OPTIMIZING RIDE MATCHES FOR DYNAMIC RIDE-SHARING SYSTEMS

A Thesis
Presented to
The Academic Faculty

by

Xing Wang

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Industrial and Systems Engineering

Georgia Institute of Technology
May 2013

OPTIMIZING RIDE MATCHES FOR DYNAMIC RIDE-SHARING SYSTEMS

Approved by:

Alan L. Erera, Advisor
School of Industrial and Systems Engineering
Georgia Institute of Technology

Martin W. P. Savelsbergh
School of Industrial and Systems Engineering
Georgia Institute of Technology

Niels Agatz
Rotterdam School of Management
Erasmus University

Ellis L. Johnson
School of Industrial and Systems Engineering
Georgia Institute of Technology

Joel S. Sokol
School of Industrial and Systems Engineering
Georgia Institute of Technology

Jorge A. Laval
School of Civil and Environmental Engineering
Georgia Institute of Technology

Date Approved: 27 February 2013

To my husband and our parents,

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my advisor Dr. Alan Erera, whose patience, support, and encouragement guided me through my research. He always builds my confidence when I am down. During my research journey, I would like to thank many other professors. I had the privilege to work with Dr. Ellis Johnson for my first year in ISYE and I really appreciate his financial support and his previous advice on my early research. I would like to thank Dr. Martin Savelsbergh who introduced me to this interesting ride-sharing research and his helpful guidance for my first year of research on my thesis topic. I would like to thank Dr. Niels Agatz for his advice and discussions. I learned team-working skills during my work with him. Dr. Joel Sokol's patience, advice, and kindness always provided me with great encouragement. I also want to thank Dr. Jorge Laval's suggestions for the completion of my thesis.

I especially want to thank my husband Xiaoyuan Lou, without his endless love and support, my PhD dream can not come true. I also want to thank our parents for their great support. It has been my great pleasure to make so many friends in ISYE. Specifically, I want to thank Yaxian Li, Luyi Gui, Qie He, Steve Tyber, Lei Wang, and Xuefeng Gao for all the discussions on coursework and research. I also want to thank my many other friends in ISYE who have made my leisure time outside of research colorful.

TABLE OF CONTENTS

DEDICATION . . . . . **iii**

ACKNOWLEDGEMENTS . . . . . **iv**

LIST OF TABLES . . . . . **viii**

LIST OF FIGURES . . . . . **ix**

SUMMARY . . . . . **x**

I INTRODUCTION **1**

II DYNAMIC RIDE-SHARING SYSTEMS AND MATCHING OPTIMIZATION **5**

2.1 Introduction 5

2.2 Problem Characteristics 6

2.2.1 Features of Dynamic Ride-sharing 6

2.2.2 The Ride-share Process 7

2.2.3 Ride-sharing System Objectives 8

2.2.4 Cost Considerations 11

2.2.5 System Versus User Benefits 13

2.3 Basic Ride-sharing Problems 14

2.3.1 Single Rider, Single Driver Arrangements 15

2.3.2 Single Driver, Multiple Rider Arrangements 17

2.3.3 Single Rider, Multiple Driver Arrangements 19

2.4 Dynamic Ride-sharing Problems 20

2.4.1 Arrival of Riders and Drivers 21

2.4.2 Anticipation of Future Requests 24

2.4.3 Deviations from Planned Trips 24

2.5 The Multi-modal Ride-sharing Problem 25

2.6 Conclusions 26

III SINGLE-RIDER, SINGLE-DRIVER RIDE-SHARE MATCHING OPTIMIZATION 30

3.1 Introduction 30

3.2 The Dynamic Ride-share Setting 31

3.3 Solving the Dynamic Ride-share Problem 34

3.3.1 Rolling Horizon Strategy 34

3.3.2 Solving the Ride-share Matching Optimization Problem . . . 36

3.3.3 Benchmarks 39

3.4 Numerical Experiments 40

3.4.1 Simulation Environment 41

3.4.2 Base Case Computational Results 43

3.4.3 The Advantages of Flexible Roles 50

3.4.4 Single Trip Ride-sharing 53

3.4.5 Fixing Ride-share Pairs on Round Trips 54

3.4.6 Varying the Participants' Flexibility 56

3.5 How to Achieve Critical Mass? 58

3.6 Conclusions 62

IV SINGLE-RIDER, SINGLE-DRIVER STABLE (AND NEARLY-STABLE) RIDE-SHARE MATCHING 63

4.1 Introduction 63

4.2 Stable Marriage Problem 64

4.3 Stable Matching Problem in Ride-Sharing 68

4.3.1 Basic Problem 69

4.3.2 Nearly-stable Problem 77

4.3.3 Greedy Algorithm 81

4.4 Simulation in System Dynamics with Unstable Participants 82

4.5 Conclusions 89

V MECHANISM DESIGN IN CHOICE ASSIGNMENT 91

5.1 Introduction 91

5.2 Problem Statement 92
5.3 Mathematical Model and Solution Approaches 93
5.3.1 Choice With Certainty 93
5.3.2 Choice With Uncertainty 96
5.4 Summary and Future Research 107
REFERENCES **109**

LIST OF TABLES

1 Ride-share Variants 14

2 Home-based Work Travel Information (ARC, 2008) 42

3 Base Case Solution Quality Comparison 44

4 Rolling Horizon Strategy Comparison 50

5 Ride-sharing with Flexible Roles 51

6 Maximize Savings versus Maximizing Matches 54

7 Fixed Ride-share Pairs 56

8 Participants' Time Flexibility and Cost Savings Threshold 57

9 Stable Matching Solution Quality Comparison 71

10 Relative Gaps Between $MWSM$ and MWM Solutions For Various Performance Metrics 72

11 Blocking Pairs in Max Weight Matching 73

12 Instability Gaps Between $MWSM$ and MWM Solutions under Different Time Flexibility 75

13 MWSM Degradation without Time Feasibility Constraints 76

14 Max Weight Nearly Stable Matching $MWNSM1$ Blocking Pairs Statistics 79

15 Max Weight Nearly Stable Matching $MWNSM2$ Blocking Pairs Statistics 81

16 Solution Quality under Different Choice Menu Sizes 96

17 Optimal Solutions for $P2$ 102

18 P2-SAA, $m = 6$ 104

19 Construction Methods, $m = 6$ 106

20 1 - exchange Local Search on Construction Methods, $m = 6$ 107

LIST OF FIGURES

1 Time Schedule Information in Ride-sharing 10

2 Riders (grey) and Drivers (white) Traveling from Origin (circle) to Destination (square) 13

3 Example of Single Rider, Single Driver Rideshare Arrangement 15

4 A Shared Trip between Driver d (squares) and Rider r (circles) 32

5 Original Trip Distances for Matched Participants 47

6 Success Rates for Riders (gray) and Drivers (black) by Original Trip Distance 48

7 Success Rate by Time of Day 49

8 Matching Results for Flexible Roles Scenarios by Original Trip Length: Matched as Rider (gray), Matched as Driver (black), Not Matched (white) 52

9 Ride-sharing System Sustainability for Various Diffusion Patterns . . 60

10 Sustainability of Ride-sharing Systems for Different Levels of Initial Participant Goodwill 61

11 An Example Instance with an Instability Gap in System Vehicle Miles Savings Close to 50% 72

12 Distribution of Feasible Matches in A Posteriori Ride-Share Matching Problems 74

13 Benefits from Forming a Blocking Pair 77

14 System Savings Dynamics with Accept/Reject Behavior Modeling . . 86

15 System Savings Dynamics in Accept/Reject Behavior Modeling with Full Backup List 87

16 System Savings Dynamics with Accept/Reject Behavior Modeling in Different Timing Variation 88

17 System Savings Comparison with Tolerance in Savings 89

SUMMARY

Ride-share systems, which aim to bring together travelers with similar itineraries and time schedules, may provide significant societal and environmental benefits by reducing the number of cars used for personal travel and improving the utilization of available seat capacity. Effective and efficient optimization technology that matches drivers and riders in real-time is one of the necessary components for a successful ride-share system.

