensure that each driver entering station s in time t, leaves that station in the same time. However, a driver can stay in the same station s at time $t + 1$. Constraints (A.3e) and (A.3f) ensure that if matched with driver d, rider r starts their itinerary from their origin station, and end their itinerary at their destination station, respectively. The flow conservation of riders is modeled by Constraint (A.3g). Constraint (A.3i) serves two purposes: first, rider r is served by driver d if they both travel on the same link, and second, the number of on-board riders served by driver d can not exceed the vehicle capacity, denoted by ξ_d, on each link. Note that since we are limiting the set of links to those that satisfy the time windows, we need not enforce any constraint regarding travel time windows.

$$
\begin{aligned}
\max \quad & \sum_{r\in R} \phi(s_r^{\mathsf{O}}, s_r^{\mathsf{D}}) z_r + \sum_{d\in D} \phi(s_d^{\mathsf{O}}, s_d^{\mathsf{D}}) - \sum_{d\in D}\sum_{\pi\in\Pi_d} \phi(s_i, s_j) x_d^{\pi} && && \text{(A.3a)} \\
\text{s.t.} \quad & \sum_{\substack{\pi\in\Pi_d: \\ s_i = s_d^{\mathsf{O}}; t_i, t_j \in T_d}} x_d^{\pi} - \sum_{\substack{\pi\in\Pi_d: \\ s_j = s_d^{\mathsf{O}}; t_i, t_j \in T_d}} x_d^{\pi} = 1\,, && \forall\, d \in D \ \ , && \text{(A.3b)} \\
& \sum_{\substack{\pi\in\Pi_d: \\ s_j = s_d^{\mathsf{D}}; t_i, t_j \in T_d}} x_d^{\pi} - \sum_{\substack{\pi\in\Pi_d: \\ s_i = s_d^{\mathsf{D}}; t_i, t_j \in T_d}} x_d^{\pi} = 1\,, && \forall\, d \in D\,, && \text{(A.3c)} \\
& \sum_{\substack{t_i, s_i \\ \pi = (t_i, s_i, t, s) \in \Pi_d}} x_d^{\pi} - \sum_{\substack{t_j, s_j \\ \pi = (t, s, t_j, s_j) \in \Pi_d}} x_d^{\pi} = 0\,, && \begin{array}{l} \forall\, d \in D\,, \forall\, t \in T_d\,, \\ \forall\, s \in S \setminus \{s_d^{\mathsf{O}} \cup s_d^{\mathsf{D}}\}\,, \end{array} && \text{(A.3d)} \\
& \sum_{\substack{\pi\in\Pi_{rd}: \\ s_i = s_r^{\mathsf{O}}; t_i, t_j \in T_r}} y_{rd}^{\pi} - \sum_{\substack{\pi\in\Pi_{rd}: \\ s_j = s_r^{\mathsf{O}}; t_i, t_j \in T_r}} y_{rd}^{\pi} = u_{rd}\,, && \forall\, r \in R\,, \forall\, d \in D\,, && \text{(A.3e)} \\
& \sum_{\substack{\pi\in\Pi_{rd}: \\ s_j = s_r^{\mathsf{D}}; t_i, t_j \in T_r}} y_{rd}^{\pi} - \sum_{\substack{\pi\in\Pi_{rd}: \\ s_i = s_r^{\mathsf{D}}; t_i, t_j \in T_r}} y_{rd}^{\pi} = u_{rd}\,, && \forall\, r \in R\,, \forall\, d \in D\,, && \text{(A.3f)} \\
& \sum_{d\in D} \sum_{\substack{t_i, s_i \\ \pi = (t_i, s_i, t, s) \in \Pi_{rd}}} y_{rd}^{\pi} - \sum_{d\in D} \sum_{\substack{t_j, s_j \\ \pi = (t, s, t_j, s_j) \in \Pi_{rd}}} y_{rd}^{\pi} = 0\,, && \begin{array}{l} \forall\, r \in R\,, \forall\, t \in T_r\,, \\ \forall\, s \in S \setminus \{s_r^{\mathsf{O}} \cup s_r^{\mathsf{D}}\}\,, \end{array} && \text{(A.3g)} \\
& \sum_{d\in D} u_{rd} = z_r\,, && \forall\, r \in R\,, && \text{(A.3h)} \\
& \sum_{r\in R} y_{rd}^{\pi} \le \xi_d x_d^{\pi}\,, && \forall\, d \in D\,, \forall\, \pi \in \Pi_{rd}\,. && \text{(A.3i)}
\end{aligned}
$$

APPENDIX B

Supporting Materials of Chapter 3

B.1 Notations for the proposed framework

1

Table B.1: List of notations.

Notation	Description
Parameters	
S	Number of stations
T	Number of time intervals
K	Number of shuttles
C	Capacity of a shuttle
ω	Maximum waiting time of passengers in their origin stations
Ω	Maximum allowed detour from the SP travel time of passengers
M	Number of alternative routes for shuttle k from historical data
ϵ	Spatio-temporal threshold for building a sub-graph around base route
β	Influence factor $\in (0, 1)$ for the cost of expected demand in the cost-benefit analysis
Π	Number of replications for the local search
R	Number of re-clustering steps for the clustering problems
ζ	Improvement factor $\in (0, 1)$ for stopping the local search
Sets	
$\mathcal{S}$	Set of stations in the road network
$\mathcal{E}$	Set of transportation links in the road network
$\mathcal{T}$	Ordered set of time intervals
$\mathcal{N}$	Set of nodes in the time-expanded network
$\mathcal{H}$	Set of passengers
$\mathcal{K}$	Set of shuttles
$\delta^+(n)$	Set of successor nodes of node n in time-expanded network
$\delta^-(n)$	Set of predecessor nodes of node n in time-expanded network

Continued on next page

Table* B.*1* – *continued from previous page

Notation	Description
$\mathcal{L}$	Set of edges in the time-expanded network
$\mathscr{N}$	Set of nodes in the min-cost flow network
$\mathscr{L}$	Set of edges in the min-cost flow network
Functions	
G	Connected, directed graph of the road network
$\mathcal{G}$	DAG of the time-expanded network
$\mathscr{D}(n)$	Node demand in the min-cost network
$\mathscr{W}(n, n')$	Weight of edges in the min-cost network
$\mathscr{U}(n, n')$	Capacity of edges in the min-cost network
$\mathscr{G}$	Weighted DAG of the min-cost network
Tables	
$\tau(t, i, j)$	SP travel time (in min) between stations i and j departing from i at time t
$\rho(t, i, j)$	SP travel distance (in miles) between stations i and j departing from i at time t
$\phi(e, o, d)$	Average demand from origin o to destination d entering system at time e
Variables	
$\mathcal{F}_\alpha(n)$	Partial route of shuttle until node n at iteration α
$\mathcal{B}_\alpha(n)$	Partial route of shuttle from node n at iteration α
$\mathcal{X}$	A fixed route of shuttle
$\mathcal{X}^*$	A local optimal route of shuttle
$\varphi_\alpha(n)$	revenue of shuttle by visiting node n at iteration α
φ^*_α	Maximum revenue of shuttle at iteration α
Δ	Relative gap between maximum revenue of two consecutive forward local search
$f^k_m(e, o, d)$	Optimal number of served trips from origin o to destination d with EDT e by alternative m of shuttle k
$\mathcal{G}^k_m$	The reduced network around the base route of shuttle k based on demand profile m
γ	Optimal revenue of shuttle in the min-cost network
θ	Simulation time step
z	Total revenue of all base routes in the offline phase
$\phi^k(e, o, d)$	Average demand from origin o to destination d with EDT at time e assigned to shuttle k
$\eta^k(e, o, d)$	Distance between base route of shuttle k and a request (o, d, e) in the offline phase
$\Phi_m(e, o, d)$	Request counts from origin o to destination d with EDT at time e in the m^{th} demand profile sampled from the Poisson distribution with mean $\phi(e, o, d)$
$\Phi^k_m(e, o, d)$	Request counts from origin o to destination d with EDT at time e in the m^{th} demand profile assigned to shuttle k
$\psi(e, o, d)$	Estimated request counts from origin o to destination d with EDT at time e
$\psi^{ca}(e, o, d)$	Realized request counts from origin o to destination d with EDT at time e
$\psi^{ca}_k(e, o, d)$	Demand from origin o to destination d with EDT at time e assigned to shuttle k
$\psi^{on}_k(p, o, d)$	Number of on-board passengers on shuttle k

Continued on next page

Table* B.*1* – *continued from previous page

Notation	Description
$\psi_k^{sc}(p,o,d)$	Number of scheduled passengers on shuttle k
$\xi^k(e,o,d)$	Proximity between base route of shuttle k and a request in the online phase
$\mathcal{P}_m^k$	Alternative route m for shuttle k computed in the offline framework
$\mathcal{R}_k$	Route of shuttle k in real time
Measures	
$\mathcal{Z}$	Total revenue of all shuttles (in miles) averaged over all runs
$\mathcal{M}$	Matching rate of customers served by all shuttles averaged over all runs
$\mathcal{W}$	Waiting time of served customers averaged over all shuttle, runs, and customers
$\mathcal{D}$	Detour time of served customers averaged over all shuttle, runs, and customers
$\mathcal{U}$	Utilization rate of shuttles' seats averaged over all shuttle, runs, and time steps
$\mathcal{A}$	Number of times shuttles switch their routes averaged over all shuttle and runs
Θ	Average computation time of online framework over all runs and all time steps

B.2 Sensitivity Analysis on ϵ in the Nguyen-Dupuis Case Study (Cont'd)

From the results of Table B.2, B.3, and B.4, we infer that ϵ does not significantly affect the out-of-vehicle waiting time, in-vehicle detour time, and the occupancy of seats in a shuttle under our proposed method.

Table B.5 clearly suggests that the capability of shuttles to switch their routes and the shuttle revenues demonstrate the same trend as we increase the value of ϵ. This table further reveals that the shuttles change their routes more frequently in the second and third scenarios. These scenarios mimic the situation where the realized online demand significantly differs from the historical demand. As a result, it is more likely for shuttles to obtain requests that are not on their current routes. This table also indicates that using demand prediction helps us use our alternative routes more effectively, as the values of $\mathcal{A}_1$ are much higher than the values of $\mathcal{A}_2$. Finally, Table B.6 presents the computation time of every time step in our simulation. This table suggests that our proposed method can be easily applied in real time, as the computation time of every time step takes

Table B.2: Impact of threshold ϵ on the average waiting time, $\mathcal{W}$, in the Nguyen-Dupuis case study

μ	σ	β	**Full**			**No Pred.**			**No Alter.**	**Insertion**	**Assignment**
			$\epsilon = 5$	$\epsilon = 10$	$\epsilon = 15$	$\epsilon = 5$	$\epsilon = 10$	$\epsilon = 15$			
0	0	0.95	1.41	1.41	1.40	1.41	1.39	1.37	1.39	1.3	1.81
		0.50	1.41	1.4	1.39						
		0.25	1.41	1.39	1.39						
5	0	0.95	1.46	1.48	1.48	1.53	1.51	1.51	1.48	1.56	1.92
		0.50	1.48	1.48	1.49						
		0.25	1.50	1.50	1.50						
0	5	0.95	1.42	1.42	1.43	1.43	1.42	1.42	1.41	1.36	1.82
		0.50	1.42	1.41	1.42						
		0.25	1.43	1.41	1.42						

Table B.3: Impact of threshold ϵ on the average detour time, $\mathcal{D}$, in the Nguyen-Dupuis case study

μ	σ	β	**Full**			**No Pred.**			**No Alter.**	**Insertion**	**Assignment**
			$\epsilon = 5$	$\epsilon = 10$	$\epsilon = 15$	$\epsilon = 5$	$\epsilon = 10$	$\epsilon = 15$			
0	0	0.95	3.86	3.82	3.83	3.78	3.67	3.74	3.81	1.91	1.95
		0.50	3.83	3.78	3.78						
		0.25	3.80	3.69	3.79						
5	0	0.95	3.98	3.97	3.96	3.86	3.82	3.85	4.05	2.28	2.24
		0.50	3.94	3.96	3.89						
		0.25	3.87	3.87	3.85						
0	5	0.95	3.87	3.86	3.85	3.82	3.77	3.77	3.87	2.02	2.01
		0.50	3.85	3.81	3.85						
		0.25	3.83	3.78	3.82						

less than a few milliseconds. This table also indicates that the computation time of the Full model is more than other configurations. This is not surprising, as we take alternative routes as well as future demand into account when conducting a cost-benefit analysis in the Full method.

Table B.4: Impact of threshold ϵ on the average seat utilization rate, $\mathcal{U}$, in the Nguyen-Dupuis case study

μ	σ	β	**Full**			**No Pred.**			**No Alter.**	**Insertion**	**Assignment**
			$\epsilon = 5$	$\epsilon = 10$	$\epsilon = 15$	$\epsilon = 5$	$\epsilon = 10$	$\epsilon = 15$			
0	0	0.95	45.84	45.85	45.75	44.89	44.49	44.89	45.7	30.5	28.95
		0.50	45.66	45.69	45.52						
		0.25	45.22	45.29	45.17						
5	0	0.95	61.91	62.32	62.17	60.13	59.8	59.47	61.27	46.29	43.85
		0.50	61.54	62.12	61.64						
		0.25	60.68	61.45	60.74						
0	5	0.95	47.43	47.59	47.45	46.28	46.47	46.13	47.09	34.21	32.58
		0.50	47.39	47.46	47.25						
		0.25	46.95	47.11	46.8						

Table B.5: Impact of threshold ϵ on the average number of switches between alternative routes, $\mathcal{A}$, in the Nguyen-Dupuis case study

μ	σ	β	**Full**			**No Pred.**			**No Alter.**	**Insertion**	**Assignment**
			$\epsilon = 5$	$\epsilon = 10$	$\epsilon = 15$	$\epsilon = 5$	$\epsilon = 10$	$\epsilon = 15$			
0	0	0.95	2.78	3.01	2.71	0.49	0.49	0.50	–	–	–
		0.50	2.8	3.04	2.77						
		0.25	2.88	3.1	2.92						
5	0	0.95	3.9	3.97	3.7	0.84	0.9	1.06	–	–	–
		0.50	3.95	4.06	3.87						
		0.25	3.99	4.09	3.94						
0	5	0.95	2.87	3.07	2.85	0.55	0.59	0.66	–	–	–
		0.50	2.99	3.16	3.01						
		0.25	3.09	3.2	3.13						

Table B.6: Impact of threshold ϵ on the average computation time of every time step, $\mathcal{T}$, in the Nguyen-Dupuis case study

μ	σ	β	**Full**			**No Pred.**			**No Alter.**	**Insertion**	**Assignment**
			$\epsilon = 5$	$\epsilon = 10$	$\epsilon = 15$	$\epsilon = 5$	$\epsilon = 10$	$\epsilon = 15$			
0	0	0.95	21.00	23.61	21.34	15.00	17.49	15.30	1.28	21.30	1.70
		0.50	20.32	23.07	21.05						
		0.25	20.00	22.71	20.57						
5	0	0.95	27.77	31.11	28.39	19.29	22.28	19.40	1.94	46.34	3.73
		0.50	26.25	29.86	27.05						
		0.25	25.08	28.84	26.07						
0	5	0.95	21.27	23.65	21.54	14.96	17.45	15.18	1.38	23.32	1.94
		0.50	20.42	23.32	20.88						
		0.25	20.16	22.70	20.34						

B.3 Sensitivity Analysis on the Distribution of Arrivals in the Nguyen-Dupuis Case Study

The results of Poisson and Uniform arrivals are summarized in Tables B.7, B.8, B.9, B.10, B.11, B.12, and B.13. Also, Table B.17 contains the p-value of the pairwise comparison between the two distributions for all measures, methods, and demand scenarios. From Table B.7, we infer that changing the arrival distribution from Poisson to Uniform results in a slight increase in the revenue of all methods, albeit the differences more or less stay the same. This might seem contradictory given our assumption of Poisson arrivals for generating the routes in the offline phase. However, it should be noted that a Poisson arrival indicates exponential inter-arrival rates. Given the memory-less property of the exponential inter-arrival times, the Poisson arrivals simulate a highly stochastic system. In contrast, arriving uniformly over time increases the chance of a customer to become available for at least one shuttle. This is why the revenue increases not only for the three configurations of our method, but also for the two benchmark methods. Table B.8, however, does not show a meaningful change in the matching rate of methods under the Uniform distribution.

Table B.9 does not show a significant change in the waiting times. However, one can infer

Table B.7: Impact of arrival distribution on the average revenue, $\mathcal{Z}$, in the Nguyen-Dupuis case study

μ	σ	β	**Full**		**No Pred.**		**No Alter.**		**Insertion**		**Assignment**	
			Poisson	Uniform	Poisson	Uniform	Poisson	Uniform	Poisson	Uniform	Poisson	Uniform
0	0	0.95	2748.9	2817.14	2677.08	2735.1	2731.89	2785.81	2003.29	2067.86	1914.25	1962.24
		0.50	2741.19	2811.7								
		0.25	2725.04	2785.94								
5	0	0.95	3729.16	3770.66	3583.72	3640.84	3629.71	3675.12	2997.8	3127.34	2835.92	2930.15
		0.50	3713.06	3749.98								
		0.25	3677.31	3717.95								
0	5	0.95	2847.92	2937.48	2785.86	2852.84	2804.75	2884.05	2242.68	2274.21	2137.66	2196.38
		0.50	2842.04	2936.15								
		0.25	2823.32	2914.35								

Table B.8: Impact of arrival distribution on the average matching rate, $\mathcal{M}$, in the Nguyen-Dupuis case study

μ	σ	β	**Full**		**No Pred.**		**No Alter.**		**Insertion**		**Assignment**	
			Poisson	Uniform	Poisson	Uniform	Poisson	Uniform	Poisson	Uniform	Poisson	Uniform
0	0	0.95	0.74	0.74	0.73	0.73	0.75	0.75	0.57	0.58	0.54	0.54
		0.50	0.74	0.75								
		0.25	0.74	0.74								
5	0	0.95	0.56	0.55	0.55	0.54	0.56	0.55	0.47	0.47	0.44	0.44
		0.50	0.56	0.55								
		0.25	0.57	0.55								
0	5	0.95	0.67	0.67	0.66	0.66	0.67	0.67	0.55	0.54	0.52	0.51
		0.50	0.67	0.67								
		0.25	0.67	0.67								

from Table B.10 a slight increase in the detour time of the three configurations of our method by using Uniform distributions. This again originates from the fact that the shuttle routes are built based on Poisson distribution. Hence, although the shuttles can still visit stations, but the order in which stations are visited may result in longer detours. The results in Table B.11 are also consistent with our previous observations. As the matching rate and detour times go up under the Uniform distribution, we expect higher utilization rates.

Table B.9: Impact of arrival distribution on the average waiting time, $\mathcal{W}$, in the Nguyen-Dupuis case study

μ	σ	β	**Full**		**No Pred.**		**No Alter.**		**Insertion**		**Assignment**	
			Poisson	Uniform	Poisson	Uniform	Poisson	Uniform	Poisson	Uniform	Poisson	Uniform
0	0	0.95	1.41	1.39	1.39	1.37	1.38	1.36	1.3	1.33	1.82	1.83
		0.5	1.4	1.37								
		0.25	1.39	1.37								
5	0	0.95	1.48	1.5	1.51	1.55	1.47	1.52	1.56	1.59	1.94	1.93
		0.5	1.48	1.5								
		0.25	1.5	1.52								
0	5	0.95	1.42	1.41	1.42	1.39	1.4	1.37	1.36	1.39	1.82	1.87
		0.5	1.41	1.4								
		0.25	1.41	1.4								

Table B.10: Impact of arrival distribution on the average detour time, $\mathcal{D}$, in the Nguyen-Dupuis case study

μ	σ	β	**Full**		**No Pred.**		**No Alter.**		**Insertion**		**Assignment**	
			Poisson	Uniform	Poisson	Uniform	Poisson	Uniform	Poisson	Uniform	Poisson	Uniform
0	0	0.95	3.82	3.92	3.67	3.79	3.77	3.88	1.91	1.87	1.9	1.97
		0.50	3.78	3.9								
		0.25	3.69	3.84								
5	0	0.95	3.97	4.15	3.82	4	4.02	4.19	2.28	2.22	2.22	2.27
		0.50	3.96	4.12								
		0.25	3.87	4.04								
0	5	0.95	3.86	3.99	3.77	3.85	3.85	3.94	2.02	2.07	2.05	2.08
		0.50	3.81	3.95								
		0.25	3.78	3.88								

In addition Table B.12 may suggest that under the Uniform distribution, shuttles alter their routes less frequently. However, this change in the value of $\mathcal{A}$ does not seem to be statistically significant. Finally, Table B.13 shows a slight decrease in the computation times of our proposed method under the Uniform arrivals case. This is due to the fact that Poisson arrivals may give rise to larger pools of customers in some time steps. Hence the capacity of arcs in the min-cost flow network increases, which causes the optimization to take longer to solve. However, it is worth mentioning that the magnitude of decrease in computation times seem to be negligible. Overall, we can conclude that the performance of the proposed framework is robust to the choice of customers' arrival distribution.

Table B.11: Impact of arrival distribution on the average seat utilization rate of shuttles, $\mathcal{U}$, in the Nguyen-Dupuis case study

μ	σ	β	Full		No Pred.		No Alter.		Insertion		Assignment	
			Poisson	Uniform	Poisson	Uniform	Poisson	Uniform	Poisson	Uniform	Poisson	Uniform
0	0	0.95	45.85	47.16	44.49	45.68	45.61	46.71	30.5	31.32	29.04	29.84
		0.50	45.69	47.04								
		0.25	45.29	46.55								
5	0	0.95	62.32	63.39	59.8	61.15	61.23	62.35	46.29	47.92	43.57	45.04
		0.50	62.12	63.09								
		0.25	61.45	62.49								
0	5	0.95	47.59	49.29	46.47	47.76	47.02	48.49	34.21	34.76	32.59	33.52
		0.50	47.46	49.22								
		0.25	47.11	48.78								

Table B.12: Impact of arrival distribution on the average number of switches between alternative routes, $\mathscr{A}$, in the Nguyen-Dupuis case study

μ	σ	β	Full		No Pred.		No Alter.		Insertion		Assignment	
			Poisson	Uniform	Poisson	Uniform	Poisson	Uniform	Poisson	Uniform	Poisson	Uniform
0	0	0.95	3.01	2.99	0.49	0.53	–	–	–	–	–	–
		0.5	3.04	3.05								
		0.25	3.1	3.06								
5	0	0.95	3.97	3.78	0.9	0.89	–	–	–	–	–	–
		0.5	4.06	3.93								
		0.25	4.09	3.96								
0	5	0.95	3.07	3.1	0.59	0.62	–	–	–	–	–	–
		0.5	3.16	3.16								
		0.25	3.2	3.16								

Table B.13: Impact of arrival distribution on the average computation time of every time step, Θ, in the Nguyen-Dupuis case study

μ	σ	β	Full		No Pred.		No Alter.		Insertion		Assignment	
			Poisson	Uniform	Poisson	Uniform	Poisson	Uniform	Poisson	Uniform	Poisson	Uniform
0	0	0.95	23.61	22.78	17.49	15.48	1.71	1.41	21.30	19.49	2.20	1.79
		0.50	23.07	22.43								
		0.25	22.71	21.05								
5	0	0.95	31.11	30.75	22.28	21.84	2.62	2.33	46.34	42.02	4.72	4.35
		0.50	29.86	29.33								
		0.25	28.84	28.45								
0	5	0.95	23.65	22.76	17.45	16.78	1.51	1.46	23.32	23.01	2.14	2.12
		0.50	23.32	23.00								
		0.25	22.70	22.36								

B.4 Paired T-Test P-Values

Table B.14: Paired t-test p-value for all pairwise comparison of the results in the Nguyen-Dupuis case study

(a) $\mathcal{Z}$ for $(\mu, \sigma) = (0, 0)$

	$\overline{\mathcal{Z}}_1^{0.50}$	$\overline{\mathcal{Z}}_1^{0.25}$	$\overline{\mathcal{Z}}_2$	$\overline{\mathcal{Z}}_3$	$\overline{\mathcal{Z}}_4$	$\overline{\mathcal{Z}}_5$
$\overline{\mathcal{Z}}_1^{0.75}$	0.00	0.00	0.00	0.01	0.00	0.00
$\overline{\mathcal{Z}}_1^{0.50}$		0.00	0.00	0.10	0.00	0.00
$\overline{\mathcal{Z}}_1^{0.25}$			0.00	0.25	0.00	0.00
$\overline{\mathcal{Z}}_2$				0.00	0.00	0.00
$\overline{\mathcal{Z}}_3$					0.00	0.00
$\overline{\mathcal{Z}}_4$						0.00

(b) $\mathcal{Z}$ for $(\mu, \sigma) = (5, 0)$

	$\overline{\mathcal{Z}}_1^{0.50}$	$\overline{\mathcal{Z}}_1^{0.25}$	$\overline{\mathcal{Z}}_2$	$\overline{\mathcal{Z}}_3$	$\overline{\mathcal{Z}}_4$	$\overline{\mathcal{Z}}_5$
$\overline{\mathcal{Z}}_1^{0.75}$	0.02	0.00	0.00	0.00	0.00	0.00
$\overline{\mathcal{Z}}_1^{0.50}$		0.00	0.00	0.00	0.00	0.00
$\overline{\mathcal{Z}}_1^{0.25}$			0.00	0.00	0.00	0.00
$\overline{\mathcal{Z}}_2$				0.00	0.00	0.00
$\overline{\mathcal{Z}}_3$					0.00	0.00
$\overline{\mathcal{Z}}_4$						0.00

(c) $\mathcal{Z}$ for $(\mu, \sigma) = (0, 5)$

	$\overline{\mathcal{Z}}_1^{0.50}$	$\overline{\mathcal{Z}}_1^{0.25}$	$\overline{\mathcal{Z}}_2$	$\overline{\mathcal{Z}}_3$	$\overline{\mathcal{Z}}_4$	$\overline{\mathcal{Z}}_5$
$\overline{\mathcal{Z}}_1^{0.75}$	0.18	0.00	0.00	0.00	0.00	0.00
$\overline{\mathcal{Z}}_1^{0.50}$		0.00	0.00	0.00	0.00	0.00
$\overline{\mathcal{Z}}_1^{0.25}$			0.00	0.03	0.00	0.00
$\overline{\mathcal{Z}}_2$				0.05	0.00	0.00
$\overline{\mathcal{Z}}_3$					0.00	0.00
$\overline{\mathcal{Z}}_4$						0.00

(d) $\mathcal{M}$ for $(\mu, \sigma) = (0, 0)$

	$\overline{\mathcal{M}}_1^{0.50}$	$\overline{\mathcal{M}}_1^{0.25}$	$\overline{\mathcal{M}}_2$	$\overline{\mathcal{M}}_3$	$\overline{\mathcal{M}}_4$	$\overline{\mathcal{M}}_5$
$\overline{\mathcal{M}}_1^{0.75}$	0.49	0.19	0.00	0.16	0.00	0.00
$\overline{\mathcal{M}}_1^{0.50}$		0.03	0.00	0.25	0.00	0.00
$\overline{\mathcal{M}}_1^{0.25}$			0.00	0.04	0.00	0.00
$\overline{\mathcal{M}}_2$				0.00	0.00	0.00
$\overline{\mathcal{M}}_3$					0.00	0.00
$\overline{\mathcal{M}}_4$						0.00

(e) $\mathcal{M}$ for $(\mu, \sigma) = (5, 0)$

	$\overline{\mathcal{M}}_1^{0.50}$	$\overline{\mathcal{M}}_1^{0.25}$	$\overline{\mathcal{M}}_2$	$\overline{\mathcal{M}}_3$	$\overline{\mathcal{M}}_4$	$\overline{\mathcal{M}}_5$
$\overline{\mathcal{M}}_1^{0.75}$	0.09	0.01	0.00	0.77	0.00	0.00
$\overline{\mathcal{M}}_1^{0.50}$		0.03	0.00	0.12	0.00	0.00
$\overline{\mathcal{M}}_1^{0.25}$			0.00	0.00	0.00	0.00
$\overline{\mathcal{M}}_2$				0.00	0.00	0.00
$\overline{\mathcal{M}}_3$					0.00	0.00
$\overline{\mathcal{M}}_4$						0.00

(f) $\mathcal{M}$ for $(\mu, \sigma) = (0, 5)$

	$\overline{\mathcal{M}}_1^{0.50}$	$\overline{\mathcal{M}}_1^{0.25}$	$\overline{\mathcal{M}}_2$	$\overline{\mathcal{M}}_3$	$\overline{\mathcal{M}}_4$	$\overline{\mathcal{M}}_5$
$\overline{\mathcal{M}}_1^{0.75}$	0.79	0.39	0.00	0.17	0.00	0.00
$\overline{\mathcal{M}}_1^{0.50}$		0.23	0.00	0.13	0.00	0.00
$\overline{\mathcal{M}}_1^{0.25}$			0.00	0.75	0.00	0.00
$\overline{\mathcal{M}}_2$				0.00	0.00	0.00
$\overline{\mathcal{M}}_3$					0.00	0.00
$\overline{\mathcal{M}}_4$						0.00

Table B.15: Paired t-test p-value for all pairwise comparison of the results in the Manhattan case study

(a) $\mathcal{Z}$

	$\overline{\mathcal{Z}}_1^{0.50}$	$\overline{\mathcal{Z}}_1^{0.25}$	$\overline{\mathcal{Z}}_2$	$\overline{\mathcal{Z}}_3$	$\overline{\mathcal{Z}}_4$	$\overline{\mathcal{Z}}_5$
$\overline{\mathcal{Z}}_1^{0.75}$	0.00	0.00	0.68	0.00	0.00	0.00
$\overline{\mathcal{Z}}_1^{0.50}$		0.00	0.00	0.00	0.00	0.00
$\overline{\mathcal{Z}}_1^{0.25}$			0.00	0.00	0.00	0.00
$\overline{\mathcal{Z}}_2$				0.00	0.00	0.00
$\overline{\mathcal{Z}}_3$					0.00	0.00
$\overline{\mathcal{Z}}_4$						0.00

(b) $\mathcal{M}$

	$\overline{\mathcal{M}}_1^{0.50}$	$\overline{\mathcal{M}}_1^{0.25}$	$\overline{\mathcal{M}}_2$	$\overline{\mathcal{M}}_3$	$\overline{\mathcal{M}}_4$	$\overline{\mathcal{M}}_5$
$\overline{\mathcal{M}}_1^{0.75}$	0.02	0.00	0.00	0.00	0.00	0.00
$\overline{\mathcal{M}}_1^{0.50}$		0.00	0.00	0.00	0.00	0.00
$\overline{\mathcal{M}}_1^{0.25}$			0.00	0.00	0.00	0.00
$\overline{\mathcal{M}}_2$				0.00	0.00	0.00
$\overline{\mathcal{M}}_3$					0.00	0.00
$\overline{\mathcal{M}}_4$						0.24

(c) $\mathcal{A}$

	$\overline{\mathcal{A}}_1^{0.50}$	$\overline{\mathcal{A}}_1^{0.25}$	$\overline{\mathcal{A}}_2$
$\overline{\mathcal{A}}_1^{0.75}$	0.00	0.00	0.00
$\overline{\mathcal{A}}_1^{0.50}$		0.00	0.00
$\overline{\mathcal{A}}_1^{0.25}$			0.00

(d) $\mathcal{W}$

	$\overline{\mathcal{W}}_1^{0.50}$	$\overline{\mathcal{W}}_1^{0.25}$	$\overline{\mathcal{W}}_2$	$\overline{\mathcal{W}}_3$	$\overline{\mathcal{W}}_4$	$\overline{\mathcal{W}}_5$
$\overline{\mathcal{W}}_1^{0.75}$	0.00	0.00	0.00	0.00	0.01	0.00
$\overline{\mathcal{W}}_1^{0.50}$		0.00	0.00	0.00	0.00	0.00
$\overline{\mathcal{W}}_1^{0.25}$			0.00	0.00	0.00	0.00
$\overline{\mathcal{W}}_2$				0.00	0.00	0.62
$\overline{\mathcal{W}}_3$					0.00	0.06
$\overline{\mathcal{W}}_4$						0.00

(e) $\mathcal{D}$

	$\overline{\mathcal{D}}_1^{0.50}$	$\overline{\mathcal{D}}_1^{0.25}$	$\overline{\mathcal{D}}_2$	$\overline{\mathcal{D}}_3$	$\overline{\mathcal{D}}_4$	$\overline{\mathcal{D}}_5$
$\overline{\mathcal{D}}_1^{0.75}$	0.00	0.08	0.03	0.00	0.00	0.00
$\overline{\mathcal{D}}_1^{0.50}$		0.00	0.00	0.00	0.00	0.00
$\overline{\mathcal{D}}_1^{0.25}$			0.02	0.00	0.00	0.00
$\overline{\mathcal{D}}_2$				0.00	0.00	0.00
$\overline{\mathcal{D}}_3$					0.00	0.00
$\overline{\mathcal{D}}_4$						0.72

(f) $\mathcal{U}$

	$\overline{\mathcal{U}}_1^{0.50}$	$\overline{\mathcal{U}}_1^{0.25}$	$\overline{\mathcal{U}}_2$	$\overline{\mathcal{U}}_3$	$\overline{\mathcal{U}}_4$	$\overline{\mathcal{U}}_5$
$\overline{\mathcal{U}}_1^{0.75}$	0.00	0.00	0.54	0.80	0.00	0.00
$\overline{\mathcal{U}}_1^{0.50}$		0.08	0.00	0.00	0.00	0.00
$\overline{\mathcal{U}}_1^{0.25}$			0.00	0.00	0.00	0.00
$\overline{\mathcal{U}}_2$				0.55	0.00	0.00
$\overline{\mathcal{U}}_3$					0.00	0.00
$\overline{\mathcal{U}}_4$						0.01

(g) $\mathcal{T}$

	$\overline{\mathcal{T}}_1^{0.50}$	$\overline{\mathcal{T}}_1^{0.25}$	$\overline{\mathcal{T}}_2$	$\overline{\mathcal{T}}_3$	$\overline{\mathcal{T}}_4$	$\overline{\mathcal{T}}_5$
$\overline{\mathcal{T}}_1^{0.75}$	0.00	0.00	0.00	0.00	0.00	0.00
$\overline{\mathcal{T}}_1^{0.50}$		0.00	0.00	0.00	0.00	0.00
$\overline{\mathcal{T}}_1^{0.25}$			0.00	0.00	0.00	0.00
$\overline{\mathcal{T}}_2$				0.00	0.00	0.00
$\overline{\mathcal{T}}_3$					0.00	0.00
$\overline{\mathcal{T}}_4$						0.00