The research conducted in this dissertation formally defines dynamic or real-time ride-sharing, identifies optimization problems for finding best sets of ride-share matches in a number of operational scenarios, develops approaches for solving ride-share optimization problems, and tests the concepts via a simulation study of work trips in the Atlanta metropolitan area.

In Chapter 2, we systematically outline the optimization challenges that arise when developing technology to support ride-sharing and survey the related operations research models in academic literature.

In Chapter 3, we develop optimization-based approaches for finding ride-share matches in a standard problem setting, with the goal of minimizing the total system-wide vehicle miles incurred by system users. To assess the merits of our methods we present a simulation study based on 2008 travel demand data from metropolitan Atlanta. The simulation results indicate that the use of sophisticated optimization methods instead of simple greedy matching rules substantially improves the performance of ride-sharing systems. Furthermore, even with relatively low participation rates, it appears that sustainable populations of dynamic ride-sharing participants

may be possible even in relatively sprawling urban areas with many employment centers.

In Chapter 4, we consider a more sophisticated ride-share setting where participants may be unlikely to accept ride-share matches if they are not stable. Generically, a set of matches between riders and drivers is defined as stable if no rider and driver, currently matched to others, would prefer to be matched together. This notion of stability is similar to that of the stable marriage problem. We develop notions of stable ride-share matching in a variety of settings, and develop approaches for finding stable (or nearly-stable) solutions. Computational results are used to compare system performance under various levels of matching stability. A system with unstable matching assignments is simulated over two months in which participants are likely to reject the system's assignment if a private arrangement between individuals could bring better benefits. The simulation results indicate that the total savings generated by a ride-sharing system deteriorate with unstable matching assignments and that enforcing stability constraints in matching models is beneficial.

In Chapter 5, we consider another set of more sophisticated ride-share matching settings where participants are not assumed to accept each match to which they are assigned. In such settings, it may be useful to present users with a menu of possible ride-share matches from which they can choose. We develop models and solution approaches to jointly present multiple options to participants based on a complete bipartite graph structure. This research could serve as a building block for future work on the dynamic ride-sharing problem.

CHAPTER I

INTRODUCTION

Finite oil supplies, rising gas prices, traffic congestion, and environmental concerns have recently increased the interest in services that allow people to use personal automobiles more wisely. The demand for ride-sharing services, which aim to bring together travelers with similar itineraries and time schedules, has increased sharply in recent years [61]. Ride-share providers across the globe are offering online notice boards for potential carpoolers, whether for daily commutes or for one-time trips to festivals, concerts, or sports events. Some online services, such as Nuride, provide incentives like restaurant coupons, gift certificates, or retail sales discounts to participants. Ride-sharing has generated much interest, and recent media coverage can be found in the Wall Street Journal [61], Newsweek [42], Business Week [68], ABC News [10], The NY Times [69], among many others.

Private car occupancy rates (the number of travelers per vehicle trip) are relatively low; average car occupancies in Europe range from 1.8 for leisure trips to 1.1 for commuters [1]. Similar occupancy rates are also found in the US [60]. The large demand for automobile transportation at peak-hours together with low occupancies leads to traffic congestion in many urban areas. The annual cost of congestion in the US in terms of lost hours and wasted fuel was estimated to be $78 billion in 2007 [63]. Private automobile usage is also the dominant transportation mode producing carbon dioxide emissions [31]. Vehicle emissions give rise to problems both on a local and global scale. Locally, the health effects of air pollution represent a serious problem in many of the most densely populated regions worldwide [14, 38]. Globally, carbon dioxide emissions are associated with climate change and global warming.

Effective usage of empty car seats by ride-sharing may represent an important opportunity to increase occupancy rates, and could substantially increase the efficiency of urban transportation systems, potentially reducing traffic congestion, fuel consumption, and pollution. Moreover, ride-sharing allows users to share car-related expenses, which can be substantial, especially since the price of oil has doubled over the past five years [2]. While ride-sharing is not a new idea, recent technological advances may increase its popularity, as we will explain. Certainly, ride-sharing must be easy, safe, flexible, efficient and economical before it will be widely adopted.

By dynamic ride-sharing, we refer to a system where an automated process provided by a ride-share provider matches up drivers and riders on very short notice or even en-route. Recent startups like Carticipate, EnergeticX, Avego, and Flinc offer dynamic ride-sharing applications that allow drivers with spare seats to connect to people wanting to share a ride. They provide applications that run on (location-aware) Internet-enabled mobile phones. To ease the fear of sharing a ride with a potential stranger, these services use reputation systems (see *e.g.*, PickupPal) or can be linked with social network tools like Facebook (see *e.g.*, GoLoco and Zimride).

The ability of a dynamic ride-share provider to successfully establish ride-shares on short notice depends on the characteristics of the environment in terms of participant geographic density, traffic patterns, and the available roadway and transit infrastructure. Hall and Qureshi [30] analyze the likelihood that a person will be successful in finding a ride-match, given a pool size of potential ride matches. Using a probabilistic analysis, they conclude that in theory ride-sharing is viable since a congested freeway corridor should offer sufficient potential ride-matches. The authors also observe that there are many obstacles, primarily in terms of communication, so that the chance of finding a ride match in practice may in fact be small. Fortunately, technological advances have greatly reduced this communication obstacle.

Although the enabling technology is available, ride-sharing success stories are still

in short supply. The development of algorithms for optimally matching drivers and riders in real-time plays an essential role in ride-sharing systems, and the operations research community has only recently started to address the related optimization challenges. This thesis will make contributions in modeling and designing algorithms for a number of relevant optimization problems for identifying ride-share matches for real-time ride-sharing systems:

In Chapter 2, we systematically outline the optimization challenges that arise when developing technology to support ride-sharing and survey the related operations research models in academic literature.

In Chapter 3, we develop optimization-based approaches for finding ride-share matches in a standard problem setting, with the goal of minimizing the total system-wide vehicle miles incurred by system users. To assess the merits of our methods we present a simulation study based on 2008 travel demand data from metropolitan Atlanta. The simulation results indicate that the use of sophisticated optimization methods instead of simple greedy matching rules substantially improves the performance of ride-sharing systems. Furthermore, even with relatively low participation rates, it appears that sustainable populations of dynamic ride-sharing participants may be possible even in relatively sprawling urban areas with many employment centers.

In Chapter 4, we consider a more sophisticated ride-share setting where participants may be unlikely to accept ride-share matches if they are not stable. Generically, a set of matches between riders and drivers is defined as stable if no rider and driver, currently matched to others, would prefer to be matched together. This notion of stability is similar to that of the stable marriage problem. We develop notions of stable ride-share matching in a variety of settings, and develop approaches for finding stable (or nearly-stable) solutions. Computational results are used to compare system performance under various levels of matching stability. A system with unstable

matching assignments is simulated over two months in which participants are likely to reject the system's assignment if a private arrangement between individuals could bring better benefits. The simulation results indicate that the total savings generated by a ride-sharing system deteriorate with unstable matching assignments and that enforcing stability constraints in matching models is beneficial.

In Chapter 5, we consider another set of more sophisticated ride-share matching settings where participants are not assumed to accept each match to which they are assigned. In such settings, it may be useful to present users with a menu of possible ride-share matches from which they can choose. We develop models and solution approaches to jointly present multiple options to participants based on a complete bipartite graph structure. This research could serve as a building block for future work on the dynamic ride-sharing problem.

CHAPTER II

DYNAMIC RIDE-SHARING SYSTEMS AND MATCHING OPTIMIZATION

2.1 Introduction

Ride-share systems, which aim to bring together travelers with similar itineraries and time schedules, may provide significant societal and environmental benefits by reducing the number of cars used for personal travel and improving the utilization of available seat capacity. Effective and efficient optimization technology that matches drivers and riders in real-time is one of the necessary components for a successful ride-share system. We systematically outline the optimization challenges that arise when developing technology to support ride-sharing and survey the related operations research models in academic literature.

The remainder of this chapter is structured as follows. In Section 2.2, we explain and characterize the dynamic ride-sharing concept and introduce several relevant planning issues that arise in this context. In Section 2.3, we present a more formal definition of the basic ride-sharing problem and its variants and survey the available literature. In Section 2.4 we discuss dynamic ride-sharing problems. In Section 2.5, we present the multi-modal version of the ride-sharing problem. Finally, in Section 2.6, we summarize our main insights and discuss directions for future research.

2.2 Problem Characteristics

2.2.1 Features of Dynamic Ride-sharing

Dynamic The ride-share can be established on short-notice, which can range from a few minutes to a few hours before departure time. The growing use of Internet-enabled mobile phones allows people to offer and request trips whenever they want, wherever they are. Thus, communication technology is a key enabler to dynamic, on-demand ride-sharing.

Independent The drivers which provide the rides are independent private entities. This is different from most traditional forms of passenger transportation where a central organization owns vehicles and/or employs drivers.

Cost-sharing The variable trip-related costs are reallocated among the ride-share participants in a way that makes it beneficial for them to participate from the perspective of cost reduction. The variable trip cost minimally includes fuel expense, but may also take into account wear and tear on vehicles, parking costs or road fees such as tolls.

Non-recurring trips Dynamic ride-sharing focuses on single, non-recurring trips. This distinguishes it from traditional carpooling or vanpooling, both of which require a long-term commitment among two or more people to travel together on recurring trips for a particular purpose, often for traveling to work. Single-trip ride-sharing is more flexible because it does not require rigid time schedules or itineraries over time.

Prearranged The trips are prearranged which means that the participants agree to share a ride in advance, typically while they are not yet at the same location. This is different from the spontaneous, so-called casual ride-sharing (see *e.g.*, [34]) in which riders and drivers establish a ride-share on the spot, similar to

hitch-hiking or hailing a taxi on the side of the street. In casual ride-sharing, drivers and riders line up at established locations to share rides to other established locations to take advantage of high occupancy vehicle lane time-savings or toll savings. The main limitation of casual ride-sharing is the inflexibility of its routes, which does not allow door-to-door transportation.

Automated matching To establish ride-shares in a way that requires minimal effort from the participants, ride matching should be automated in a dynamic setting. This means that a system matches up riders and drivers and communicates the matches to the participants. We do not include in our definition simple (online) notice boards where riders and drivers can post desired or planned trips and choose to contact potential ride-share partners themselves.

2.2.2 The Ride-share Process

To facilitate a discussion on the planning issues in dynamic ride-sharing, we briefly explain the process of Avego, an Ireland-based software company that currently offers a dynamic ride-share application for Internet-enabled mobile phones. The service that they offer is quite generic and similar to that of other existing ride-share providers such as Carticipate, Piggyback, and EnergeticX.

With the Avego Shared Transport software application, users can offer a ride as a *driver* or a request for transportation as a *rider*. To facilitate easy trip specification, the application lets users store and select pre-defined locations such as home, work, and the grocery store. With a GPS-enabled phone, users can set their current location as the origin of their trip, even en-route. If a ride-share match is established, Avego proposes the arrangement to the participants. If the driver and the rider agree on the proposed arrangement, the driver picks up the rider at the agreed time and location. Avego sends the driver the rider's photo and personal identification number, which allows him to verify the rider's identity.

Avego will guide the driver to an appropriate pickup location and from thereon to the rider's destination via the incorporated navigation system. When the driver is in range for the pickup, the application will notify the rider in real-time. Avego automatically assesses a trip fee to the rider, of which the company receives a fixed percentage.

2.2.3 Ride-sharing System Objectives

Ride-sharing allows people to save on travel-related expenses by sharing trip costs. A ride-share provider, either private or public, helps people to establish ride-shares on short-notice by automatically matching up drivers and riders. If the system is private and operated for profit, the value provided by the ride-share provider is to reduce the total costs of all participants by the largest amount possible; by enabling this economy, the provider receives as payment a percentage of the savings generated. Private ride-share providers typically charge a commission per successful ride-share, either a fixed fee or proportional to the trip cost. As a result, the objective of the provider is mostly in line with the goals of the participants.

This is also true for a public system with a societal objective, such as the reduction of pollution and congestion. The objectives of the ride-share provider and ride-share users are aligned because both the total travel costs of the users and the external costs to society relate to the total system-wide vehicle-miles. Note that we implicitly assume here that a rider has a car at his disposal and will use that car to reach his destination if not matched up for a ride. This seems a reasonable assumption, especially in many US settings where 9 out of 10 people own a car [49]. Enhancing the mobility of system users without cars can be thought of as an important additional societal benefit.

Given these system objectives, most studies on ride-sharing consider one (or a combination) of the following objectives when determining ride-share matches:

- *Minimize system-wide vehicle-miles (M)* The system-wide vehicle-miles represent the total vehicle-miles driven by all participants traveling to their destinations, either in a ride-share or driving alone. This objective is important from a societal point of view since it helps to reduce pollution (emissions) and congestion. This objective is also compatible with *minimizing total travel costs*, which is an important consideration for the participating drivers and riders and directly related to the revenues of the ride-share provider.

- *Minimize the system-wide travel time (T)* The travel time is the time spent in the vehicle while actually traveling between origin and destination. From a societal perspective, this is an important measure since vehicle emissions not only relate to vehicle-miles but also to vehicle speeds. Obviously, time is also an important convenience consideration for the participants.

- *Maximize the number of participants (P)* This objective maximizes the number of satisfied drivers and riders in the system. This objective may be beneficial for a private ride-share provider whose revenues are linked to the number of successful ride-share arrangements. Moreover, the matching success-rate may also be an important performance indicator for users of a particular ride-share service, and a high success rate may spur larger participant pools in the future.

2.2.3.1 Constraints on Matches

When determining matches between drivers and riders in a ride-share system, a number of constraints on the feasibility of matches must be observed. The timing of rides is probably the most important consideration since time tends to be a more constraining factor than the availability of spare seats.

Both riders and drivers must provide information on their time schedule preferences. Many of the currently available and proposed dynamic ride-share applications simply let each potential participant specify a desired departure time. The provider

then attempts to find an assignment with a departure time that is as close as possible to this desired departure time. This approach minimizes the information that participants must supply, but, at the same time, provides only limited information regarding a participant's time preferences and flexibility. Therefore, most studies capture a participant's time preferences by a time window representation. For example, a participant could specify an earliest possible departure time and latest possible arrival time (see Figure 1) [4]. Furthermore, limits on the actual time that users may spend traveling on a given trip could be allowed [7, 8]. That is, each participant is allowed to specify the maximum excess travel time (over the direct travel time for his origin to destination) he is willing to accept.

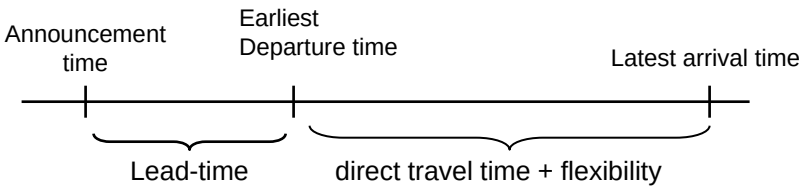

Figure 1: Time Schedule Information in Ride-sharing

In addition to time, there are other important feasibility considerations that determine whether a particular ride-share match is one that the participants would accept. For example, female participants may not feel safe sharing a ride alone with a male stranger [41], while smoking may be another critical issue [28]. The user may only feel comfortable sharing a ride with certain groups of people, where the group preferences may be motivated by personal safety or social considerations. For example, one may not be willing to share a ride with a complete stranger and may only want to share rides with friends and colleagues. Of course, the more restrictions a potential user places on his pool of potential ride-share partners, the more difficult it will be to find successful matches for that user [20].

Lastly, ride-share users may choose to participate primarily to reduce their travel

costs. Therefore, it is probably also necessary to include constraints that restrict feasible matches to those that reduce the travel costs of each ride-share participant. Of course, determining whether or not a user reduces his costs via a ride-share depends on how costs are shared in such systems, which is the subject of the next section.