Table B.16: Paired t-test p-values for sensitivity analysis on ϵ

Measures	μ	σ	β	**Full**			**No Pred.**		
				(5,10)	**(5,15)**	**(10,15)**	**(5,10)**	**(5,15)**	**(10,15)**
$\mathcal{Z}$	0	0	0.95	0.58	0.53	0.13	0.22	0.67	0.06
			0.50	0.46	0.62	0.10			
			0.25	0.14	0.67	0.01			
	5	0	0.95	0.00	0.00	0.17	0.50	0.01	0.04
			0.50	0.00	0.01	0.03			
			0.25	0.00	0.29	0.00			
	0	5	0.95	0.09	0.72	0.26	0.04	0.64	0.02
			0.50	0.42	0.31	0.02			
			0.25	0.10	0.28	0.01			
$\mathcal{M}$	0	0	0.95	0.64	0.33	0.07	0.34	0.41	0.07
			0.50	0.41	0.26	0.02			
			0.25	0.12	0.65	0.01			
	5	0	0.95	0.01	0.41	0.04	0.77	0.06	0.11
			0.50	0.08	0.84	0.06			
			0.25	0.00	0.57	0.00			
	0	5	0.95	0.21	0.48	0.10	0.44	0.45	0.11
			0.50	0.70	0.09	0.02			
			0.25	0.34	0.07	0.01			
$\mathcal{W}$	0	0	0.95	0.99	0.66	0.71	0.02	0.00	0.07
			0.50	0.16	0.08	0.58			
			0.25	0.05	0.00	0.43			
	5	0	0.95	0.17	0.03	0.39	0.20	0.16	0.81
			0.50	0.80	0.43	0.23			
			0.25	0.97	0.69	0.76			
	0	5	0.95	0.69	0.41	0.15	0.43	0.21	0.62
			0.50	0.39	0.95	0.25			
			0.25	0.28	0.84	0.30			
$\mathcal{D}$	0	0	0.95	0.02	0.19	0.40	0.00	0.20	0.01
			0.50	0.02	0.03	0.79			
			0.25	0.00	0.65	0.00			
	5	0	0.95	0.42	0.38	0.87	0.18	0.69	0.04
			0.50	0.53	0.01	0.00			
			0.25	0.87	0.29	0.39			
	0	5	0.95	0.31	0.25	0.91	0.02	0.09	0.80
			0.50	0.08	0.87	0.03			
			0.25	0.04	0.81	0.15			
$\mathcal{A}$	0	0	0.95	0.00	0.19	0.00	1.00	0.72	0.73
			0.50	0.00	0.68	0.00			
			0.25	0.01	0.45	0.02			
	5	0	0.95	0.37	0.01	0.00	0.13	0.00	0.00
			0.50	0.20	0.27	0.00			
			0.25	0.17	0.58	0.00			
	0	5	0.95	0.01	0.73	0.00	0.35	0.00	0.04
			0.50	0.02	0.85	0.01			
			0.25	0.12	0.49	0.15			
$\mathcal{U}$	0	0	0.95	0.88	0.52	0.19	0.00	1.00	0.00
			0.50	0.81	0.31	0.06			
			0.25	0.64	0.73	0.12			
	5	0	0.95	0.00	0.00	0.08	0.14	0.01	0.15
			0.50	0.00	0.41	0.00			

Table B.16 – continued from previous page

Measures	μ	σ	β	**Full**			**No Pred.**		
				(5,10)	**(5,15)**	**(10,15)**	**(5,10)**	**(5,15)**	**(10,15)**
			0.25	0.00	0.76	0.00			
	0	5	0.95	0.07	0.81	0.22	0.28	0.43	0.07
			0.50	0.49	0.29	0.04			
			0.25	0.12	0.32	0.04			
$\mathcal{T}$	0	0	0.95	0.00	0.00	0.00	0.00	0.00	0.00
			0.50	0.00	0.00	0.00			
			0.25	0.00	0.00	0.00			
	5	0	0.95	0.00	0.00	0.00	0.00	0.00	0.00
			0.50	0.00	0.00	0.00			
			0.25	0.00	0.00	0.00			
	0	5	0.95	0.00	0.00	0.00	0.00	0.00	0.00
			0.50	0.00	0.00	0.00			
			0.25	0.00	0.00	0.00			

Table B.17: Paired t-test p-values for sensitivity analysis on the arrival distribution

Measure	μ	σ	β	**Full**	**No Pred.**	**No Alter.**	**Insertion**	**Assignment**
$\mathcal{Z}$	0	0	0.95	0.02	0.08	0.08	0.05	0.14
			0.50	0.03				
			0.25	0.06				
	5	0	0.95	0.02	0.00	0.02	0.00	0.02
			0.50	0.07				
			0.25	0.04				
	0	5	0.95	0.02	0.04	0.02	0.44	0.12
			0.50	0.01				
			0.25	0.02				
$\mathcal{M}$	0	0	0.95	0.75	0.92	0.91	0.20	0.98
			0.50	0.79				
			0.25	0.95				
	5	0	0.95	0.00	0.00	0.00	0.92	0.84
			0.50	0.00				
			0.25	0.00				
	0	5	0.95	0.41	0.34	0.54	0.01	0.20
			0.50	0.57				
			0.25	0.56				
$\mathcal{W}$	0	0	0.95	0.23	0.46	0.54	0.40	0.47
			0.50	0.17				
			0.25	0.27				
	5	0	0.95	0.22	0.06	0.05	0.29	0.57
			0.50	0.17				
			0.25	0.28				
	0	5	0.95	0.78	0.17	0.08	0.33	0.01
			0.50	0.58				
			0.25	0.48				

Table B.17 – continued from previous page

Measure	μ	σ	β	Full	No Pred.	No Alter.	Insertion	Assignment
$\mathcal{D}$	0	0	0.95	0.01	0.01	0.01	0.51	0.31
			0.50	0.00				
			0.25	0.00				
	5	0	0.95	0.00	0.00	0.00	0.36	0.28
			0.50	0.00				
			0.25	0.00				
	0	5	0.95	0.06	0.21	0.19	0.42	0.56
			0.50	0.07				
			0.25	0.20				
$\mathcal{A}$	0	0	0.95	0.74	0.33	–	–	–
			0.50	0.88				
			0.25	0.52				
	5	0	0.95	0.03	0.78	–	–	–
			0.50	0.08				
			0.25	0.07				
	0	5	0.95	0.81	0.38	–	–	–
			0.50	1.00				
			0.25	0.66				
$\mathcal{U}$	0	0	0.95	0.75	0.92	0.91	0.20	0.98
			0.50	0.79				
			0.25	0.95				
	5	0	0.95	0.00	0.00	0.00	0.92	0.84
			0.50	0.00				
			0.25	0.00				
	0	5	0.95	0.41	0.34	0.54	0.01	0.20
			0.50	0.57				
			0.25	0.56				
$\mathcal{T}$	0	0	0.95	0.00	0.00	0.00	0.00	0.00
			0.50	0.00				
			0.25	0.00				
	5	0	0.95	0.00	0.00	0.00	0.00	0.00
			0.50	0.00				
			0.25	0.00				
	0	5	0.95	0.00	0.00	0.00	0.00	0.00
			0.50	0.00				
			0.25	0.00				

APPENDIX C

Supporting Materials of Chapter 4

C.1 Table of Notations

Table C.1: List of notations.

Notation	Definition
Sets:	
$N = \{n\}$	the set of all users
$\mathcal{R} = \{r\}$	the set of all riders
$\mathcal{D} = \{d\}$	the set of all drivers
$N' = \{n'\}$,$N'' = \{n''\}$	the set of trips in the morning/evening
$\mathcal{R}' = \{r'\}$,$\mathcal{R}'' = \{r''\}$	the set of rider trips in the morning/evening
$\mathcal{D}' = \{d'\}$,$\mathcal{D}'' = \{d''\}$	the set of driver trips in the morning/evening
N'''	the set of users with trips both in the morning and evening
$\mathcal{R}' = \{r'\}$	the set of riders with trips both in the morning and evening
$\mathcal{D}' = \{d'\}$	the set of drivers with trips both in the morning and evening
$A' = \{a'\}$,$A'' = \{a''\}$	the set of potential morning/evening trip matches
$V = \{v\}$	the set of nodes in the min-cost max flow network
$E = \{e\}$	the set of edges in the min-cost max flow network
Parameters:	
$\mathsf{I}(n')$,$\mathsf{I}(n'')$	the origin station of user *n* in the morning/evening
$\mathsf{J}(n')$,$\mathsf{J}(n'')$	the destination station of user *n* in the morning/evening
$\mathsf{T}(n')$,$\mathsf{T}(n'')$	the desired earliest departure time of user *n* in the morning/evening
$\mathsf{Q}(n')$,$\mathsf{Q}(n'')$	the desired latest arrival time of user *n* in the morning/evening
$\mathsf{F}(n)$	the valuation of every minutes of travel time for user *n*
$\mathsf{H}(n)$	the valuation of every mile driven for user *n*
B	the total budget available for monetary incentives
$\mathsf{U}(r'\|d')$,$\mathsf{U}(r''\|d'')$	the valuation of rider *r* if matched to driver *d* in the morning/evening
$\mathsf{U}(d'\|r')$,$\mathsf{U}(d''\|r'')$	the valuation of driver *d* if matched to rider *r* in the morning/evening
$\mathsf{W}(e)$	the original gain of pair of trips *e*

Continued on next page

Table C.1 – continued from previous page

Notation	Definition
$\overline{\gamma}(d'\|r')$,$\overline{\gamma}(d''\|r'')$	the optimal BA incentive of driver d if matched with rider r in the morning/evening
$\overline{\gamma}(r'\|d')$,$\overline{\gamma}(r''\|d'')$	the optimal BA incentive of rider r if matched with driver d in the morning/evening
$\overline{\lambda}(d', r')$,$\overline{\lambda}(r'', d'')$	the optimal IR incentive of pair $(d', r')/(r'', d'')$ if matched in the morning/evening
$\bar{t}(d'\|r')$,$\bar{t}(d''\|r'')$	the optimal trip start time of driver d if matched with rider r in the morning/evening
$\bar{t}(r'\|d')$,$\bar{t}(r''\|d'')$	the optimal trip start time of rider r if matched with driver d in the morning/evening
$\Psi(d', r')$,$\Psi(r'', d'')$	the optimal subsidy allocated to the pair $(d', r')/(r'', d'')$ if matched in the morning/evening
s ,t	the source/target node in the min-cost max flow network
ϵ	an infinitesimal positive, real number
$\mathsf{C}(e)$	the difference between the adjusted gain and subsidy of pair of trips e
$\mathsf{C}_{\max}$	the largest value of C
OPT	the optimal objective of the problem in (4.7)
β	the Lagrange multiplier
$\mathsf{C}_{\beta}(e)$	the coefficient of pair of trips e in LR subproblem for a given β
$\alpha(e)$	the weight of edge e in the Gasoline Lemma
ϕ	a flat tax rate for budget balanced variant of TIP
π	a rate for dividing the budget between the morning and evening periods
Functions:	
G	the min-cost flow network of a ridesharing system with ride-back guarantee
G'	the graph containing the cycles between two flows S_l and S_h
G''	the min-cost flow network of an infeasible flow S
$\delta^{+}(v)$,$\delta^{-}(v)$	the in-going and out-going edges of node v in G
Tables:	
τ	the shortest-path travel times matrix for the stations
ρ	the shortest-path driving distances matrix for the stations
Variables:	
$x(e)$	1 if the driver and rider trip in e are matched, and 0 otherwise
$t(n')$,$t(n'')$	the morning/evening trip start time of user n
$q(n')$,$q(n'')$	the morning/evening trip end time of user n
$\gamma^{-}(n')$,$\gamma^{-}(n'')$	the morning/evening time extension of the earliest departure time for user n
$\gamma^{+}(n')$,$\gamma^{+}(n'')$	the morning/evening time extension of the latest arrival time for user n
$\gamma(n')$,$\gamma(n'')$	the BA incentive for the trip of user n in the morning/evening
$\lambda(e)$	the IR incentive for the pair of trips e
$w(e)$	the adjusted gain of pair of trips e including the subsidies
$S_h = \{e\}$,$S_l = \{e\}$	A budget-feasible/infeasible flow that maximizes the LR subproblem
$\tilde{S} = \{e\}$	A near-optimal solution obtained from Algorithm 4.2

C.2 Proof of Proposition 1

Proof. Let us consider different possible scenarios of subsequence Y and show that in each scenario we can obtain a feasible flow by removing a set of edges from S. Let us denote the first and last edges of subsequence Y as $e_1 = (i_1, j_1)$ and $e_2 = (i_2, j_2)$, respectively. For any node in subsequence Y which is not incident to these 2 edges, it is easy to show that the flow conservation constraint is satisfied in $S = (S_h \cup Y_f) \setminus Y_b$. However, this is not the case for i_1 and j_2.

From the assumptions on Y and the procedure described in the proof of the Gasoline Lemma, we infer that e_1 is one of the following six scenarios: $(i)\,(d', r')$, $(ii)(r', d')$, $(iii)(d'', r'')$, $(iv)(r'', d'')$, $(v)(r', r'')$, and $(vi)(r'', r')$. In scenario (i), by adding (d', r') to S_h the flow conservation will be violated if d' is matched to another rider trip in S_h. Therefore, we must remove edge (d', r') from S. Moreover, if $r \in \mathcal{R}'''$, we may have to remove $(r'', d_s(r''))$ from S in the worst case. Note that $d_s(r'')$ is the driver trip matched with rider r in the evening. In scenario (ii), removing (d', r') from S_h will violate flow conservation for node r'' in S if $r \in \mathcal{R}'''$. Therefore, we may have to remove edge $(r'', d_s(r''))$ in the worst case. Using the same line of reasoning as in scenario (ii), we can show that we have to remove (r'', d'') for the fourth scenario, and $(r'', d_s(r''))$ for the fifth and sixth scenarios from S in the worst-case. Note that scenario (iii) does not require removing any edges from S. It is worth mentioning that we may also need to remove some of the auxiliary edges in S which involve the source and target nodes, but those edges have no effect on the objective function value.

From the fact that Y is the longest subsequence in X, we conclude that e_2 cannot be of type (d', r'), (r'', d''), and (r'', r'), because their following edges in X will have to be included in Y. As such, e_2 can only follow one of the following scenarios: $(i)(r', d')$, $(ii)(d'', r'')$, and $(iii)(r', r'')$. In the first case, we do not need to remove any edge from S other than the auxiliary edges. In the next two scenarios, however, we have to remove $(d_s(r'), r')$ if $r \in \mathcal{R}'''$.

Let r''_1 and r''_2 be two rider trips in the evening for $r_1 \in \mathcal{R}'''$ and $r_2 \in R'''$. Now, let us consider the following possible cases for Y based on the scenarios for $e_1 = (i_1, j_1)$ and $e_2 = (i_2, j_2)$:

1. $i_1 \neq r''_1$ and $j_2 \neq r''_2$: In this case, the property of Y in (4.14) implies that:

$$\sum_{e \in Y} \alpha(e) = \sum_{e \in Y_f} \mathsf{C}_{\beta^*}(e) - \sum_{e \in Y_b} \mathsf{C}_{\beta^*}(e) \geq 0\,.$$

By adding $\mathsf{C}_{\beta^*}(S_h)$ and $\beta^*\,\Psi(S)$ to both sides of the inequality, we have:

$$\mathsf{C}(S) \geq \mathsf{C}_{\beta^*}(S_h) + \beta^*\,\Psi(S)\,.$$

Also, based on the scenarios discussed above, at most two edges will be removed from S to get a feasible solution for the min-cost max flow in graph G'' in this case. Therefore, $\mathsf{C}(\tilde{S}) \geq \mathsf{C}(S) - 2\,\mathsf{C}_{\max}$, because $\tilde{S}$ is a feasible solution that maximizes the costs in graph G''.

Thus, we conclude that $\mathsf{C}(\tilde{S}) \geq \mathsf{C}_{\beta^*}(S_h) + \beta^* \Psi(S) - 2\,\mathsf{C}_{\max}$.

2. $i_1 = r_1''$ and $j_2 \neq r_2''$: In this case, the property in (4.14) implies that:

$$\mathsf{C}(S) \geq \mathsf{C}_{\beta^*}(S_h) + \beta^* \Psi(S) - \max\{\mathsf{C}(d_l'(r_1'), r_1'),\ \mathsf{C}(d_h'(r_1'), r_1'),\ 0\}\,.$$

   Also, based on the scenarios discussed above, at most one edge will be removed in this case which yields $\mathsf{C}(\tilde{S}) \geq \mathsf{C}(S) - \mathsf{C}_{\max}$. Thus, we again conclude that $\mathsf{C}(\tilde{S}) \geq \mathsf{C}_{\beta^*}(S_h) + \beta^* \Psi(S) - 2\,\mathsf{C}_{\max}$.

3. $i_1 \neq r_1''$ and $j_2 = r_2''$: In this case, the property in (4.14) implies that:

$$\mathsf{C}(S) \geq \mathsf{C}_{\beta^*}(S_h) + \beta^* \Psi(S) + \max\{\mathsf{C}(d_l'(r_2'), r_2'),\ \mathsf{C}(d_h'(r_2'), r_2'),\ 0\}\,.$$

   Based on the scenarios discussed above, at most two edges will be removed due to i_1 and edge $(d_s(r_2'), r_2')$ due to j_2 in this case. As a result, we have $\mathsf{C}(\tilde{S}) \geq \mathsf{C}(S) - 2\,\mathsf{C}_{\max} - \mathsf{C}(d_s(r_2'), r_2')$. Note that $d_s(r_2')$ is either $d_h(r_2')$ or $d_l(r_2')$. Thus, we again conclude that $\mathsf{C}(\tilde{S}) \geq \mathsf{C}_{\beta^*}(S_h) + \beta^* \Psi(S) - 2\,\mathsf{C}_{\max}$.

4. $i_1 = r_1''$ and $j_2 = r_2''$: In this case, the property in (4.14) implies that:

$$\begin{aligned}\mathsf{C}(S) \geq{} & \mathsf{C}_{\beta^*}(S_h) - \beta^* \Psi(S) - \max\{\mathsf{C}(d_l'(r_1'), r_1'),\ \mathsf{C}(d_h'(r_1'), r_1'),\ 0\} + \\ & \max\{\mathsf{C}(d_l'(r_2'), r_2'),\ \mathsf{C}(d_h'(r_2'), r_2'),\ 0\}\,.\end{aligned}$$

Based on the scenarios discussed above, at most one edge will be removed due to i_1 and edge $(d_s(r_2'), r_2')$ due to j_2 in this case. Therefore, $\mathsf{C}(\tilde{S}) \geq \mathsf{C}(S) - 2\,\mathsf{C}_{\max} - \mathsf{C}(d_s(r_2'), r_2')$. Using the same reasoning as in the last two cases, we can again conclude that $\mathsf{C}(\tilde{S}) \geq \mathsf{C}_{\beta^*}(S_h) + \beta^* \Psi(S) - 2\,\mathsf{C}_{\max}$.

In all the cases above, we have that $\mathsf{C}(\tilde{S}) \geq \mathsf{C}_{\beta^*}(S_h) + \beta^* \Psi(S) - 2\,\mathsf{C}_{\max}$. Let us rewrite this inequality as:

$$\mathsf{C}(\tilde{S}) \geq \mathsf{C}_{\beta^*}(S_h) + \beta^* \mathsf{B} - \beta^* \mathsf{B} + \beta^* \Psi(S) - 2\,\mathsf{C}_{\max}\,. \tag{C.1}$$

We know that $\mathsf{C}_{\beta^*}(S_h) + \beta^* \mathsf{B}$ is a solution to the LR subproblem, and thus, we have:

$$\mathsf{C}_{\beta^*}(S_h) + \beta^* \mathsf{B} \geq \mathsf{OPT}\,. \tag{C.2}$$

Also, from the fact that Y is the longest subsequence in X for which $\Psi(S) \leq \mathsf{B}$, we know that there always exist an edge $e_3 \in X \setminus Y$ such that $\Psi(e_3) + \Psi(S) > \mathsf{B}$. Now, consider the two following cases for edge e_3:

1. $\mathsf{C}_{\beta^*}(e_3) \geq 0$: In this case, $\mathsf{C}(e_3) \geq \beta^* \Psi(e_3)$ which yields $-\beta^* \Psi(e_3) \geq -\mathsf{C}(e_3) \geq -\mathsf{C}_{\max}$.
2. $\mathsf{C}_{\beta^*}(e_3) < 0$: Due to the fact that e_3 is a part the min-cost max flow solution in graph

G with costs set as $-\mathsf{C}_{\beta^*}$, this case can happen only if there exists an edge e_4 such that $\mathsf{C}_{\beta^*}(e_3) + \mathsf{C}_{\beta^*}(e_4) \geq 0$ which implies that $\mathsf{C}(e_4) \geq \beta^* \Psi(e_3)$. Thus, again, we have $-\beta^* \Psi(e_3) \geq -\mathsf{C}(e_4) \geq -\mathsf{C}_{\max}$.

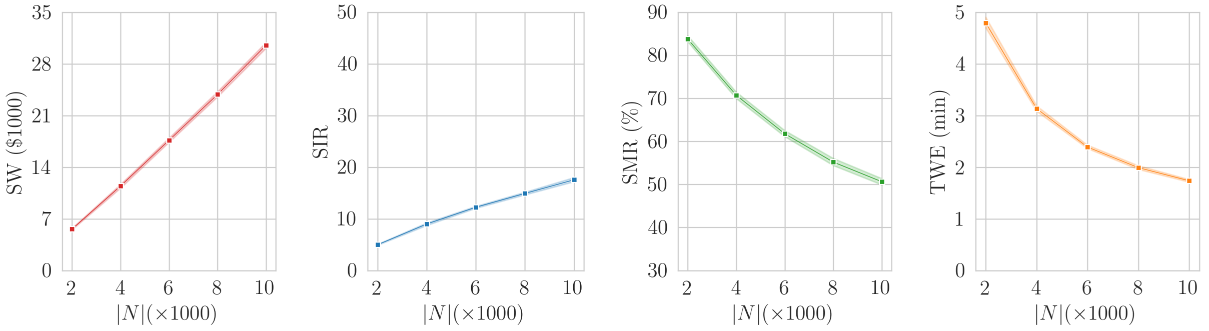
From the cases above, we conclude that

$$-\beta^* \, \mathsf{B} + \beta^* \Psi(S) \geq -\mathsf{C}_{\max} \,. \tag{C.3}$$

Combining the inequalities in (C.1), (C.2), and (C.3) yields $\mathsf{C}(\tilde{S}) \geq \mathsf{OPT} - 3\,\mathsf{C}_{\max}$ and the result follows. □

C.3 Sensitivity Analysis (Cont'd)

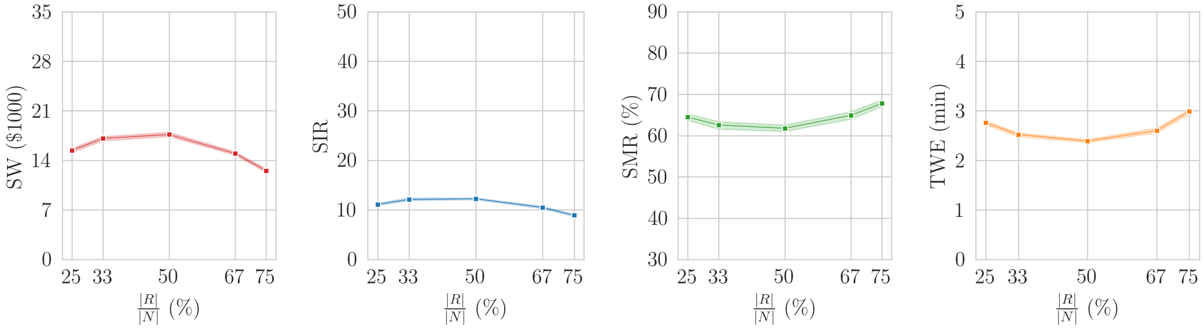
In the base scenario, we assume that the total number of participants in the ridesharing system during the morning and evening peak hours is 6000. In order to study the impact of the number of participants on different performance metrics, we change its value to 2000, 4000, 6000, 8000, and 10,000. The results are presented in Figure C.1. Figure C.1(a) suggests that the social welfare of the system increases significantly as the penetration rate of the system increases. Also, the rate of increase in this figures is linear, which is not surprising because the system is not saturated and adding more users increases the possibility of matching, and thereby, increasing the system's social welfare. The upward trend of the subsidy impact rate in Figure C.1(b) indicates that the added value to the system per one dollar spent on subsidy significantly increases with the number of participants. This is due to the fact that as the number of participants grows, the spatio-temporal proximity of trips increases, which results in (*i*) fewer users requiring the BA incentive to get matched, and (*ii*) a smaller amount of the BA incentive for those who need it (see Figure C.1(d)). Finally C.1(c) and C.1(d) suggest that at least 50% of the matches are subsidized while the average time window extension for those that received the BA incentive does not exceed 5 minutes. The higher spatio-temporal proximity of trips that follows from a higher penetration rate allows the system to subsidize fewer rides and the participants to experience less deviation from their preferred time windows.



(a) Social Welfare (b) Subsidy Impact Rate (c) Subsidized Matches Rate (d) Time Window Extension

Figure C.1: Impact of number of participants

The number of riders and drivers are assumed to be the same in the base scenario. Next, we investigate the impact of changing the percentage of riders from 50 to 25, 33, 67 and 75, respectively. Figure C.2 displays the results of these scenarios for different performance metrics. Figure C.2(a) clearly shows that the social welfare decreases when the percentage of riders diverges from 50%, especially when the percentage of riders is greater than the percentage of drivers. This is partly due to the fact that we are implementing a one-to-one system where a single driver carries at most a single rider in each peak period, and therefore the best results are obtained when there is a balance between the number of riders and drivers. When we have fewer riders than drivers, more drivers are available to serve them, and therefore the percentage of matched riders will be higher than the case where we have more riders than drivers due to (1) more resources to match the riders, and (2) fewer riders to be served. This trend can also be partly explained by the assumption of ride-back guarantees, as 50% of riders in these scenarios register both their morning and evening trips in the system and will be served if and only if both of their trips are served by the drivers in the system. Since this type of requests are harder to satisfy, we expect that the matching rate and social welfare decrease with a higher rate for scenarios above 50%. On top of having a smaller social welfare, Figure C.2(c) shows that a higher percentage of matches need to be subsidized when the percentage of riders is higher than 50. This is why the subsidy impact rates are lower for these scenarios compared to the base scenario, as shown in Figure C.2(b).

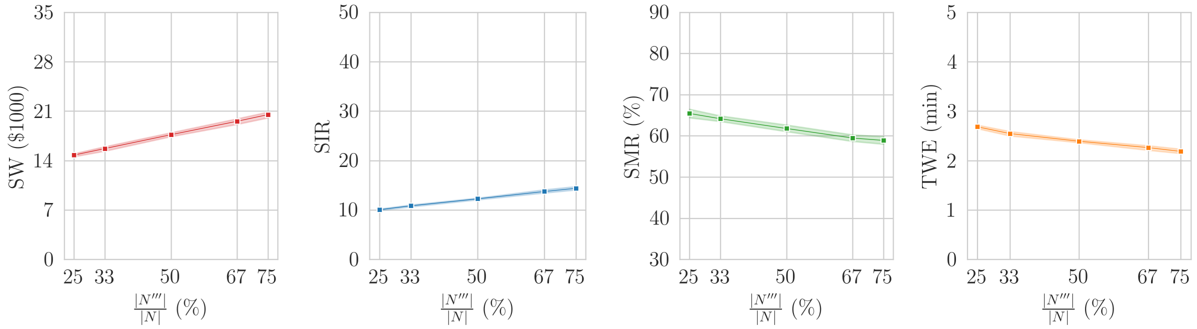


(a) Social Welfare (b) Subsidy Impact Rate (c) Subsidized Matches Rate (d) Time Window Extension

Figure C.2: Impact of percentage of riders

The base scenario assumes that 50% of users register both their morning and evening trips in the system. In this part, we investigate the impact of having lower percentages (25% and 33%) or higher percentages (67%, 75%) of participants be present in both the morning and evening peak hours. Figure C.3 demonstrates how different performance metrics change as this parameter increases from 25% to 75%. Given a fixed number of participants, increasing the value of this parameter clearly increases the number of users (both riders and drivers) in the morning and in the evening. Thus, the number of matches and hence the social welfare increases as shown in Figure C.3(a). Also, the upward trend of the subsidy impact rate in Figure C.3(b) implies that the proposed incentive program is more beneficial when users register both trips in the system. The downward trend of subsidized matched users in Figure C.3(c) originates from the fact that with a higher number of riders and drivers in the morning or evening, the likelihood of finding a match without the help of the BA incentive increases, because of the higher spatio-temporal proximity between trips. Higher spatio-temporal proximity between trips further explains the linearly decreasing trend in Figure

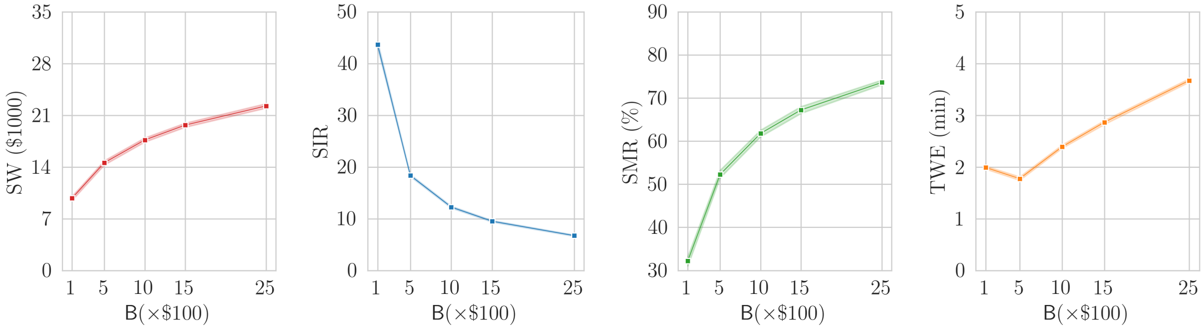
C.3(d), as users will be required to expand their time window less when this parameter increases.



(a) Social Welfare (b) Subsidy Impact Rate (c) Subsidized Matches Rate(d) Time Window Extension

Figure C.3: Impact of percentage of ride-back trips

Figure C.4 demonstrates the results of changing the available budget for subsidy from \$1,000 in the base scenario to \$100 \$500, \$1500 and \$2,500. Figure C.4(a) shows that the social welfare increases sublinearly as we increase the value of B. This is consistent with our earlier results in Section 4.5.6, where we observed that social welfare grows sublinearly until it converges to the maximum possible social welfare. More interestingly, the same trend is also observed in Figure C.4(c). Moreover, the subsidy impact rate in Figure C.4(b) has the reverse trend: we can initially make a huge impact by investing a small amount of budget, but the rate of return-on-investment diminishes as we increase the budget. However, as we try to incentivize higher number of matches, we have to invest more, and the margin of profit gets smaller. Finally, we reach a point that either no further matches can become feasible with the help of the BA incentive or the required budget becomes higher than its subsequent added social welfare. Figure C.4(d) shows an interesting result where the time window expansion decreases slightly as we increase the budget from \$100 to \$500

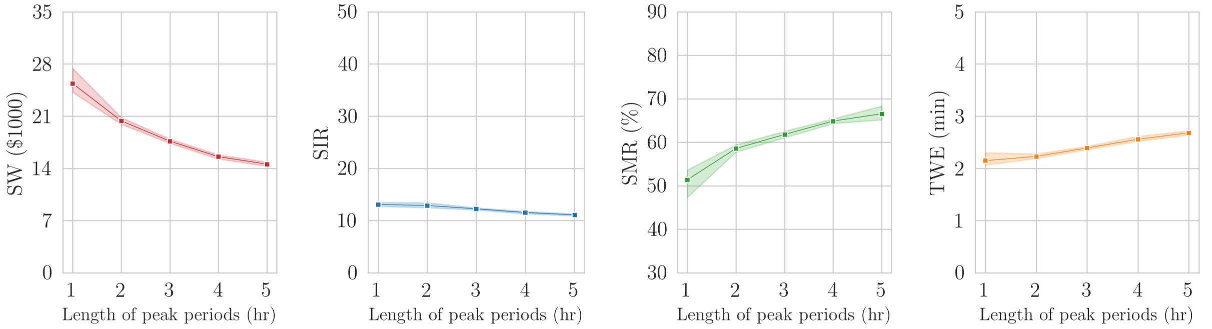


(a) Social Welfare (b) Subsidy Impact Rate (c) Subsidized Matches Rate (d) Time Window Extension

Figure C.4: Impact of total budget

and then increases from \$500 to \$2500. One possible explanation would be that from \$100 to \$500, the rate of increase in the number of subsidized matches is very high (more than 20%) while the amount of increase in time windows for those users is not that much higher, and thus, the average time window expansion slightly decreases.

Finally, Figure C.5 displays the impact of the length of peak hour periods in the morning and the evening on the performance metrics. In the base scenario, we assume that the length of both peak hour periods is 3 hours. Here, we consider tighter periods (1 and 2 hours) and wider periods (4 and 5 hours). Obviously, increasing the length of the peak hour periods causes the trips to spread over a larger horizon, which results in reducing the temporal proximity of trips. This is the main reason behind the descending trend in the social welfare presented in Figure C.5(a). Also, for the same reason, more participants need the BA incentive (see Figure C.5(c)), and the magnitude of allocated incentive per user increases as shown in Figure C.5(d). Moreover, since trips become more temporally sparse, the savings due to sharing rides decreases, which consequently decreases the impact of each dollar spent on subsidy, as shown in Figure C.5(b).