2.2.4 Cost Considerations

People may choose to participate in ride-sharing primarily for potential cost-savings; trip-related expenses, such as fuel and tolls, are shared. Thus, ride-shares should only be established if they reduce the cost of each individual participant. Although large cost savings may eventually come from riders giving up automobiles, freeing themselves from the capital and insurance costs associated with owning or leasing a car, ride-sharing is unlikely to reduce private car ownership in the near future. In the short term, individuals will not give up their cars, making the costs of ownership essentially fixed and thus not pertinent to the travel decisions. Therefore, the literature on ride-sharing typically focusses on variable travel costs that are proportional to vehicle-miles.

When travel costs are roughly proportional to distance traveled, cost reduction is only possible when the length of a ride-share trip is shorter than the sum of the lengths of the separate trips. Note that a complete ride-share trip should be defined as all travel required to move each participant from his origin to his destination. For example, in the case where a driver shares a ride with a single rider this would include travel from the driver's origin to the rider's origin, then onto to the rider's destination, and finally onto the driver's destination.

If the cost of ride-share trip is less than the sum of the costs of individual trips of its participants, it is always feasible to allocate the cost savings among the participants such that each individual receives cost savings. Each driver can reduce his trip cost by receiving compensation that is greater than the marginal cost required to

accommodate the rider(s), *i.e.*, the marginal travel cost required by detours necessary to serve the riders. For the ride-share to be beneficial for a rider, the compensation he pays to the driver(s) should be lower than the cost of driving themselves with their own car. There are various ways to divide the trip costs between the ride-share partners. A natural way to allocate the costs of the joint trip is proportional to the distances of the separate trips [4]. It is also reasonable to simply split the cost equally among all participants.

The cost-savings threshold for participation may differ from person to person. Some people may only participate if their trip cost is reduced by at least x percent, whereas for others the social and environmental benefits of ride-sharing may be reason enough to participate (they may even be willing to accept a small increase in their trip costs). Riders without a car at their disposal may be willing to pay more than the cost of driving alone since alternative transportation options may be costly and/or inconvenient. An auction-based mechanism to determine the driver's compensation is proposed by Kleiner et al. [35]. This approach takes into account the different ride valuations of individual riders. For each rider, they simulate an individual willingness to pay per mile that lies between the cost of driving alone in a private car and the cost of taking a taxi. In this environment, higher driver compensations correspond with more ride-share matches because drivers accept longer detours.

Note that local regulations may not allow cost-sharing on a per trip basis (see *e.g.*, http://save.pickuppal.com/). In such situations, riders may compensate rides over time by offering rides as a driver. Several papers study the expected *fairness* of different algorithms to decide in real-time whether a participant should act as a driver or a rider (see *e.g.*, [5, 25] and [48]).

2.2.5 System Versus User Benefits

It is important to recognize that a system-wide optimal solution aimed at minimizing the external societal costs may not necessarily optimize the cost-savings of all individual ride-share participants. Consider a system with two drivers d_1 and d_2, two riders r_1 and r_2, and locations and distances given in Figure 2. Each driver can accommodate a single rider. When minimizing system-wide vehicle-miles, the optimal solution has value 20 and matches d_1 with r_1 and d_2 with r_2; represented by bold paths in Figure 2. When costs are allocated proportionally, each driver and rider pays the costs of 5 miles, *i.e.*, $\frac{6}{12}$ of the joint trip length of 10 miles. However, driver d_1 and rider r_2 could reduce their trip costs even more by establishing a ride-share on their own, since the joint trip length would be 9 miles of which each would pay for 4.5 miles. In this case, driver d_2 and rider r_1 would be without a ride-share since their joint trip length would be 17 miles. In the terminology of cooperative game theory, the system-wide solution is not stable.

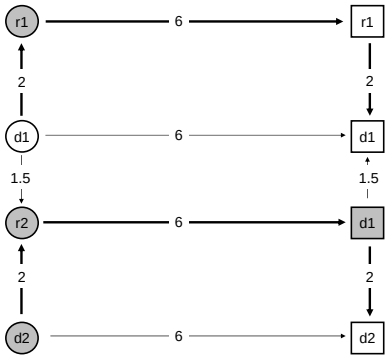

Figure 2: Riders (grey) and Drivers (white) Traveling from Origin (circle) to Destination (square)

2.3 Basic Ride-sharing Problems

In this section, we describe optimization problems for finding matches. We limit our attention here to what we will denote as static ride-sharing variants, where it is assumed that all driver and rider requests are known in advance prior to the execution of a matching process. In Section 2.4, this restriction will be relaxed as we examine the more relevant problems of dynamic matching.

Drivers offering a ride may want to take a *single* rider or may be willing to take *multiple* riders. Similarly, riders requesting a ride may want to ride with a *single* driver or may be willing to ride with *multiple* drivers and transfer from one to another en route to their destinations. Thus, we can distinguish four basic ride-sharing system variants as shown in Table 1.

Optimally matching drivers offering a ride and riders requesting a ride is easy for the static variant in which a single driver takes along a single rider. In all other variants, determining the best route sequence for a given match, which may involve multiple drivers and riders, can be more complicated.

Table 1: Ride-share Variants

	Single Rider	**Multiple Riders**
Single Driver	Matching of pairs of drivers and riders	Routing of drivers to pickup and deliver riders
Multiple Drivers	Routing of riders to transfer between drivers	Routing of riders and drivers

To be able to properly discuss the ride-share problems, we start by introducing some notation.

We are given a set of locations P and travel time t_{ij} and travel distance d_{ij} between each pair of locations $i, j \in P$. Furthermore, we are given a set of drivers D and a set of riders R. Each driver $d \in D$ (rider $r \in R$) wants to travel from his origin $v(d) \in P$

$(v(r) \in P)$ to his destination $w(d) \in P$ $(w(r) \in P)$. Each driver $d \in D$ (rider $r \in R$) has an earliest time $e(d)$ $(e(r))$ at which he can depart from his origin $v(d)$ $(v(r))$ and a latest time $l(d)$ $(l(r))$ at which he can arrive at his destination $w(d)$ $(w(r))$. Each driver d has $q(d)$ spare seats available.

2.3.1 Single Rider, Single Driver Arrangements

If a driver would like to share a ride with at most a single rider, then at most one pickup and delivery take place during his trip. Thus, if driver d and rider r are matched, then their joint trip length is $d_{v(d),v(r)} + d_{v(r),w(r)} + d_{w(r),w(d)}$. By comparing the vehicle-miles of the joint trip with the two separate trips, we can easily calculate the potential savings for each driver-rider match (see Figure 3).

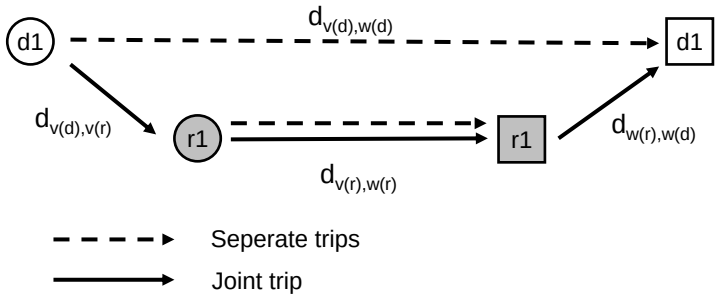

Figure 3: Example of Single Rider, Single Driver Rideshare Arrangement

If we want to match drivers and riders in the system in a way that minimizes the total system-wide vehicle-miles, the driver-rider match optimization problem can be represented as a maximum-weight bipartite matching problem (also known as the assignment problem). The bipartite graph consists of two disjoint sets of vertices, a set representing drivers D and a set representing riders R. An edge between a driver and a rider exists if the match is feasible, with a weight that represents the positive savings in distance when traveling together compared to when each of them drives separately. More formally, a constraint on positive cost savings implies a necessary condition for the feasibility of a match between driver d and rider r: only

if $d_{v(d),w(d)} + d_{v(r),w(r)} - (d_{v(d),v(r)} + d_{v(r),w(r)} + d_{w(r),w(d)}) > 0$. Moreover, matches must also be time feasible, where both the rider's and the driver's travel windows are respected. The assignment problem has been studied extensively in the literature with algorithmic approaches abound (see [51] for a review).

Amey [7] studies the ride-share potential at the MIT campus in Cambridge, Massachusetts. Given the home locations and time schedules of the faculty and staff, the author identifies potential ride-share arrangements in which two commuters share a trip together. The ride-share match optimization in this case must not only decide on the assignment of riders to drivers but also assign a role to each of the participants. It is therefore not possible to model this problem using bipartite matching. The author formulates the problem as a general network flow problem with side constraints to ensure that a commuter was not matched up as both a driver and a rider in separate ride-share arrangements. The study indicates a potential reduction of system vehicle miles of between 9% and 27%, depending on the maximum acceptable driver detour. Note that when a single driver travels with at most a single rider and riders do not transfer, the total system-wide vehicle-miles traveled by participants can be reduced by no more than 50%. The reason for this is that the length of the joint trip can not be smaller than the larger of the individual trips of the ride-share partners. Again when transfers are not allowed and a driver can ride with at most $q(d)$ passengers, we can save at most $1/q(d)$ of the system-wide vehicle-miles by ride-sharing.

As an additional note, we should point out that it is likely that the very large majority of ride-share participants will need to plan round trips. In a dynamic ride-share system with a sufficient amount of capacity, the rider should be able to arrange the trips separately shortly before departure. However, some riders may not feel comfortable going to certain destinations without a guarantee that they will be able to find a ride back (for example, because the alternatives may be very costly). The need for round trip planning may necessitate that systems allow riders to place two

transportation requests at the same time. These two request are directly linked if the rider's alternative means of transportation is his own car. In this case, a rider may only want to commit to a ride-share if both legs are covered. The return trip does not necessarily have to be conducted by the same driver that provided the departing trip.

2.3.2 Single Driver, Multiple Rider Arrangements

If drivers have sufficient time flexibility, they may be willing to provide rides to several riders on a trip, either one after the other or simultaneously for portions of the time. The pickup and drop-off of multiple riders in a single trip gives rise to more complex routing decisions.

The carpool problem is a special case of this ride-sharing variant. In the carpool problem workers, partitioned into riders and drivers, want to go to their common work location from their homes. The objective is to assign riders to drivers and construct feasible routes for drivers to minimize the travel costs plus a penalty associated with unserved riders. Each worker has an earliest time he can leave home and a latest time he can arrive at work. Furthermore, each driver has a maximum time he is willing to spend driving from home to work.

Baldacci et al. [8] addresses the *to-work* variant of the carpool problem separately from the *return-from-work* variant. They propose both an exact and heuristic method to solve the problem based on two integer programming formulations. They solve several instances, some based on real-world data, with the number of workers ranging from 50 to 250. Calvo et al. [15] studies the problem using a model that allows different network travel times at different times of the day. They develop a heuristic approach to solve the problem based on construction and local search. In a computational study using real-life carpool data, the authors investigate the impact of varying the ratio of drivers to riders and show that the total system-wide travel

time increases with the ratio between driver and riders.

The carpool problem is a special case of the so-called pickup and delivery problem, which has been studied extensively in the operations research literature; see *e.g.*, [62]. These problems involve the construction of vehicle routes and schedules to satisfy transportation requests between origins and destinations. A fleet of vehicles with a given capacity is available to operate the routes, typically based at one or more depot locations. It is also usually assumed that the pickup and delivery of each individual request is made by one vehicle.

The dial-a-ride problem is a special case of the pickup and delivery problem that focuses on the transportation of passengers [12, 17]. Consequently, passenger convenience considerations become important. Passenger service quality may be measured, for example, in terms of the ratio of actual drive time and direct drive time, the waiting time, the number of stops while on board, and the difference between actual and desired delivery times [50]. These criteria may be treated as constraints or may be incorporated into the objective function.

Dynamic ride-sharing differs from conventional on-demand transportation primarily with regards to the supply of drivers and vehicles. Instead of being employed by a company, drivers in a ride-sharing system are private independent entities. Like riders, they arise dynamically over time at various locations in a process that may be difficult to predict with certainty. Since they are independent, they are not obligated to accept ride-share arrangements that they do not like. Therefore, driver preferences need to be accounted for when matching drivers and riders in a ride-sharing system. Driver preferences may include a maximum deviation from the direct trip duration, a maximum number of simultaneous riders, and a maximum number of stops.

Another important difference between a dial-a-ride system and a ride-share system is that in a dial-a-ride system all vehicles typically operate out of one or more

depot locations, whereas in a ride-share system each driver may have a unique origin and destination. This implies that in a ride-share system, routing decisions are represented and evaluated as deviations from a driver's direct path from origin to destination. Deviations from a given path are also at the heart of a Mobility Allowance Shuttle Transport (MAST) service, in which a vehicle has a predefined route but is allowed to deviate from this route to pick-up and drop-off passengers at preferred locations within a certain service area [56, 73]. In addition, customers who board the vehicle at a scheduled stop can request a drop off location that is within half a mile from the predefined route [73]. The MAST concept aims at combining the flexibility and convenience of on-demand transportation with the cost-efficiency of fixed route transit. Los Angeles County operates a MAST during the night hours. Passengers located within half a mile off the route may call-in for pick-ups at off-route locations.

2.3.3 Single Rider, Multiple Driver Arrangements

If we allow riders to transfer between drivers, a rider may travel with more than one driver to reach his final destination. Gruebele [29] describes such a multi-hop ride-share system in detail. Potential transfer points could include public transport stops, shopping malls, or park-and-ride lots.

Herbawi and Weber [32] considers a version of the multi-hop ride-share problem in which drivers do not deviate from their routes and time schedules. As such, the drivers' ride-share offers form the transportation network over which the rider has to find a route that minimizes costs, time and number of transfers. The authors model this problem as a multi-objective shortest path problem on a time-expanded graph representing the drivers' offers. The authors present an evolutionary solution approach to solve the problem and show that this approach is able to provide good quality solutions in reasonable running times.

The multi-hop ride-sharing problem is more difficult when also considering the

routing of the drivers. In a dial-a-ride environment, Cortes et al. [19] extends the standard pickup and delivery problem formulation to facilitate passenger transfers from one vehicle to another. The formulation allows the specification of one or more potential transfer points and maximum passenger waiting times at these points. The authors present an exact solution method and shown its effectiveness on small instances with one transfer point, 2 vehicles and up to 6 requests. Furthermore, they show that allowing transfers between vehicles can be beneficial in some settings.

An interesting area of research related to multi-hop passenger transportation systems also concerns system design. Rather than focusing only on effectively routing passengers through a given network with transfer points, another important focus could be on where to locate the transfer locations. There is a huge area of research on this topic in freight transportation and airline passenger networks, where transhipment points are typically called hubs. For a comprehensive review see [6] and [16].

2.4 Dynamic Ride-sharing Problems

In any practical dynamic ride-share implementation, new riders and drivers continuously enter and leave the system. A driver enters the system by announcing a planned trip and offering a ride, while a rider enters the system by announcing a planned trip and requesting a ride. Drivers and riders leave the system when a ride-share arrangement has been planned and accepted, or when their planned trips "expire," *i.e.*, when the latest possible departure time of a planned trip occurs before a successful arrangement can be found.

To avoid one potential worry for potential participants, it may be better to have each participant specify a trip expiration time in addition to (and which may be earlier than) his latest possible departure time. If the announcement time of a trip for driver d (rider r) is denoted $a(d)$ ($a(r)$) and the expiration time of the trip by $b(d)$

$(b(r))$, then the window for matching driver d (rider r) is $[a(d), b(d)]$ ($[a(r), b(r)]$). Due to these matching windows, a match between driver d and rider r (assuming they have otherwise compatible trips) can be established only in the time interval $[\max\{a(d), a(r), \min\{b(d), b(r)]$.

2.4.1 Arrival of Riders and Drivers

Since new drivers and riders continuously arrive, not all relevant offers and requests may be known at the time the ride-share provider executes an algorithm for planning ride-sharing arrangements. In recent years, several authors have addressed this issue in dynamic ride-sharing.