(a) Social Welfare (b) Subsidy Impact Rate (c) Subsidized Matches Rate (d) Time Window Extension

Figure C.5: Impact of the morning and evening peak hours' lengths

APPENDIX D

Supporting Materials of Chapter 5

D.1 Mathematical Formulation for the Minimum Blocking Value Problem

In this formulation, we have three sets of variables: (*i*) matching $x \in \{0,1\}^{|E|}$, (*ii*) payoffs $\rho \in \mathbb{R}^{|V|}$, and (*iii*) blocking value $b \in \mathbb{R}^{|E|}$. The objective in (D.1a) minimizes the total blocking value over all edges in graph G. Constraint (D.1b) makes certain that x is a feasible matching. Constraint (D.1c) ensures that the set of payoffs with the addition of blocking values satisfy the necessary condition for a stable outcome. Constraint (D.1d) enforces our market to be budget-balanced, i.e., the social welfare must be equal to the sum of payoffs. Constraint (D.1e) serves two purposes. First, it sets a cap on the blocking value of each edge as it cannot exceed the potential gain by that match. Second, it ensures that the blocking value of the edges in a matching is equal to 0, which with the help of the two previous sets of constraints implies that the sum of payoffs for the users that are matched together should be equal to the gain from that match ($\rho_u + \rho_v = w_{uv}$). Constraint (D.1f) also serve two purposes. First, it makes sure that the payoff of a matched user does not exceed the gain of that match. Second, it ensures that an unmatched user will have a 0 payoff. Finally, Constraints (D.1g), (D.1h), and (D.1i) are the integrality and non-negativity constraints.

$$\min \quad \sum_{(u,v)\in E} b_{uv} \tag{D.1a}$$

$$\text{s.t.} \quad \sum_{\substack{u\in V:\\(v,u)\in E}} x_{vu} + \sum_{\substack{u\in V:\\(u,v)\in E}} x_{uv} \leq 1\,, \qquad \forall\, v \in V\,, \tag{D.1b}$$

$$\rho_u + \rho_v + b_{uv} \geq w_{uv}\,, \qquad \forall\, (u,v) \in E\,, \tag{D.1c}$$

$$\sum_{(u,v)\in E} w_{uv} x_{uv} = \sum_{v\in V} \rho_v\,, \tag{D.1d}$$

$$b_{uv} \leq w_{uv}(1 - x_{uv})\,, \qquad \forall\, (u,v) \in E\,, \tag{D.1e}$$

$$\rho_v \leq \sum_{\substack{u \in V: \\ (v,u) \in E}} w_{vu} x_{vu} + \sum_{\substack{u \in V: \\ (u,v) \in E}} w_{uv}\, x_{uv}\,, \qquad \forall\, v \in V\,, \tag{D.1f}$$

$$x_{uv} \in \{0, 1\}\,, \qquad \forall\, (u, v) \in E\,, \tag{D.1g}$$

$$\rho_v \geq 0\,, \qquad \forall\, v \in V\,, \tag{D.1h}$$

$$b_{uv} \geq 0\,, \qquad \forall\, (u, v) \in E\,. \tag{D.1i}$$

D.2 Mathematical Formulation for the Maximum Stable Subset Problem

There are four sets of decision variables in this formulation including (*i*) matching $x \in \{0,1\}^{|E|}$, (*ii*) payoffs $\rho \in \mathbb{R}^V$, (*iii*) user inclusion $y \in \{0,1\}^{|V|}$, and (*iv*) edge inclusion $z \in \{0,1\}^{|E|}$. The objective function in (D.2a) aims to maximize the social welfare of the system. Constraint (D.2b) ensures that if included in the system, user v can match to at most one other user. Constraints (D.2c) and (D.2d) combined ensure that the possible coalition (edge) between users u and v must be included, i.e., $z_{uv} = 1$, if and only if both users are included in the system, i.e., $y_u = y_v = 1$. The absence of blocking pairs over included edges is enforced by Constraint (D.2e). Constraint (D.2f) requires that the sum of payoffs must be equal to the system social welfare. Constraint (D.2g) serves 2 purposes: it ensures that (*i*) an unmatched user does not receive a positive payoff, (*ii*) the payoff of user u matched with user v cannot exceed the potential gain of their match. Finally, Constraints (D.2h)-(D.2j) imposes integrality for variables x, y, and z, and Constraint (D.2k) ensures the rationality of payoffs.

$$\begin{aligned}
\max \quad & \sum_{(u,v)\in E} w_{uv}\, x_{uv} && && \text{(D.2a)}\\
\text{s.t.} \quad & \sum_{\substack{u\in V:\\(v,u)\in E}} x_{vu} + \sum_{\substack{u\in V:\\(u,v)\in E}} x_{uv} \le y_v\,, && \forall\, v \in V\,, && \text{(D.2b)}\\
& z_{u,v} \ge y_u + y_v - 1\,, && \forall\, (u,v) \in E\,, && \text{(D.2c)}\\
& z_{u,v} \le \frac{y_u + y_v}{2}\,, && \forall\, (u,v) \in E\,, && \text{(D.2d)}\\
& \rho_u + \rho_v \ge w_{uv} z_{uv}\,, && \forall\, (u,v) \in E\,, && \text{(D.2e)}\\
& \sum_{(u,v)\in E} w_{uv}\, x_{uv} = \sum_{v\in V} \rho_v\,, && && \text{(D.2f)}\\
& \rho_v \le \sum_{\substack{u\in V:\\(u,v)\in E}} w_{uv} x_{vu} + \sum_{\substack{u\in V:\\(v,u)\in E}} w_{vu} x_{vu}\,, && \forall\, v \in V\,, && \text{(D.2g)}\\
& x_{uv} \in \{0,1\}\,, && \forall\, (u,v) \in E\,, && \text{(D.2h)}\\
& y_v \in \{0,1\}\,, && \forall\, v \in V\,, && \text{(D.2i)}\\
& z_{uv} \in \{0,1\}\,, && \forall\, (u,v) \in E\,, && \text{(D.2j)}\\
& \rho_v \ge 0\,, && \forall\, v \in V\,. && \text{(D.2k)}
\end{aligned}$$

APPENDIX E

Supporting Materials of Chapter 6

E.1 Table of Notations

Table E.1: List of notations.

Notation	Definition
Sets:	
$N = \{n_\ell\}$	the set of all users
$R = \{r_i\}$	the set of all riders
$D = \{d_i\}$	the set of all drivers
$E = \{(r_i, d_i)\}$	the set of spatio-temporally feasible trip matches
$X = \{(r_i, d_i)\}$	the set of finalized matches of a mechanism
$R_1 = \{r_i\}$, $D_1 = \{r_i\}$	the set of matched riders/drivers in group 1 of subsidy scheme
$X_1 = \{(r_i, d_j)\}$	the set of finalized matches in group 1 of subsidy scheme
$R_2^{\mathsf{L}} = \{r_i\}$, $D_2^{\mathsf{L}} = \{r_i\}$	the set of riders/drivers in the left market of group 2 of subsidy scheme
$R_2^{\mathsf{R}} = \{r_i\}$, $D_2^{\mathsf{R}} = \{r_i\}$	the set of riders/drivers in the right market of group 2 of subsidy scheme
$X_2^{\mathsf{L}} = \{(r_i, d_j)\}, X_2^{\mathsf{R}} = \{(r_i, d_j)\}$	the set of finalized matches in the left/right market of group 2 of subsidy scheme
Parameters:	
I_{n_ℓ}	the origin station of user n_ℓ
J_{n_ℓ}	the destination station of user n_ℓ
T_{n_ℓ}	the desired earliest departure time of user n_ℓ
Q_{n_ℓ}	the desired latest arrival time of user n_ℓ
δ_{n_ℓ}	the value of every mile driven for user n_ℓ
ϑ_{d_j}	the value of every minute of time window extension for driver d_j
$\theta_{r_i} = \delta_{r_i}$, $\theta_{d_j} = (\delta_{d_j}, \vartheta_{d_j})$	the private information of rider r_i/driver d_j
B	the total budget available for subsidizing drivers
u_{n_ℓ}	the utility of user n_ℓ
V_i , B_i	the true valuation/bid of rider r_i
V_j^i , A_j^i	the true valuation/ask of driver d_j if matched to rider r_i
w_{r_i,d_j}	the weight of pair of trips (r_i, d_j) in the mapping objective function

Continued on next page

Table E.1 – continued from previous page

Notation	Definition
$\bar{\gamma}^{r_i}_{d_j}$	the optimal time window extension of driver d_j if matched with rider r_i
$\bar{t}^{d_j}_{r_i}$, $\bar{t}^{r_i}_{d_j}$	the optimal trip start time of rider r_i/driver d_j if matched together
$\bar{q}^{d_j}_{r_i}$, $\bar{q}^{r_i}_{d_j}$	the optimal trip end time of rider r_i/driver d_j if matched together
Γ	the threshold for extending a user's time window
Υ_{n_ℓ}	the time flexibility of user n_ℓ
Functions:	
$\mathcal{M} = (f, p_{n_\ell})$	the ridesharing mechanism
f	the decision function of the ridesharing mechanism
p_{n_ℓ}	the pricing function of the ridesharing mechanism for user n_ℓ
σ	the mapping function for finding initial matches of the subsidy scheme
Tables:	
τ	the shortest-path travel times matrix for the stations
ρ	the shortest-path driving distances matrix for the stations
Variables:	
x_{r_i,d_j}	1 if driver d_j and rider r_i are matched, and 0 otherwise
t_{n_ℓ}	the trip start time of user n_ℓ
q_{n_ℓ}	the trip end time of user n_ℓ
$\gamma^-_{n_\ell}$, $\gamma^+_{n_\ell}$	the time extension of the earliest departure time/latest arrival time for user n_ℓ
p^R_1 ,	the price of a rider in group 1 of the subsidy scheme
p^D_1 ,	the price of a driver in group 1 of the subsidy scheme
p^{L}_2 , p^{R}_2	the trade price in the left/right market of group 2 of the subsidy scheme
s^{L}_2 , s^{R}_2	the subsidy in the left/right market of group 2 of the subsidy scheme

E.2 Shortest-Path Travel Times and Distances for Nguyen-Dupuis Network

Table E.2: The shortest-path travel times for Nguyen-Dupuis network

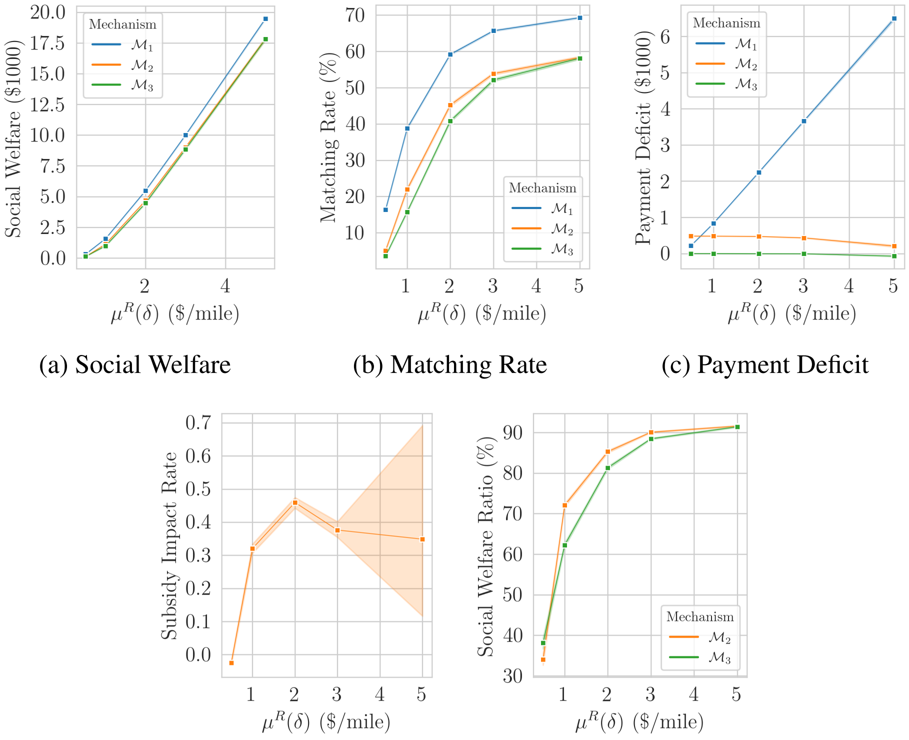
$o\backslash d$	1	2	3	4	5	6	7	8	9	10	11	12	13
1	0	12	14	4	2	6	12	11	4	11	20	1	10
2	12	0	4	18	15	11	5	1	14	9	1	11	8
3	14	4	0	14	12	17	11	5	10	11	3	15	4
4	4	18	14	0	2	6	12	15	4	9	17	5	10
5	2	14	12	2	0	4	10	13	2	9	18	3	8
6	6	11	17	6	4	0	6	10	6	5	14	1	12
7	12	5	11	12	10	6	0	4	12	11	8	7	15
8	11	1	5	16	14	10	4	0	15	10	2	10	9
9	4	14	10	4	2	6	12	15	0	5	13	5	6
10	11	9	11	9	9	5	11	15	5	0	8	6	11
11	13	1	3	17	18	14	8	2	13	8	0	12	7
12	1	11	18	5	3	1	7	10	5	6	15	0	11
13	10	8	4	10	8	12	15	9	6	11	7	11	0

Table E.3: The shortest-path driving distances for Nguyen-Dupuis network

$o\backslash d$	1	2	3	4	5	6	7	8	9	10	11	12	13
1	0	6.7	5.4	3.3	1.2	2.9	5.3	4.3	2.0	4.9	7.2	0.9	4.0
2	7.2	0.0	4.1	8.8	7.6	5.7	3.7	2.5	6.3	4.7	1.5	6.0	5.4
3	5.1	3.7	0.0	5.6	4.0	6.8	4.8	6.2	3.1	5.9	2.7	9.7	1.3
4	3.5	8.0	5.4	0.0	2.4	4.1	6.5	7.8	2.0	3.7	7.0	4.4	4.0
5	1.1	7.8	4.2	2.1	0.0	1.7	4.1	5.4	0.8	3.7	6.0	2.0	2.8
6	3.0	6.4	6.9	4.0	1.9	0.0	2.4	4.0	2.7	2.0	4.3	1.4	4.7
7	5.0	4.0	4.5	6.0	3.9	2.0	0.0	1.6	4.7	4.0	1.9	3.4	5.8
8	4.7	2.4	6.5	7.2	5.1	3.2	1.2	0.0	8.7	7.1	3.9	3.5	7.8
9	2.0	6.0	3.4	2.5	0.9	2.6	5.0	6.3	0.0	1.7	5.0	2.9	2.0
10	5.0	4.3	5.9	4.1	3.9	2.0	4.4	6.0	1.6	0.0	3.3	3.4	3.6
11	8.2	1.0	2.6	7.3	6.0	4.1	2.1	3.5	4.8	3.2	0.0	7.0	3.9
12	1.2	5.8	8.0	4.5	2.4	1.1	3.5	3.4	3.2	3.1	5.4	0.0	5.2
13	3.8	5.1	1.4	4.3	2.7	4.4	6.2	7.6	1.8	3.5	4.1	4.7	0.0

E.3 Sensitivity Analysis (Cont'd)

Let us consider the impact of changes in the mean of rider's value of distance on the performance metrics. Figure E.1 demonstrates the results of changing $\mu^R(\delta)$ from \$2/mile to \$0.5/mile, \$1/mile, \$3/mile and \$5/mile. Figures E.1(a) and E.1(b) show that the social welfare and matching rate of all the mechanisms increase with the rider's value of distance. This is not surprising because the possibility of having a profitable match grows as riders increase their bids, which ultimately leads to a higher number of finalized matches and higher social welfare. However, in this case the externality impact of drivers on the system, i.e., the value that their participation adds to other users, increases with a higher rate than that of the riders. As a result, the increase in payments to the drivers is more than the increase in charged prices of riders. That is why we observe an upward trend in the payment deficit of VCG in Figure E.1(c). On the contrary, our subsidy scheme does not increase the payment to the drivers as their matched riders can already pay their asks, and thus, the payment deficit of $\mathcal{M}_2$ decreases with the mean rider's value of distance. Figure E.1(d) suggests that the subsidy impact rate initially increases up to a point and then starts decreasing. This is due to the reason that with low values of distance, subsidizing the drivers helps the system increase the



(a) Social Welfare

(b) Matching Rate

(c) Payment Deficit

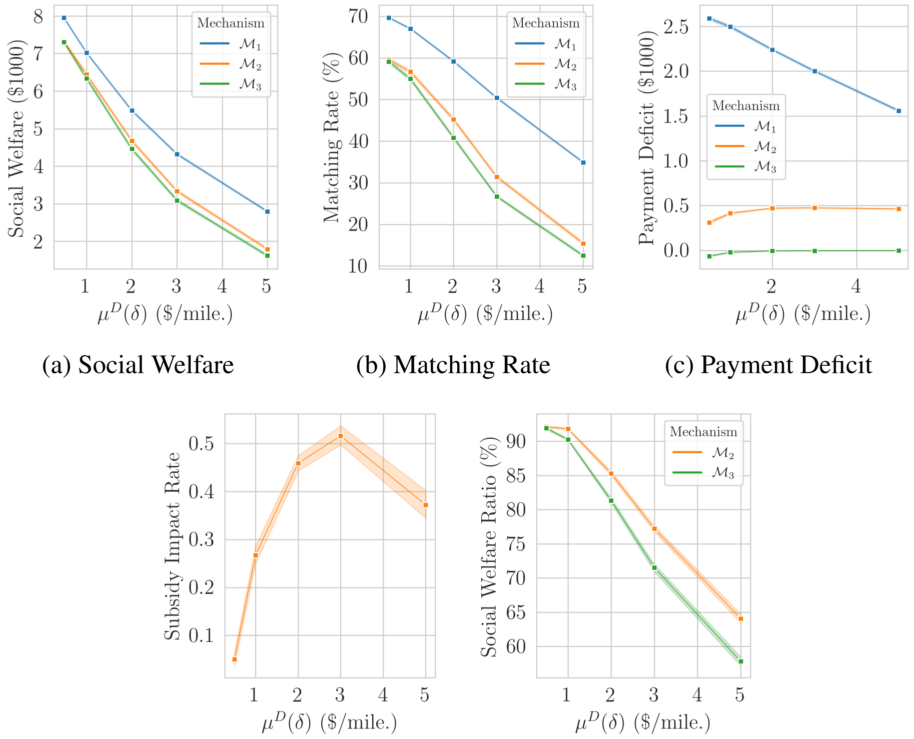
(d) Subsidy Impact Rate

(e) Social Welfare Ratio

Figure E.1: Impact of rider's mean value of distance

number of matches, and thereby, increase the number of profitable matches. However, as riders increase their bids, adding a subsidy only increases the payments to the drivers that could have been matched without the help of a subsidy. That also explains the lower difference between the social welfare ratios of $\mathcal{M}_2$ and $\mathcal{M}_3$ in higher rider's values of distance in Figure E.1(e). Also, note that when $\mu(\delta)$ is very low, the budget increases the matching rate, but it mostly helps match unprofitable matches that decrease the social welfare ratio and yield a negative subsidy impact rate.