In [4], we deal with this planning uncertainty by using a rolling horizon solution approach. In this approach, the optimization problem to be solved includes all of the offered rides (drivers) and requested rides (riders) that are known at the time of execution and that have not yet been matched. We evaluate different re-optimization frequencies and run the algorithm for finding ride-share arrangements each time a new request arrives or at fixed time intervals. Moreover, we experiment with different commitment strategies, *i.e.*, immediately notify drivers and riders of the matches identified by the optimization, or wait before notifying so as to find improved matches at the next execution time. We observe that systems that employ the latest commitment strategy for matches should be optimized more frequently. However, in a system that immediately commits matches, we observe that there are advantages of optimizing less frequently since it allows the accumulation of more trip announcements between optimization runs. Moreover, the results indicate that sophisticated optimization methods outperform simple greedy matching rules in terms of the number of established rides-share matches and vehicle miles savings.

Several papers consider an agent-based system where autonomous rider and driver agents *locally* establish ride-shares. [70] and [71] consider such agent-based rideshare

systems with the objective of maximizing the number of served riders that do not have a car to their disposal. Winter and Nittel [70] considers a setting in which wireless communication devices are used that only enable short-range communications (*e.g.*, Bluetooth or WiFi). They show that limiting the information dissemination between agents does not significantly impact the solution quality. Xing et al. [71] considers a highly dynamic ride-share system where drivers and riders are matched *en-route*. The participants announce their trips (offers and requests) at their departure time, *i.e.*, $a(d) = e(d)(a(r) = e(r))$. They incorporate gender and smoking preferences and specify a maximum acceptable service response time for the riders. Riders may walk to a pickup-point to facilitate easy pickup by the driver. Simulation experiments on a real-life urban map of the Bremen metropolitan area show that the probability of a successful ride-share arrangement increases with the number of available drivers. The experiments also suggest that with sufficient drivers, dynamic ride-sharing may be an attractive alternative to public transportation in terms of travel time.

In a similar setting, Kleiner et al. [35] applies a rolling horizon solution approach where arrangements are committed as late as possible given the time considerations. They present an auction-based solution mechanism that takes into account the individual preferences of the participants for ride-share partners and rides. The simulation experiments show that the auction-based approach provides close-to-optimal solutions to the ride-sharing problem.

The dynamics of the arrival of new rider requests and driver offers is not unique to ride-sharing. Various other passenger transit applications such as taxis share similar features. For an excellent recent review on *dynamic* pick-up-and-delivery problems see [11] and [18]. A transportation request for an urban taxi typically arrives only a short time before the desired departure [39] and vehicle routes and schedules are updated each time a new transportation request arrives. The dynamic ride-sharing environment resembles an urban taxi environment in terms of the arrival process of

transportation requests, *i.e.*, rides, but also has the added complexity of an arrival process of transportation resources, *i.e.*, drivers.

Note that it is often passenger convenience rather than physical capacity that keeps taxis from serving multiple passengers simultaneously. Horn [33] demonstrates that allowing multiple passenger parties together in a single taxi trip may decrease system-wide vehicle miles but increase the individual travel times of the passengers. They present a dispatching software to manage a fleet of demand-responsive taxis taking into account both passenger service quality considerations and fleet efficiency considerations. The system assigns new travel requests to vehicles based on minimum cost criteria and then periodically applies improvement procedures. The author conducts a number of simulation studies based on data from a real-life taxi operator in Australia. The tests show that the software tool operates effectively in a fairly dynamic environment and realistic problems sizes.

Dial [21] proposes an autonomous dial-a-ride taxi service that shares many similarities with dynamic ride-sharing. The fully automated system lets passengers reserve trips by phone or computer on short-notice. For routing and dispatching, the author suggests the use of a dynamic programming approach [55]. The dynamic algorithm reoptimizes the not yet executed part of the tentative optimal route each time a new requests appears. Since the algorithm can only solve very small instances the system only includes passenger requests that must be served in the near future and run the algorithm for each vehicle individually.

In the area of freight transportation, full truckload carriers have to manage fleets of vehicles (e.g. containers, trailers, boxcars) that serve one load at a time, with orders continuously arriving over time (for a review see [53]). The problem of sequentially assigning transportation requests to vehicles is typically referred to as the dynamic assignment problem [64] or the dynamic stacker crane problem [11]. Yang et al. [72] considers the real-time multi-vehicle truckload pickup and delivery problem. In this

problem, requests for truck-load moves arrive over time. Each request has a time window during which a pickup must take place. The authors have modeled the static problem as an integer linear program. To handle the dynamics, the static problem is solved repeatedly in a rolling-horizon framework. The authors compare three rolling-horizon, as well as two reoptimization policies.

2.4.2 Anticipation of Future Requests

After ride-share systems have been in operation for a while, it is likely that some information about future unknown ride offers and ride requests may become available. Instead of myopically optimizing for the offered trips and requested trips that are known, it may of course be possible to incorporate information that partially describes the stochastic future into a modeling and solution approach in order to improve system-wide cost savings. Powell [52] provides a formulation of the dynamic assignment problem in the context of a load matching problem that arises in long-haul truckload trucking. They assign drivers to loads on a real-time basis. A hybrid model is presented that handles the detailed assignment of drivers to loads, as well as handling forecasts of future loads. They compared a myopic model to an approximation of the stochastic, dynamic problem,and showed that the stochastic, dynamic model outperformed the myopic model in rolling horizon experiments.

2.4.3 Deviations from Planned Trips

Even if the participants agree on a specific ride-share arrangement, the identified arrangement may not be executed as planned because of no-shows, last-minute cancelations or delays. In case there are many such deviations from the agreed plan other, more robust, solution methods may have to be considered. For example, Powell et al. [54] studies this issue in a truckload-trucking setting. They conclude that even in a situation with a relatively small number of deviations from the recommended plan, simple greedy solutions can outperform optimal solutions.

Reputation systems, which are commonly used to support online sales transactions (the online marketplace eBay uses perhaps one of the most widely known reputation systems), could help to establish trust among participants and encourage reliable system behavior. A ride-share reputation system could provide drivers and riders the opportunity to rate each other. In addition, the ride-share provider could rate participants by monitoring cancelations, no-shows, and late arrivals. Such ratings could be converted into a reliability score. The reliability and feedback scores could then be used in the matching optimization to favor matches involving participants with high scores.

2.5 The Multi-modal Ride-sharing Problem

Instead of providing door-to-door transportation, the ride-share concept could be integrated with other modes of transportation, such as public transit. Ride-sharing may provide a very effective means to increase the use of a scheduled public transportation system if it can be used as a feeder service. In such a setting, a driver would first take a rider from the rider's origin to a public transport stop, then he would use public transit to get close to his destination, and finally he would walk or use another ride-share driver to travel from the transit stop to his destination. Aktalita, a project currently in development in Guadalajara, Mexico (www.aktalita.com) aims at developing such an integrated ride-share system.

To increase public transit usage, on-demand taxis to serve as a feeder for scheduled transit haa been proposed [40, 45]. Liaw et al. [45] considers the integration of paratransit dial-a-ride vehicles with fixed-route buses, in a system where transportation bookings are made in advance. They show that the combination of on-demand vehicles and scheduled transit allows for an increase of the number of accommodated requests while at the same time decreasing the number of required taxis. Lee et al. [40] considers the integration of dial-a-ride taxis with a metropolitan rapid transit

line. They study a highly dynamic environment where new transportation requests continuously arrive and are to be assigned to taxis en-route. They propose a dispatch strategy and determine an optimal required fleet-size taking into account passenger waiting and travel time, number of satisfied requests, and system costs. Li and Quadrifoglio [44] studies the service performance of an on-demand taxi feeder system known as the demand responsive connecter. They demonstrate that the on-demand system outperforms a scheduled feeder system if customer densities are relatively low.

An effective integration of a ride-share system and a scheduled public transit system will potentially increase, in a sense, the coverage area of the public transit system, which has many societal and environmental benefits. However, the transfer from one mode of transportation to another must be seamless and efficient, and without long waiting times, before large numbers of people will make use of the integrated system. Consistent seamless and efficient mode transfers will only be possible with effective optimization technology.

2.6 Conclusions

New dynamic ride-sharing systems have the potential to provide huge societal and environmental benefits. The development of algorithms for optimally matching drivers and riders in real-time is at the heart of the ride-sharing concept. We have formally defined dynamic ride-sharing, have highlighted many of the interesting optimization challenges that arise when developing technology to support dynamic ride-sharing and have reviewed the relevant operations research models in this area. We have seen that there is a growing interest from the research community to address the optimization issues in dynamic ride-sharing but that the number of specific contributions to date is still small. We see room for contributions in all areas of dynamic ride-sharing. In particular, we see the following broad areas for future research: (1) fast optimization approaches for real-life instance sizes (2) incentives schemes to build critical mass and

(3) optimization approaches that allow choice.

Optimization

It may be unlikely that realistic-size instances of the model can be solved fast enough to be of use in a matching engine of an actual, sustainable ride-share system. In a major metropolitan area, thousands of riders and drivers travel between thousands of origins and destinations during the same time periods, which leads to very large optimization problems that may have to be solved very quickly as often as once every few minutes. Thus there is a clear need for fast solution approaches producing high-quality results.

Another area of research that should be of interest to the transportation science and logistics community is the design of de-centralized ride-share matching techniques. Centralized ride-share matching may not be practical for many larger metropolitan areas, or may not be computationally feasible. In such cases, effective decomposition approaches will be necessary. A simple decomposition based on a geographic partition is likely to be challenging since driver and rider trips involve both an origin and a destination location, and these locations may often be separated by significant distances. The existence of trip requests for which the origin location is in one subregion and the destination location is in another subregion is therefore quite likely. It is also not clear whether a static partition of the region suffices or whether the partition of the region should be adjusted dynamically based on the set of driver and rider trip requests that need to be matched.

Incentives

The financial benefits of sharing trip-related expenses may motivate people to participate in ride-sharing. Rising fuel costs, pay-per-mile auto insurance and congestion pricing may further increase the cost of private car use in the future, and thus strengthen the advantages of ride-sharing. However, without a sufficient number of drivers and riders, the chance of finding a ride, especially one close to the

desired departure time, may be very small, and thus the inconvenience may outweigh the financial benefits. To achieve the required density, *i.e.*, the number of necessary drivers and riders for a sustainable ride-share system, local governments and businesses may need to subsidize ride-share initiatives. These subsidies can be used to reward ride-share participants, either on a per-trip basis or a per-offer basis. Subsidizing commercial urban taxis to act as drivers may also be advantageous especially in a start-up phase. Subsidized ride-share systems may provide a relatively inexpensive way to increase the capacity and efficiency of the transportation system, a potentially interesting alternative to the capital investments required to build or expand the road network or expand public transportation.

Choices

A good understanding of participant behavior and participant preferences will be essential when designing of a dynamic ride-sharing system. If ride-share matches do not satisfy participant preferences, the match may not be accepted, or the participant may not make use of the ride-share system in the future. Unfortunately, providing comprehensive preferences may be difficult and time-consuming for participants, partly because preferences may be interdependent and may change from one day to the next. For example, a driver's time flexibility may depend on the day of the week, the financial benefits, and the specific rider. Moreover, some participants may be hesitant or unwilling to disclose certain preferences for privacy reasons.

Rather than being notified of a specific single ride-share match, participants may prefer to choose from a menu of available ride-share options. However, the selection process must not take too much time. Minimally, then, the ride-share provider should present only the best options and only the most relevant information regarding these options, which may include the pickup and drop-off times, the travel time, the financial benefits, but also person specific information, such as gender, age, professional profile, and feedback and reliability scores. Providing a menu of ride-share options

introduces various system synchronization issues. If driver trips (rider trips) appear as an option for several riders (drivers) simultaneously, there is a chance that preferred options clash, *i.e.*, the same driver trip or the same rider trip is chosen multiple times. Designing a selection-based matching process is non-trivial and would likely pose interesting new additional challenges for the underlying matching optimization engine.

CHAPTER III

SINGLE-RIDER, SINGLE-DRIVER RIDE-SHARE MATCHING OPTIMIZATION

3.1 Introduction

In this chapter, we develop optimization-based approaches for finding ride-share matches in a standard problem setting, with the goal of minimizing the total system-ride vehicle miles incurred by the system users. To assess the merits of our methods we present a simulation study based on 2008 travel demand data from metropolitan Atlanta. The simulation results indicate that the use of sophisticated optimization methods instead of simple greedy matching rules substantially improve the performance of ride-sharing systems. Furthermore, even with relatively low participation rates, it appears that sustainable populations of dynamic ride-sharing participants may be possible even in relatively sprawling urban areas with many employment centers.

The main contributions of this chapter can be summarized as follows:

- We develop optimization approaches specifically tailored to the dynamics of a practical ride-share environment where new drivers and riders continuously enter and leave the system. The rolling horizon approach provides high quality solutions to practical dynamic ride-share problem instances;

- We build a simulation environment based on travel demand model data from the Atlanta Regional Commission, and use it to test dynamic ride-sharing concepts. The simulation results suggest that dynamic ride-sharing may represent a useful option to reduce system-wide vehicle miles, reduce trips and save travel costs,

even when participation rates are relatively small;

- We demonstrate the value of more sophisticated optimization techniques over simple greedy matching methods in dynamic ride-sharing systems.

The remainder of this chapter is structured as follows. In Section 3.2, we describe the dynamic ride-sharing setting and explain the planning issues that arise in this context. In Section 3.3, we explain our approach to solve the dynamic ride-share problem. In Section 3.4 we present a simulation study based on the travel demand model of the Atlanta Regional Commission. In Section 3.5 we focus on understanding the performance of a ride-sharing system over time. Finally, in Section 3.6, we summarize our main insights and discuss directions for future research.

3.2 The Dynamic Ride-share Setting

We consider a specific dynamic ride-share system setting that we believe is representative of many new and proposed systems. In this setting, a ride-share provider for a particular metropolitan area receives a sequence S of trip announcements over time from potential participants. Each announced trip specifies whether the participant intends to be a driver, intends to be a rider, or is flexible to perform either role. A trip announcement also contains an origin and a destination location, and additional information that specifies its potential timing. With this information, the provider automatically establishes ride-shares over time, matching potential drivers and riders.

Suppose for simplicity that each origin and destination location is a member of a set P of locations, and that the travel time t_{ij} and travel distance d_{ij} between each pair of locations $i, j \in P$ are known and constant. Let $v(s)$ and $w(s)$ represent respectively the origin and destination of trip announcement $s \in S$.

We furthermore adopt the following reasonable model of trip timing, assuming that most trips are made with some flexibility in their schedule [23]. For each announcement $s \in S$, the participant provides an earliest time $e(s)$ at which he can

depart from his origin $v(s)$ and a time flexibility $f(s)$ that specifies the difference between $e(s)$ and the latest time he would like to depart by if he were driving alone (see Figure 1). For example, if a driver wished to arrive at his destination no later than $l(s)$, then we have time flexibility $f(s) = l(s) - e(s) - t_{v(s),w(s)}$. In this research, one condition for the feasibility of a ride-share match is that the participant for announcement s departs his origin no earlier than $e(s)$ and arrives at his destination no later than $l(s)$. We choose not to model any additional constraints that limit the amount of time participants spend traveling in-vehicle.

A participant announces his trip at time $a(s)$ shortly before or at his earliest departure time. The announcement lead-time $a^l(s) \geq 0$ denotes the difference between the participant's earliest departure time and his announcement time.

Although a potential driver may typically have several spare seats available (see *e.g.*, [1]), time considerations will restrict the number of stops he is willing to make in a single trip. To minimize the inconvenience of the participants, in this research we limit our attention to systems where at most one pickup and delivery can take place during the trip and no transfers occur (see 4). This does not imply that a driver cannot accommodate multiple riders if they are traveling from the same origin to the same destination at the same time.