Figure E.2 demonstrates the impact of changing the mean of driver's value of distance from \$2/mile to \$0.5/mile, \$1/mile, \$3/mile and \$5/mile. On the contrary to the result of the previous parameter, Figures E.2(a) and E.2(b) show that the social welfare and matching rate of all the mechanisms decrease with increase in the driver's value of distance. This can be explained by the fact that the possibility of having a profitable match decreases as the drivers' asks increase. Thus, we end up having a lower number of finalized matches and subsequently lower social welfare. Also, the externality impact of the drivers on the system in this case decreases with a higher rate than that of the riders. As a result, the decrease in payments to drivers is more than the decrease in charged prices of riders, which translates to a downward trend in the payment deficit of VCG as shown in Figure E.2(c). However, our subsidy scheme has to increase the payment to the drivers as their



(a) Social Welfare

(b) Matching Rate

(c) Payment Deficit

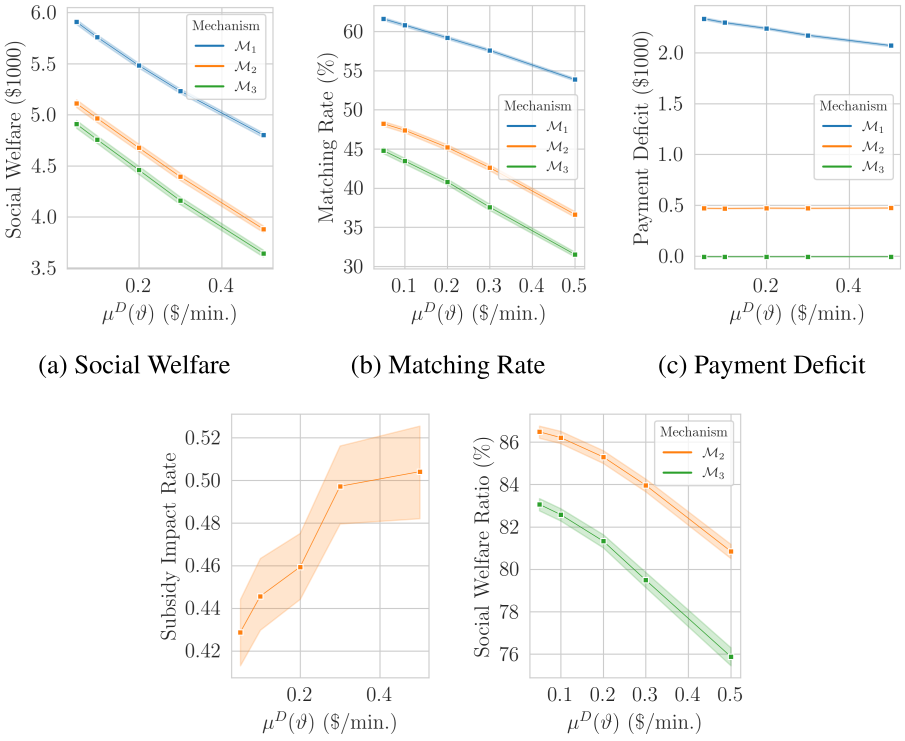
(d) Subsidy Impact Rate

(e) Social Welfare Ratio

Figure E.2: Impact of driver's average value of distance

matched riders cannot pay their asks alone, and thus, the payment deficit of $\mathcal{M}_2$ initially increases with the value of $\mu^D(\delta)$ until it reaches to the maximum allowed deficit at $2/mile. Figure E.2(d) suggests that the subsidy impact rate initially increases up to a point, and then starts decreasing. When the value of $\mu^D(\delta)$ is very low, riders can pay the entirety of drivers' asks, and adding subsidy only increases the payment to the drivers in the matches that could be also matched in the absence of subsidy. As the value of this parameter increases, subsidy allows profitable matches to finalize. However, after a point, the ratio of profitable matches becomes lower compared to the unprofitable matches, and thus, the subsidy impact rate starts to decrease. Since, the objective function of mapping assumes a lower value of distance for drivers, we find initial matches that are less profitable compared to the ones obtained in the efficient matching of VCG. This explains the downward trend of the social welfare ratio for both mechanisms in Figure E.2(e).

In the base scenario, we assumed that the user's value of time window extension, $\mu(\vartheta)$, has a mean of $0.2/min. In order to study the impact of different values of this parameter on the performance metrics, we change its value to $0.05/min., $0.1/min., $0.2/min. and $0.5/min. The results of changing this parameter are presented in Figure E.3. Similar to the previous parameter, the social welfare and matching rate of all mechanisms decrease with an increase in the value of $\mu(\vartheta)$.



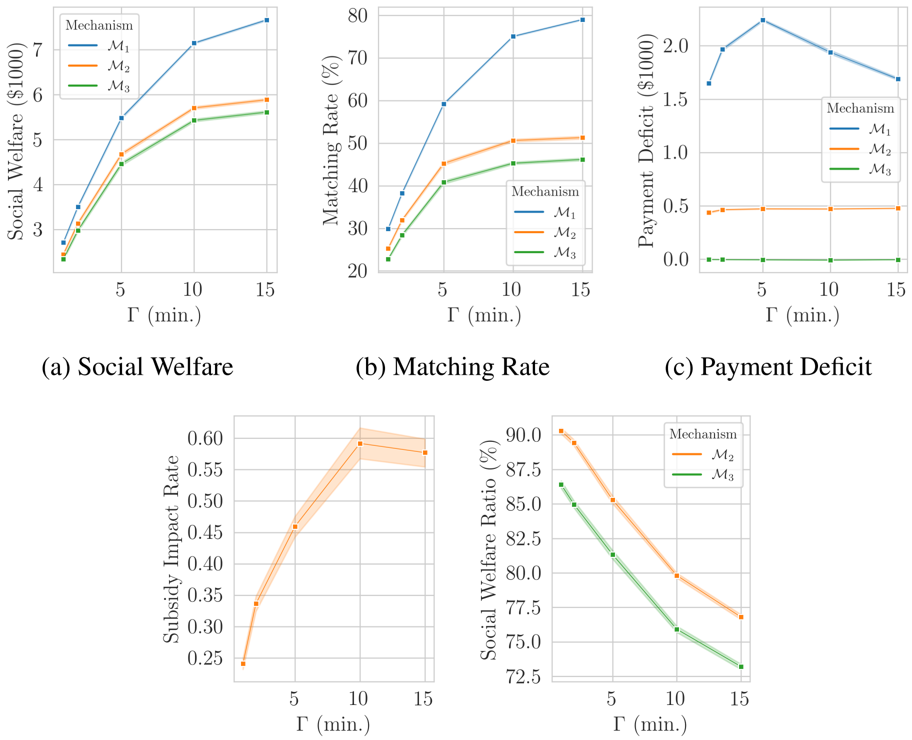
(a) Social Welfare (b) Matching Rate (c) Payment Deficit

(d) Subsidy Impact Rate (e) Social Welfare Ratio

Figure E.3: Impact of driver's average value of time extension

Also, the deficit of VCG decreases due to the same reason described above. However, the magnitude of changes is smaller because of the magnitude of value of time extension compared with that of the value of distance. Also, unlike the previous parameters, the change in $\mu(\vartheta)$ does not seem to have a significant impact on the budget deficit of our subsidy scheme. However, the increase in the value of this parameter significantly increases the subsidy impact rate. Since the mean value of distances for both riders and drivers are the same, the increase in this parameter causes a discrepancy in the bids of riders and asks of drivers. As a result, subsidizing the drivers helps us generate more social welfare compared to its no-deficit variant. Also, this impact increases with larger discrepancy between the bids and asks. Similar to the result of the previous parameter, the downward trend of social welfare ratio in Figure E.3(e) points out that finding the initial matching using a type-independent objective results in higher welfare loss compared to the efficient outcome as the value of $\mu(\vartheta)$ increases.

The maximum threshold for extending the time windows of users is assumed to be 5 minutes in the base scenario. Next, we investigate the impact of changing this parameter from 5 minutes to 1, 2, 10, and 15 minutes, respectively. Figure E.4 displays the results of these scenarios for different performance metrics. As expected, Figures E.4(a) and E.4(b) show that the social welfare and matching rate increase when we let users extend their time windows by a larger amount. However,



(a) Social Welfare

(b) Matching Rate

(c) Payment Deficit

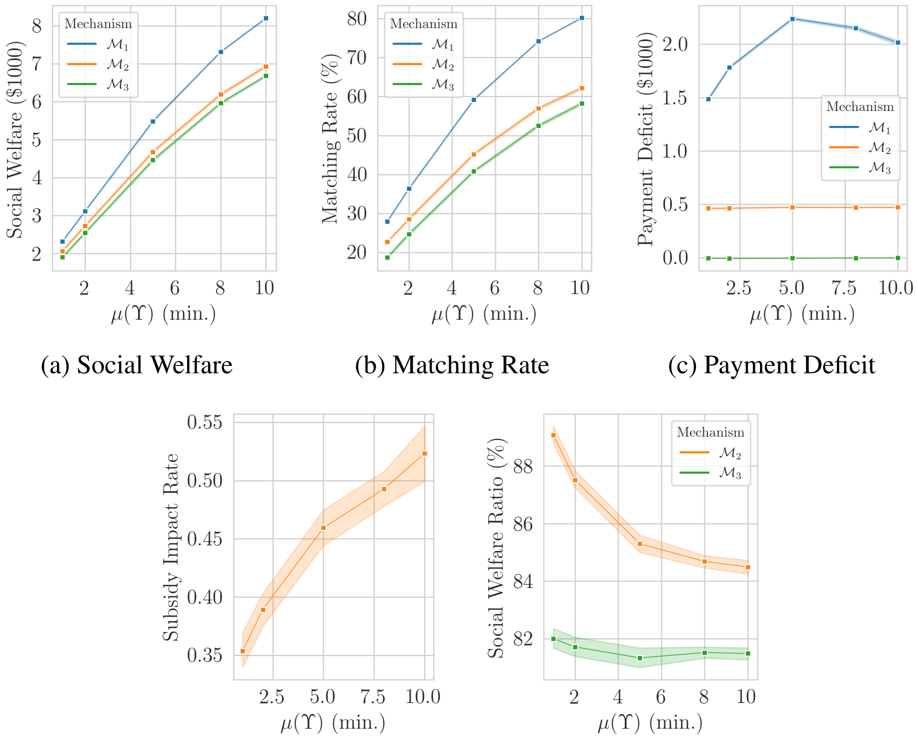
(d) Subsidy Impact Rate

(e) Social Welfare Ratio

Figure E.4: Impact of time window extension threshold

the sub-linear trend in these figures along with Figure E.4(e) indicate that using a type-independent mapping function in step 1 of our subsidy scheme seems to have a huge negative impact on the social welfare. Although the social welfare ratio goes down, Figure E.4(d) shows an upward trend for the subsidy impact rate, which can be again explained by the discrepancy between the asks and bids of users in higher values of Γ. Also, Figure E.4(c) suggests that letting this parameter increase from 1 to 10 minutes initially causes the deficit of VCG to increase and then decrease. One possible explanation for this trend is that increasing Γ initially only increases the number of feasible matches, while the available options for users is low. As a result, the externality impact of users is high. As we let Γ take larger values, the number of options for riders and drivers starts to increase, and as a result, the externality impact of them on the system goes down.

Finally, Figure E.5 displays the impact of increasing the average time flexibility of users on the performance metrics. In the base scenario, we assume that the average time flexibility of users is 5 minutes. Here, we change the value of this parameter from 5 minutes to 1, 2, 8, and 10 minutes. Obviously, increasing the time flexibility of users increases the temporal feasibility of trips. Therefore, it is not surprising to observe the increase in social welfare and matching rate of all mechanisms in Figures E.5(a) and E.5(b), respectively. Also, the quadratic trend in the payment



(a) Social Welfare

(b) Matching Rate

(c) Payment Deficit

(d) Subsidy Impact Rate

(e) Social Welfare Ratio

Figure E.5: Impact of time flexibility

deficit of VCG as a result of an increase in $\mu(\Upsilon)$, as shown in Figure E.5(c), can be attributed to the same reasoning provided for the previous parameter. In Figure E.5(d), we observe an upward trend for the subsidy impact rate. This is due to the fact that increasing the time flexibility lowers the need of drivers for extending their time windows. As a result, the discrepancy between the bids of riders and asks of drivers decreases. Given a set budget for subsidy, we can create more profitable matches, and hence, $\mathcal{M}_2$ yields a higher social welfare compared to $\mathcal{M}_3$. However, Figure E.5(e) shows that the difference between the social welfare of these two mechanisms compared to the optimal social welfare decreases with the value of $\mu(\Upsilon)$. Thus, similar to the conclusions drawn for increasing the number of participants, for the systems in which users provide very flexible schedules, subsidizing time window extension does not have a significant impact on the social welfare of the system.

BIBLIOGRAPHY

Abdolmaleki, M., Masoud, N., and Yin, Y. (2019). Vehicle-to-vehicle wireless power transfer: Paving the way toward an electrified transportation system. *Transportation Research Part C: Emerging Technologies*, 103:261–280.

Abeledo, H. G. and Rothblum, U. G. (1994). Stable matchings and linear inequalities. *Discrete Applied Mathematics*, 54(1):1–27.

Abraham, D. J., Biró, P., and Manlove, D. F. (2005). "almost stable" matchings in the roommates problem. In *International Workshop on Approximation and Online Algorithms*, pages 1–14. Springer.

Agatz, N., Erera, A., Savelsbergh, M., and Wang, X. (2010). Sustainable passenger transportation: Dynamic ride-sharing.

Agatz, N., Erera, A., Savelsbergh, M., and Wang, X. (2012). Optimization for dynamic ride-sharing: A review. *European Journal of Operational Research*, 223(2):295–303.

Agatz, N., Erera, A. L., Savelsbergh, M. W., and Wang, X. (2011). Dynamic ride-sharing: A simulation study in metro atlanta. *Procedia-Social and Behavioral Sciences*, 17:532–550.

Ahmadian, S., Hosseinzadeh, H., and Sanità, L. (2018). Stabilizing network bargaining games by blocking players. *Mathematical Programming*, 172(1-2):249–275.

Ahuja, R. K., Magnanti, T. L., and Orlin, J. B. (1988). Network flows.

Alonso, M., Amaris, H., Germain, J. G., and Galan, J. M. (2014). Optimal charging scheduling of electric vehicles in smart grids by heuristic algorithms. *Energies*, 7(4):2449–2475.

Alonso-Mora, J., Samaranayake, S., Wallar, A., Frazzoli, E., and Rus, D. (2017). On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment. *Proceedings of the National Academy of Sciences*, 114(3):462–467.

Amey, A. (2011). A proposed methodology for estimating rideshare viability within an organization, applied to the mit community. In *TRB Annual Meeting Procediings*, pages 1–16.

Amey, A., Attanucci, J., and Mishalani, R. (2011). Real-time ridesharing: opportunities and challenges in using mobile phone technology to improve rideshare services. *Transportation Research Record*, 2217(1):103–110.

Baldacci, R., Maniezzo, V., and Mingozzi, A. (2004). An exact method for the car pooling problem based on lagrangean column generation. *Operations Research*, 52(3):422–439.

Becker, H., Balac, M., Ciari, F., and Axhausen, K. W. (2020). Assessing the welfare impacts of shared mobility and mobility as a service (maas). *Transportation Research Part A: Policy and Practice*, 131:228–243.

Benjaafar, S., Bernhard, H., and Courcoubetis, C. (2017). Drivers, riders and service providers: The impact of the sharing economy on mobility. *Available at SSRN 3035478*.

Benjaafar, S. and Hu, M. (2020). Operations management in the age of the sharing economy: what is old and what is new? *Manufacturing & Service Operations Management*, 22(1):93–101.

Bent, R. W. and Van Hentenryck, P. (2004). Scenario-based planning for partially dynamic vehicle routing with stochastic customers. *Operations Research*, 52(6):977–987.

Berger, A., Bonifaci, V., Grandoni, F., and Schäfer, G. (2011). Budgeted matching and budgeted matroid intersection via the gasoline puzzle. *Mathematical Programming*, 128(1):355–372.

Bian, Z. and Liu, X. (2019). Mechanism design for first-mile ridesharing based on personalized requirements part i: Theoretical analysis in generalized scenarios. *Transportation Research Part B: Methodological*, 120:147–171.

Biró, P., Bomhoff, M., Golovach, P. A., Kern, W., and Paulusma, D. (2012). Solutions for the stable roommates problem with payments. In *International Workshop on Graph-Theoretic Concepts in Computer Science*, pages 69–80. Springer.

Biró, P., Bomhoff, M., Golovach, P. A., Kern, W., and Paulusma, D. (2014). Solutions for the stable roommates problem with payments. *Theoretical computer science*, 540:53–61.

Biró, P., Inarra, E., and Molis, E. (2016). A new solution concept for the roommate problem: Q-stable matchings. *Mathematical Social Sciences*, 79:74–82.

Bock, A., Chandrasekaran, K., Könemann, J., Peis, B., and Sanità, L. (2015). Finding small stabilizers for unstable graphs. *Mathematical Programming*, 154(1-2):173–196.

Börgers, T. and Krahmer, D. (2015). *An introduction to the theory of mechanism design*. Oxford University Press, USA.

Boyacı, B., Zografos, K. G., and Geroliminis, N. (2015). An optimization framework for the development of efficient one-way car-sharing systems. *European Journal of Operational Research*, 240(3):718–733.

Braverman, A., Dai, J. G., Liu, X., and Ying, L. (2019). Empty-car routing in ridesharing systems. *Operations Research*, 67(5):1437–1452.

Brenner, U. (2008). A faster polynomial algorithm for the unbalanced hitchcock transportation problem. *Operations Research Letters*, 36(4):408–413.

Brownstone, D. and Golob, T. F. (1992). The effectiveness of ridesharing incentives: Discrete-choice models of commuting in southern california. *Regional Science and Urban Economics*, 22(1):5–24.

Brunetta, L., Conforti, M., and Rinaldi, G. (1997). A branch-and-cut algorithm for the equicut problem. *Mathematical Programming*, 78(2):243–263.

Buluç, A., Meyerhenke, H., Safro, I., Sanders, P., and Schulz, C. (2016). Recent advances in graph partitioning. In *Algorithm Engineering*, pages 117–158. Springer.

Byrka, J., Rybicki, B., and Uniyal, S. (2016). An approximation algorithm for uniform capacitated k-median problem with $1 + \epsilon$ capacity violation. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 262–274. Springer.

Calvo, R. W., de Luigi, F., Haastrup, P., and Maniezzo, V. (2004). A distributed geographic information system for the daily car pooling problem. *Computers & Operations Research*, 31(13):2263–2278.

Carlisle, M. C. and Lloyd, E. L. (1995). On the k-coloring of intervals. *Discrete Applied Mathematics*, 59(3):225–235.

Chan, N. D. and Shaheen, S. A. (2012). Ridesharing in north america: Past, present, and future. *Transport Reviews*, 32(1):93–112.

Chandrasekaran, K. (2017). Graph stabilization: A survey. In *Combinatorial Optimization and Graph Algorithms*, pages 21–41. Springer.

Chandrasekaran, K., Gottschalk, C., Könemann, J., Peis, B., Schmand, D., and Wierz, A. (2019). Additive stabilizers for unstable graphs. *Discrete Optimization*, 31:56–78.

Chang, J., Yu, M., Shen, S., and Xu, M. (2017). Location design and relocation of a mixed car-sharing fleet with a co2 emission constraint. *Service Science*, 9(3):205–218.

Chau, S. C.-K., Shen, S., and Zhou, Y. (2020). Decentralized ride-sharing and vehicle-pooling based on fair cost-sharing mechanisms. *IEEE Transactions on Intelligent Transportation Systems*.

Chen, W., Mes, M., Schutten, M., and Quint, J. (2019). A ride-sharing problem with meeting points and return restrictions. *Transportation science*, 53(2):401–426.

Chen, X. M., Zahiri, M., and Zhang, S. (2017). Understanding ridesplitting behavior of on-demand ride services: An ensemble learning approach. *Transportation Research Part C: Emerging Technologies*, 76:51–70.

Chou, S.-K., Jiau, M.-K., and Huang, S.-C. (2016). Stochastic set-based particle swarm optimization based on local exploration for solving the carpool service problem. *IEEE transactions on cybernetics*, 46(8):1771–1783.

Christofides, N., Mingozzi, A., and Toth, P. (1981). Exact algorithms for the vehicle routing problem, based on spanning tree and shortest path relaxations. *Mathematical programming*, 20(1):255–282.

Clarke, E. H. (1971). Multipart pricing of public goods. *Public choice*, pages 17–33.

Clewlow, R. R. and Mishra, G. S. (2017). Disruptive transportation: The adoption, utilization, and impacts of ride-hailing in the united states.

Cordeau, J.-F. (2006). A branch-and-cut algorithm for the dial-a-ride problem. *Operations Research*, 54(3):573–586.

Cordeau, J.-F. and Laporte, G. (2007). The dial-a-ride problem: models and algorithms. *Annals of operations research*, 153(1):29–46.

Correia, G. H. D. A., Jorge, D. R., and Antunes, D. M. (2014). The added value of accounting for users' flexibility and information on the potential of a station-based one-way car-sharing system: An application in lisbon, portugal. *Journal of Intelligent Transportation Systems*, 18(3):299–308.

Coslovich, L., Pesenti, R., and Ukovich, W. (2006). A two-phase insertion technique of unexpected customers for a dynamic dial-a-ride problem. *European Journal of Operational Research*, 175(3):1605–1615.

Cui, S., Li, K., Yang, L., and Wang, J. (2019). Slugging: Casual carpooling for urban transit. *Available at SSRN 3441901*.

Deakin, E., Frick, K. T., and Shively, K. M. (2010). Markets for dynamic ridesharing? case of berkeley, california. *Transportation Research Record*, 2187(1):131–137.

Desaulniers, G., Errico, F., Irnich, S., and Schneider, M. (2016). Exact algorithms for electric vehicle-routing problems with time windows. *Operations Research*, 64(6):1388–1405.

Di, X., Ma, R., Liu, H. X., and Ban, X. J. (2018). A link-node reformulation of ridesharing user equilibrium with network design. *Transportation Research Part B: Methodological*, 112:230–255.

Di Febbraro, A., Gattorna, E., and Sacco, N. (2013). Optimization of dynamic ridesharing systems. *Transportation Research Record*, 2359(1):44–50.

Donath, W. and Hoffman, A. (1972). Algorithms for partitioning of graphs and computer logic based on eigenvectors of connections matrices. *IBM Technical Disclosure Bulletin*, 15.

Dong, Y., Wang, S., Li, L., and Zhang, Z. (2018). An empirical study on travel patterns of internet based ride-sharing. *Transportation research part C: emerging technologies*, 86:1–22.

Dütting, P., Talgam-Cohen, I., and Roughgarden, T. (2017). Modularity and greed in double auctions. *Games and Economic Behavior*, 105:59–83.

Edmonds, J. (1965). Paths, trees, and flowers. *Canadian Journal of mathematics*, 17(3):449–467.

Edmonds, J. and Karp, R. M. (1972). Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the ACM (JACM)*, 19(2):248–264.

Eriksson, K. and Karlander, J. (2001). Stable outcomes of the roommate game with transferable utility. *International Journal of Game Theory*, 29(4):555–569.

Ersal, T., Kolmanovsky, I., Masoud, N., Ozay, N., Scruggs, J., Vasudevan, R., and Orosz, G. (2020). Connected and automated road vehicles: state of the art and future challenges. *Vehicle System Dynamics*, 58(5):672–704.

Fang, Z., Huang, L., and Wierman, A. (2020). Loyalty programs in the sharing economy: Optimality and competition. *Performance Evaluation*, page 102105.

Fazeli, S. S., Venkatachalam, S., Chinnam, R. B., and Murat, A. (2020a). Two-stage stochastic choice modeling approach for electric vehicle charging station network design in urban communities. *IEEE Transactions on Intelligent Transportation Systems*, 22(5):3038–3053.

Fazeli, S. S., Venkatachalam, S., and Smereka, J. M. (2020b). Efficient algorithms for autonomous electric vehicles' min-max routing problem. *arXiv preprint arXiv:2008.03333*.

Feng, X., Chen, Y., Zhang, J., Zhang, Q., and Li, B. (2012). Tahes: A truthful double auction mechanism for heterogeneous spectrums. *IEEE Transactions on Wireless Communications*, 11(11):4038–4047.

Fiduccia, C. M. and Mattheyses, R. M. (1982). A linear-time heuristic for improving network partitions. In *Proceedings of the 19th design automation conference*, pages 175–181. IEEE Press.

Fiedler, M. (1975). A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory. *Czechoslovak Mathematical Journal*, 25(4):619–633.

Fisher, M. L. (1981). The lagrangian relaxation method for solving integer programming problems. *Management science*, 27(1):1–18.

Fredman, M. L. and Tarjan, R. E. (1987). Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of the ACM (JACM)*, 34(3):596–615.

Friedman, J., Hastie, T., and Tibshirani, R. (2001). *The elements of statistical learning*, volume 1. Springer series in statistics New York.

Fu, L. (2002). Scheduling dial-a-ride paratransit under time-varying, stochastic congestion. *Transportation Research Part B: Methodological*, 36(6):485–506.

Furuhata, M., Dessouky, M., Ordóñez, F., Brunet, M.-E., Wang, X., and Koenig, S. (2013). Ridesharing: The state-of-the-art and future directions. *Transportation Research Part B: Methodological*, 57:28–46.

Gabow, H. N. (1990). Data structures for weighted matching and nearest common ancestors with linking. In *Proceedings of the first annual ACM-SIAM symposium on Discrete algorithms*, pages 434–443. Society for Industrial and Applied Mathematics.

Gale, D. and Shapley, L. S. (1962). College admissions and the stability of marriage. *The American Mathematical Monthly*, 69(1):9–15.

Galil, Z., Micali, S., and Gabow, H. (1986). An o(ev\logv) algorithm for finding a maximal weighted matching in general graphs. *SIAM Journal on Computing*, 15(1):120–130.

Gallo, G. and Sodini, C. (1978). Extreme points and adjacency relationship in the flow polytope. *Calcolo*, 15(3):277–288.

Ghilas, V., Demir, E., and Van Woensel, T. (2016). A scenario-based planning for the pickup and delivery problem with time windows, scheduled lines and stochastic demands. *Transportation Research Part B: Methodological*, 91:34–51.

Groves, T. (1973). Incentives in teams. *Econometrica: Journal of the Econometric Society*, pages 617–631.

Häme, L. (2011). An adaptive insertion algorithm for the single-vehicle dial-a-ride problem with narrow time windows. *European Journal of Operational Research*, 209(1):11–22.

Hasan, M. H., Van Hentenryck, P., and Legrain, A. (2020). The commute trip-sharing problem. *Transportation Science*, 54(6):1640–1675.

Hawkins, A. J. (2019). Uber and lyft finally admit they're making traffic congestion worse in cities - the verge. https://www.theverge.com/2019/8/6/20756945/uber-lyft-tnc-vmt-traffic-congestion-study-fehr-peers. (Accessed on 07/26/2020).

He, L., Mak, H.-Y., Rong, Y., and Shen, Z.-J. M. (2017). Service region design for urban electric vehicle sharing systems. *Manufacturing & Service Operations Management*, 19(2):309–327.

Healy, P. and Moll, R. (1995). A new extension of local search applied to the dial-a-ride problem. *European Journal of Operational Research*, 83(1):83–104.

Herbawi, W. and Weber, M. (2011). Evolutionary multiobjective route planning in dynamic multi-hop ridesharing. In *European conference on evolutionary computation in combinatorial optimization*, pages 84–95. Springer.

Ho, S. C., Szeto, W., Kuo, Y.-H., Leung, J. M., Petering, M., and Tou, T. W. (2018). A survey of dial-a-ride problems: Literature review and recent developments. *Transportation Research Part B: Methodological*, 111:395–421.

Holzhauser, M., Krumke, S. O., and Thielen, C. (2016). Budget-constrained minimum cost flows. *Journal of Combinatorial Optimization*, 31(4):1720–1745.

Hosni, H., Naoum-Sawaya, J., and Artail, H. (2014). The shared-taxi problem: Formulation and solution methods. *Transportation Research Part B: Methodological*, 70:303–318.

Hyafil, L. and Rivest, R. L. (1973). *Graph partitioning and constructing optimal decision trees are polynomial complete problems*. IRIA. Laboratoire de Recherche en Informatique et Automatique.

Ichoua, S., Gendreau, M., and Potvin, J.-Y. (2006). Exploiting knowledge about future demands for real-time vehicle dispatching. *Transportation Science*, 40(2):211–225.

Iqbal, M. (2021). Lyft revenue and usage statistics (2020) - business of apps. https://www.businessofapps.com/data/lyft-statistics/. (Accessed on 06/19/2021).

Irving, R. W. (1985). An efficient algorithm for the “stable roommates” problem. *Journal of Algorithms*, 6(4):577–595.

Ito, T., Kakimura, N., Kamiyama, N., Kobayashi, Y., and Okamoto, Y. (2017). Efficient stabilization of cooperative matching games. *Theoretical Computer Science*, 677:69–82.

Jacob, J. and Roet-Green, R. (2018). Ride solo or pool: Designing price-service menus for a ride-sharing platform.

Jafari, E., Rambha, T., Khani, A., and Boyles, S. D. (2016). The for-profit dial-a-ride problem on dynamic networks. In *the 95th Annual Meeting of the Transportation Research Board, Washington, DC (No. 16-4686)*.

Jaw, J.-J., Odoni, A. R., Psaraftis, H. N., and Wilson, N. H. (1986). A heuristic algorithm for the multi-vehicle advance request dial-a-ride problem with time windows. *Transportation Research Part B: Methodological*, 20(3):243–257.

Johnson, D. B. (1975). Finding all the elementary circuits of a directed graph. *SIAM Journal on Computing*, 4(1):77–84.

Jung, J., Jayakrishnan, R., and Park, J. Y. (2016). Dynamic shared-taxi dispatch algorithm with hybrid-simulated annealing. *Computer-Aided Civil and Infrastructure Engineering*, 31(4):275–291.

Karisch, S. E., Rendl, F., and Clausen, J. (2000). Solving graph bisection problems with semidefinite programming. *INFORMS Journal on Computing*, 12(3):177–191.

Kelly, K. L. (2007). Casual carpooling-enhanced. *Journal of Public Transportation*, 10(4):6.

Kernighan, B. W. and Lin, S. (1970). An efficient heuristic procedure for partitioning graphs. *The Bell system technical journal*, 49(2):291–307.

Kleiner, A., Nebel, B., and Ziparo, V. A. (2011). A mechanism for dynamic ride sharing based on parallel auctions. In *IJCAI*, volume 11, pages 266–272.

Knuth, D. E. (1997). *Stable marriage and its relation to other combinatorial problems: An introduction to the mathematical analysis of algorithms*, volume 10. American Mathematical Soc.

Laporte, G. and Nobert, Y. (1987). Exact algorithms for the vehicle routing problem. In *North-Holland Mathematics Studies*, volume 132, pages 147–184. Elsevier.

Lee, A. and Savelsbergh, M. (2015). Dynamic ridesharing: Is there a role for dedicated drivers? *Transportation Research Part B: Methodological*, 81:483–497.

Li, D., Antoniou, C., Jiang, H., Xie, Q., Shen, W., and Han, W. (2019). The value of prepositioning in smartphone-based vanpool services under stochastic requests and time-dependent travel times. *Transportation Research Record*, 2673(2):26–37.

Li, R., Liu, Z., and Zhang, R. (2018). Studying the benefits of carpooling in an urban area using automatic vehicle identification data. *Transportation Research Part C: Emerging Technologies*, 93:367–380.

Li, R., Nie, Y., and Liu, X. (2020). Pricing carpool rides based on schedule displacement. *Transportation Science*, 54(4):1134–1152.

Li, S. and Mi, C. C. (2014). Wireless power transfer for electric vehicle applications. *IEEE journal of emerging and selected topics in power electronics*, 3(1):4–17.

Lin, S. and Kernighan, B. W. (1973). An effective heuristic algorithm for the traveling-salesman problem. *Operations research*, 21(2):498–516.

Lloret-Batlle, R., Masoud, N., and Nam, D. (2017a). P2p ridesharing with ride-back on hov lanes: Towards a practical alternative mode for daily commuting.

Lloret-Batlle, R., Masoud, N., and Nam, D. (2017b). Peer-to-peer ridesharing with ride-back on high-occupancy-vehicle lanes: Toward a practical alternative mode for daily commuting. *Transportation Research Record: Journal of the Transportation Research Board*, (2668):21–28.

Lokhandwala, M. and Cai, H. (2018). Dynamic ride sharing using traditional taxis and shared autonomous taxis: A case study of nyc. *Transportation Research Part C: Emerging Technologies*, 97:45–60.

Long, J., Tan, W., Szeto, W., and Li, Y. (2018). Ride-sharing with travel time uncertainty. *Transportation Research Part B: Methodological*, 118:143–171.

Lowalekar, M., Varakantham, P., and Jaillet, P. (2018). Online spatio-temporal matching in stochastic and dynamic domains. *Artificial Intelligence*, 261:71–112.

Lu, M., Chen, Z., and Shen, S. (2018). Optimizing the profitability and quality of service in carshare systems under demand uncertainty. *Manufacturing & Service Operations Management*, 20(2):162–180.

Luo, Q., Saigal, R., Chen, Z., and Yin, Y. (2019). Accelerating the adoption of automated vehicles by subsidies: A dynamic games approach. *Transportation Research Part B: Methodological*, 129:226–243.

Ma, S., Zheng, Y., and Wolfson, O. (2013). T-share: A large-scale dynamic taxi ridesharing service. In *2013 IEEE 29th International Conference on Data Engineering (ICDE)*, pages 410–421. IEEE.

Ma, S., Zheng, Y., and Wolfson, O. (2014). Real-time city-scale taxi ridesharing. *IEEE Transactions on Knowledge and Data Engineering*, 27(7):1782–1795.

Maalouf, M., MacKenzie, C. A., Radakrishnan, S., and Court, M. (2014). A new fuzzy logic approach to capacitated dynamic dial-a-ride problem. *Fuzzy Sets and Systems*, 255:30–40.

Madsen, O. B., Ravn, H. F., and Rygaard, J. M. (1995). A heuristic algorithm for a dial-a-ride problem with time windows, multiple capacities, and multiple objectives. *Annals of operations Research*, 60(1):193–208.

Martin, E. and Shaheen, S. (2011). The impact of carsharing on household vehicle ownership.

Martin, E., Shaheen, S. A., and Lidicker, J. (2010). Impact of carsharing on household vehicle holdings: Results from north american shared-use vehicle survey. *Transportation Research Record*, 2143(1):150–158.

Masoud, N. and Jayakrishnan, R. (2016). Formulations for optimal shared ownership and use of autonomous or driverless vehicles. In *Proceedings of the Transportation Research Board 95th Annual Meeting*, pages 1–17.

Masoud, N. and Jayakrishnan, R. (2017a). Autonomous or driver-less vehicles: Implementation strategies and operational concerns. *Transportation research part E: logistics and transportation review*, 108:179–194.

Masoud, N. and Jayakrishnan, R. (2017b). A decomposition algorithm to solve the multi-hop peer-to-peer ride-matching problem. *Transportation Research Part B: Methodological*, 99:1–29.

Masoud, N. and Jayakrishnan, R. (2017c). A real-time algorithm to solve the peer-to-peer ride-matching problem in a flexible ridesharing system. *Transportation Research part B: Methodological*, 106:218–236.

Masoud, N., Lloret-Batlle, R., and Jayakrishnan, R. (2017a). Using bilateral trading to increase ridership and user permanence in ridesharing systems. *Transportation Research Part E: Logistics and Transportation Review*, 102:60–77.

Masoud, N., Nam, D., Yu, J., and Jayakrishnan, R. (2017b). Promoting peer-to-peer ridesharing services as transit system feeders. *Transportation Research Record: Journal of the Transportation Research Board*, 2650:74–83.

Masoud, S., Son, Y. J., Masoud, N., and Jayakrishnan, J. (2019). Impact of traffic conditions and carpool lane availability on peer to peer ridesharing demand. *arXiv preprint arXiv:1912.08931*.

McAfee, R. P. (1992). A dominant strategy double auction. *Journal of economic Theory*, 56(2):434–450.

Megiddo, N. (1978). Combinatorial optimization with rational objective functions. In *Proceedings of the tenth annual ACM symposium on Theory of computing*, pages 1–12.

Milgrom, P. and Segal, I. (2014). Deferred-acceptance auctions and radio spectrum reallocation. In *Proceedings of the fifteenth ACM conference on Economics and computation*, pages 185–186.

MirHassani, S. and Ebrazi, R. (2013). A flexible reformulation of the refueling station location problem. *Transportation Science*, 47(4):617–628.

Mitrović-Minić, S., Krishnamurti, R., and Laporte, G. (2004). Double-horizon based heuristics for the dynamic pickup and delivery problem with time windows. *Transportation Research Part B: Methodological*, 38(8):669–685.

Mote, J. E. and Whitestone, Y. (2011). The social context of informal commuting: Slugs, strangers and structuration. *Transportation Research Part A: Policy and Practice*, 45(4):258–268.

Moug, K., Jia, H., and Shen, S. (N.D.). A shared mobility based framework for evacuation planning and operations under demand uncertainty. http://www.optimization-online.org/DB_HTML/2020/08/7963.html.

Myerson, R. B. and Satterthwaite, M. A. (1983). Efficient mechanisms for bilateral trading. *Journal of economic theory*, 29(2):265–281.

Najmi, A., Rey, D., and Rashidi, T. H. (2017). Novel dynamic formulations for real-time ride-sharing systems. *Transportation research part E: logistics and transportation review*, 108:122–140.

Nam, D., Yang, D., An, S., Yu, J. G., Jayakrishnan, R., and Masoud, N. (2018). Designing a transit-feeder system using multiple sustainable modes: Peer-to-peer (p2p) ridesharing, bike sharing, and walking. *Transportation Research Record*, page 0361198118799031.

Nisan, N., Roughgarden, T., Tardos, E., and Vazirani, V. V. (2007). *Algorithmic Game Theory*. Cambridge University Press, USA.

Nourinejad, M. and Roorda, M. J. (2015). Carsharing operations policies: a comparison between one-way and two-way systems. *Transportation*, 42(3):497–518.

Nourinejad, M. and Roorda, M. J. (2016). Agent based model for dynamic ridesharing. *Transportation Research Part C: Emerging Technologies*, 64:117–132.

Parvez, M., Mekhilef, S., Tan, N. M., and Akagi, H. (2014). Model predictive control of a bidirectional ac-dc converter for v2g and g2v applications in electric vehicle battery charger. In *2014 IEEE Transportation Electrification Conference and Expo (ITEC)*, pages 1–6. IEEE.

Pelzer, D., Xiao, J., Zehe, D., Lees, M. H., Knoll, A. C., and Aydt, H. (2015). A partition-based match making algorithm for dynamic ridesharing. *IEEE Transactions on Intelligent Transportation Systems*, 16(5):2587–2598.

Peng, Z., Shan, W., Jia, P., Yu, B., Jiang, Y., and Yao, B. (2020). Stable ride-sharing matching for the commuters with payment design. *Transportation*, 47(1):1–21.

Puentes, R. (2017). What new census data reveal about american commuting patterns. https://www.usnews.com/opinion/economic-intelligence/articles/2017-09-18/what-new-census-data-reveal-about-american-commuting-patterns. (Accessed on 06/08/2021).

Qi, W. and Shen, Z.-J. M. (2019). A smart-city scope of operations management. *Production and Operations Management*, 28(2):393–406.

Qian, X., Zhang, W., Ukkusuri, S. V., and Yang, C. (2017). Optimal assignment and incentive design in the taxi group ride problem. *Transportation Research Part B: Methodological*, 103:208–226.

Ramezani, M. and Nourinejad, M. (2017). Dynamic modeling and control of taxi services in large-scale urban networks: A macroscopic approach. *Transportation research procedia*, 23:41–60.

Rasulkhani, S. and Chow, J. Y. (2019). Route-cost-assignment with joint user and operator behavior as a many-to-one stable matching assignment game. *Transportation Research Part B: Methodological*, 124:60–81.

Regue, R., Masoud, N., and Recker, W. (2016). Car2work: A shared mobility concept to connect commuters with workplaces. *Transportation Research Record: Journal of the Transportation Research Board*, 2542:102–110.

Ritzinger, U., Puchinger, J., and Hartl, R. F. (2016). Dynamic programming based metaheuristics for the dial-a-ride problem. *Annals of Operations Research*, 236(2):341–358.

Roth, A. E. and Sotomayor, M. (1992). Two-sided matching. *Handbook of game theory with economic applications*, 1:485–541.

Roth, A. E. and Sotomayor, M. (1996). Stable outcomes in discrete and continuous models of two-sided matching: A unified treatment. *Brazilian Review of Econometrics*, 16(2):1–24.

Sanchez-Martin, P., Sanchez, G., and Morales-España, G. (2012). Direct load control decision model for aggregated ev charging points. *IEEE Transactions on Power Systems*, 27(3):1577–1584.

Santi, P., Resta, G., Szell, M., Sobolevsky, S., Strogatz, S. H., and Ratti, C. (2014). Quantifying the benefits of vehicle pooling with shareability networks. *Proceedings of the National Academy of Sciences*, 111(37):13290–13294.

Savelsbergh, M. W. and Sol, M. (1995). The general pickup and delivery problem. *Transportation science*, 29(1):17–29.

Schilde, M., Doerner, K. F., and Hartl, R. F. (2011). Metaheuristics for the dynamic stochastic dial-a-ride problem with expected return transports. *Computers & operations research*, 38(12):1719–1730.

Schneider, B. (2018). New study of global traffic reveals that traffic is bad - bloomberg. https://www.bloomberg.com/news/articles/2018-02-07/new-study-of-global-traffic-reveals-that-traffic-is-bad. (Accessed on 07/26/2020).

Schneider, M., Stenger, A., and Goeke, D. (2014). The electric vehicle-routing problem with time windows and recharging stations. *Transportation Science*, 48(4):500–520.

Segal-Halevi, E., Hassidim, A., and Aumann, Y. (2016). Sbba: a strongly-budget-balanced double-auction mechanism. In *International Symposium on Algorithmic Game Theory*, pages 260–272. Springer.

Segal-Halevi, E., Hassidim, A., and Aumann, Y. (2018). Muda: A truthful multi-unit double-auction mechanism. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Shaheen, S. and Chan, N. (2016). Mobility and the sharing economy: Potential to facilitate the first-and last-mile public transit connections. *Built Environment*, 42(4):573–588.

Shaheen, S., Cohen, A., and Jaffee, M. (2018). Innovative mobility: Carsharing outlook.

Shaheen, S. A., Chan, N. D., and Gaynor, T. (2016). Casual carpooling in the san francisco bay area: Understanding user characteristics, behaviors, and motivations. *Transport Policy*, 51:165–173.

Shaheen, S. A. and Cohen, A. P. (2007). Growth in worldwide carsharing: An international comparison. *Transportation Research Record*, 1992(1):81–89.

Shaheen, S. A., Guzman, S., and Zhang, H. (2010). Bikesharing in europe, the americas, and asia: past, present, and future. *Transportation Research Record*, 2143(1):159–167.

Shapley, L. S. and Shubik, M. (1971). The assignment game i: The core. *International Journal of game theory*, 1(1):111–130.

Shen, B., Huang, Y., and Zhao, Y. (2016a). Dynamic ridesharing. *Sigspatial Special*, 7(3):3–10.

Shen, W., Lopes, C. V., and Crandall, J. W. (2016b). An online mechanism for ridesharing in autonomous mobility-on-demand systems. *arXiv preprint arXiv:1603.02208*.

Shoham, Y. and Leyton-Brown, K. (2008). *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*. Cambridge University Press.

Simon, H. D. (1991). Partitioning of unstructured problems for parallel processing. *Computing systems in engineering*, 2(2):135–148.

Simonetto, A., Monteil, J., and Gambella, C. (2019). Real-time city-scale ridesharing via linear assignment problems. *Transportation Research Part C: Emerging Technologies*, 101:208–232.

Stiglic, M., Agatz, N., Savelsbergh, M., and Gradisar, M. (2015). The benefits of meeting points in ride-sharing systems. *Transportation Research Part B: Methodological*, 82:36–53.

Stiglic, M., Agatz, N., Savelsbergh, M., and Gradisar, M. (2016). Making dynamic ride-sharing work: The impact of driver and rider flexibility. *Transportation Research Part E: Logistics and Transportation Review*, 91:190–207.

Ströhle, P., Flath, C. M., and Gärttner, J. (2019). Leveraging customer flexibility for car-sharing fleet optimization. *Transportation Science*, 53(1):42–61.

Stüdli, S., Crisostomi, E., Middleton, R., and Shorten, R. (2014). Optimal real-time distributed v2g and g2v management of electric vehicles. *International Journal of Control*, 87(6):1153–1162.

Su, Q. and Zhou, L. (2012). Parking management, financial subsidies to alternatives to drive alone and commute mode choices in seattle. *Regional Science and Urban Economics*, 42(1-2):88–97.

Sweda, T. M., Dolinskaya, I. S., and Klabjan, D. (2017). Optimal recharging policies for electric vehicles. *Transportation Science*, 51(2):457–479.

Tafreshian, A., Abdolmaleki, M., Masoud, N., and Wang, H. (2021). Proactive shuttle dispatching in large-scale dynamic dial-a-ride systems. *Transportation Research Part B: Methodological*, page to be appeared.

Tafreshian, A. and Masoud, N. (2020a). Trip-based graph partitioning in dynamic ridesharing. *Transportation Research Part C: Emerging Technologies*, 114:532–553.

Tafreshian, A. and Masoud, N. (2020b). Using subsidies to stabilize peer-to-peer ridesharing markets with role assignment. *Transportation Research Part C: Emerging Technologies*, 120:102770.

Tafreshian, A. and Masoud, N. (N.D.a). A traveler incentive program for promoting community-based ridesharing. *Under second round of review*.

Tafreshian, A. and Masoud, N. (N.D.b). A truthful subsidy scheme for a peer-to-peer ridesharingmarket with incomplete information. *To be submitted*.

Tafreshian, A., Masoud, N., and Yin, Y. (2020). Frontiers in service science: Ride matching for peer-to-peer ride sharing: A review and future directions. *Service Science*, 12(2-3):44–60.

Tamannaei, M. and Irandoost, I. (2019). Carpooling problem: A new mathematical model, branch-and-bound, and heuristic beam search algorithm. *Journal of Intelligent Transportation Systems*, 23(3):203–215.

Tarjan, R. E. (1983). *Data structures and network algorithms*, volume 44. Siam.

UN Department of Economics and Social Affairs (2019). World population prospects 2019 highlights. https://population.un.org/wpp/Publications/Files/WPP2019_Highlights.pdf. (Accessed on 02/07/2020).

Thaithatkul, P., Seo, T., Kusakabe, T., and Asakura, Y. (2015). A passengers matching problem in ridesharing systems by considering user preference. *Journal of the Eastern Asia Society for Transportation Studies*, 11:1416–1432.

Thaithatkul, P., Seo, T., Kusakabe, T., and Asakura, Y. (2017). Simulation approach for investigating dynamics of passenger matching problem in smart ridesharing system. *Transportation Research Procedia*, 21:29–41.

Ulmer, M. W., Goodson, J. C., Mattfeld, D. C., and Thomas, B. W. (2017). Route-based markov decision processes for dynamic vehicle routing problems. Technical report, Technical report, Braunschweig.

Vate, J. H. V. (1989). Linear programming brings marital bliss. *Operations Research Letters*, 8(3):147–153.

Vickrey, W. (1961). Counterspeculation, auctions, and competitive sealed tenders. *The Journal of finance*, 16(1):8–37.

Wang, H. and Yang, H. (2019). Ridesourcing systems: A framework and review. *Transportation Research Part B: Methodological*, 129:122–155.

Wang, M., Ismail, M., Zhang, R., Shen, X., Serpedin, E., and Qaraqe, K. (2016a). Spatio-temporal coordinated v2v energy swapping strategy for mobile pevs. *IEEE Transactions on Smart Grid*, 9(3):1566–1579.

Wang, X., Agatz, N., and Erera, A. (2017). Stable matching for dynamic ride-sharing systems. *Transportation Science*.

Wang, X., Agatz, N., and Erera, A. (2018a). Stable matching for dynamic ride-sharing systems. *Transportation Science*, 52(4):850–867.

Wang, X., Dessouky, M., and Ordonez, F. (2016b). A pickup and delivery problem for ridesharing considering congestion. *Transportation letters*, 8(5):259–269.

Wang, X., Yang, H., and Zhu, D. (2018b). Driver-rider cost-sharing strategies and equilibria in a ridesharing program. *Transportation Science*, 52(4):868–881.

Wei, C., Wang, Y., Yan, X., and Shao, C. (2017). Look-ahead insertion policy for a shared-taxi system based on reinforcement learning. *IEEE Access*, 6:5716–5726.

Weikl, S. and Bogenberger, K. (2013). Relocation strategies and algorithms for free-floating car sharing systems. *IEEE Intelligent Transportation Systems Magazine*, 5(4):100–111.

West, D. B. et al. (1996). *Introduction to graph theory*, volume 2. Prentice hall Upper Saddle River, NJ.

Xia, J., Curtin, K. M., Li, W., and Zhao, Y. (2015). A new model for a carpool matching service. *PloS one*, 10(6).

Xiang, Z., Chu, C., and Chen, H. (2008). The study of a dynamic dial-a-ride problem under time-dependent and stochastic environments. *European Journal of Operational Research*, 185(2):534–551.

Xiong, C., Shahabi, M., Zhao, J., Yin, Y., Zhou, X., and Zhang, L. (2020). An integrated and personalized traveler information and incentive scheme for energy efficient mobility systems. *Transportation Research Part C: Emerging Technologies*, 113:57–73.

Xu, H., Pang, J.-S., Ordóñez, F., and Dessouky, M. (2015). Complementarity models for traffic equilibrium with ridesharing. *Transportation Research Part B: Methodological*, 81:161–182.

Xu, Z., Yin, Y., and Ye, J. (2020). On the supply curve of ride-hailing systems. *Transportation Research Part B: Methodological*, 132:29–43.

Yang, D., Fang, X., and Xue, G. (2011). Truthful auction for cooperative communications. In *Proceedings of the Twelfth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 1–10.

Yu, M., Nagarajan, V., and Shen, S. (2021). Improving column-generation for vehicle routing problems via random coloring and parallelization.

Yu, M. and Shen, S. (2020). An integrated car-and-ride sharing system for mobilizing heterogeneous travelers with application in underserved communities. *IISE Transactions*, 52(2):151–165.

Yu, X. and Shen, S. (2019). An integrated decomposition and approximate dynamic programming approach for on-demand ride pooling. *IEEE Transactions on Intelligent Transportation Systems*, 21(9):3811–3820.

Zha, L., Yin, Y., and Du, Y. (2018). Surge pricing and labor supply in the ride-sourcing market. *Transportation Research Part B: Methodological*, 117:708–722.

Zha, L., Yin, Y., and Yang, H. (2016). Economic analysis of ride-sourcing markets. *Transportation Research Part C: Emerging Technologies*, 71:249–266.

Zhang, J., Wen, D., and Zeng, S. (2015). A discounted trade reduction mechanism for dynamic ridesharing pricing. *IEEE Transactions on Intelligent Transportation Systems*, 17(6):1586–1595.

Zhang, Y., Lu, M., and Shen, S. (2021). On the values of vehicle-to-grid electricity selling in electric vehicle sharing. *Manufacturing & Service Operations Management*, 23(2):488–507.

Zhang, Z., Tafreshian, A., and Masoud, N. (2020). Modular transit: Using autonomy and modularity to improve performance in public transportation. *Transportation Research Part E: Logistics and Transportation Review*, 141:102033.

Zhao, D. and Chen, M. (2019). Ex-ante versus ex-post destination information model for on-demand service ride-sharing platform. *Annals of Operations Research*, 279(1-2):301–341.

Zhao, D., Ramchurn, S. D., and Jennings, N. R. (2015). Incentive design for ridesharing with uncertainty. *arXiv preprint arXiv:1505.01617*.

Zhao, D., Zhang, D., Gerding, E. H., Sakurai, Y., and Yokoo, M. (2014). Incentives in ridesharing with deficit control. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, pages 1021–1028. International Foundation for Autonomous Agents and Multiagent Systems.