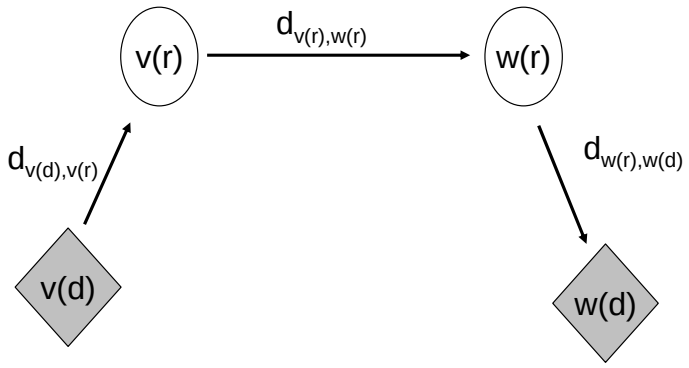

Figure 4: A Shared Trip between Driver d (squares) and Rider r (circles)

People may choose to participate in a ride-sharing to reduce travel costs. In this research, we focus on systems designed to enable users to share variable trip costs. When such costs are roughly proportional to distance traveled, cost reduction is only possible when the length of a ride-share trip is shorter than the sum of the lengths of the separate trips. If the cost of ride-share trip is less than the sum of the costs of individual trips of its participants, it is always possible to allocate the cost savings among the participants such that each individual benefits. We consider a match feasible only if it provides positive cost savings: a ride-share between driver d and rider r is feasible only if $d_{v(d),w(d)} + d_{v(r),w(r)} - (d_{v(d),v(r)} + d_{v(r),w(r)} + d_{w(r),w(d)}) > 0$.

A trip announcement s is said to expire when the latest possible departure time $e(s)+f(s)$ occurs before a successful ride-share match can be found. Thus, ride-shares cannot be arranged for potential drivers that are already en-route. Furthermore, virtually all trips in practice are likely to be round trips. While a potential rider participant may choose to arrange ride-shares for the trips separately, some may not feel comfortable traveling to certain destinations without having a confirmed ride back. The need for round trip planning may necessitate that systems allow riders to place two trip announcements at the same time, and only agree to participate if both requests are matched in ride-shares. Of course, the return trip need not be with the same driver that provides the outbound trip.

Although ride-sharing systems may provide opportunities to increase the mobility of people that do not have access to public transit or a private vehicle, we focus on ride-sharing as a means to reduce travel costs, congestion and pollution. We therefore limit our attention to a setting where both drivers and riders have a car available which they could use to drive to their destination alone if no ride-share can be identified.

Given this setting, we explore ride-share optimization problems in which the ride-share provider seeks to minimize *total system-wide vehicle-miles*, the total vehicle-miles driven by all potential participants traveling to their destinations, either in a ride-share or driving alone if unmatched. This objective is aligned with societal objectives for reducing emissions and traffic congestion. Furthermore, since this objective seeks to maximize the total travel distance savings of all participants, it also coincides with *minimizing total travel costs*, an important consideration for the participating drivers and riders. Finally, if the ride-share provider is compensated with a fraction of the total travel cost savings of all participants, the objective is also consistent with *maximizing the revenues* of the provider.

3.3 Solving the Dynamic Ride-share Problem

3.3.1 Rolling Horizon Strategy

Since new driver and rider trip announcements continuously arrive each day, it seems clear that any dynamic ride-sharing service provider must determine potential matches at many time points during the day. Each time the provider executes a procedure for planning matches, there are likely to be future requests that are not yet known. A common mechanism for handling uncertainty of this type when planning is to use a deterministic rolling horizon solution approach, in which plans are made using all known information within a planning horizon, but decisions are not finalized until necessitated by a deadline. At each execution of the algorithm, the planning horizon is "rolled" forward to include more known information, and the process continues. Our proposed approach uses a planning horizon that extends forward from the current time and captures all currently known requests, regardless of their timing during the day.

A key decision when implementing a rolling horizon solution approach is how frequently, and specifically when, to execute the planning algorithm. One possibility

would be to initiate a matching optimization each time a new request becomes known. This, however, may lead to synchronization issues when a new announcement arrives before the end of the previous optimization run. For simplicity, therefore, we consider strategies that reoptimize at specific, regularly-spaced time points. Even so, in this study we ignore the time required to execute a planning algorithm, and assume that it is negligible.

In our solution approach, optimization run q at time $t(q)$ during an operational day considers all trip announcements s that were announced (at times $a(s)$) prior to $t(q)$, excluding expired announcements (where $e(s)+f(s) < t(q)$) and those that have been matched within finalized ride-share arrangements. For run q, we set the earliest departure time $e(s)$ of each remaining announcement s to $\max(t(q), e(s))$.

The optimization procedure then determines a best set of proposed ride-share matches as its output. Although matches may be found throughout the planning horizon, only a subset are finalized. We assume that the ride-share provider may notify participants about a ride-share as late as possible. Thus, a ride-share match is finalized only if the latest implied departure time of the driver must occur before the next scheduled optimization run. For a ride-share match with driver d sharing a ride with rider r, the implied latest departure time $\hat{l}(d, r)$ is given by $\min(l(r) - t_{v(r),w(r)} - t_{v(d),v(r)}, l(d) - t_{w(r),w(d)} - t_{v(r),w(r)} - t_{v(d),v(r)})$.

In the case where we determine round trip matches for riders, note that we also finalize the return ride-share match for a rider prior to the latest implied departure time of the driver for his outbound trip. Furthermore, for round-trip announcements in which the participant is willing to serve as a driver or rider, the role of the participant is finalized when his outbound ride-share match is finalized, and his role cannot change between the outbound and return trips; *i.e.*, a rider for an outbound trip cannot be scheduled in a return trip as a driver, and vice versa, since both cases are likely infeasible in practice.

In Section 3.3.2, we discuss the details of the optimization procedures used to determine matches within this rolling horizon approach.

3.3.2 Solving the Ride-share Matching Optimization Problem

Suppose that the optimization procedure is seeking to find the best ride-share matches from within the current set of active announcements, $S_A \subset S$. We first discuss the simplest case, where each participant declares whether he intends to be a driver or rider.

3.3.2.1 Fixed Driver, Rider Roles

There are two disjoint sets of announcements: a set $D \subset S_A$ representing driver trips, and a set $R \subset S_A$ representing rider trips. If the total benefit of a set of ride-share matches can be expressed as the sum of the benefits of individual matches, we can represent the ride-share problem using a maximum-weight bipartite matching model and then solve the problem using any linear programming or network optimization code. Since we consider a setting where the ride-share provider seeks to maximize the total distance savings produced for all participants, we can use this model as follows. We create a node for each announcement in $R \cup D$, and an arc connecting a node $i \in R$ on one side of the bipartition with a node $j \in D$ on the other side if it is feasible to propose a ride-share match with driver j and rider i; recall that a match must be both time feasible, and produce positive travel distance savings. The weight c_{ij} assigned to feasible match arc (i, j) is simply the travel distance savings. To complete the specification, let x_{ij} be a binary decision variable equal to 1 if ride-share match (i, j) is proposed, and 0 if not. Then, a formulation of the maximum weight bipartite matching optimization problem to maximize system travel distance savings uses objective function $\sum_{i,j} c_{ij}x_{ij}$, along with a set of constraints to ensure that each driver and rider is included in at most one proposed ride-share match: $\sum_j x_{ij} \leq 1 \ \forall i \in R$ and $\sum_i x_{ij} \leq 1 \ \forall j \in D$.

To solve the problem in our computational study, we use the standard commercial optimization software CPLEX. We transform the bipartite matching into a network flow maximum cost circulation problem by adding a source node s and a sink node t, along with an arc from s to rider node $i \in R$ with zero cost and unit capacity and an identical arc from each driver node $j \in D$ to t. Connecting to t to s with a zero cost and no capacity completes the specification.

It is not difficult to extend the bipartite matching model to the case where some (or all) of the riders wish to schedule round trip matches. To do so, we simply need to ensure that if a rider is matched on his outbound trip, that he is also matched on his return trip. Such riders i will be represented with two separate rider nodes i^1 and i^2, representing the two trip segments respectively. To ensure that these two segments are either both matched or neither are matched, we must add a bundle constraint for each such round-trip rider: $\sum_j x_{i^1 j} - \sum_k x_{i^2 k} = 0$.

3.3.2.2 Driver, Rider Role Assignment

We now consider the more complex case where some ride-share participants announce trips in which they are flexible to serve as drivers or riders. Clearly, ride-share match optimization in this case must not only decide on the assignment of riders to driver but also assign a role to each of the participants. It is therefore no longer possible to model this problem using bipartite matching, but we can instead use a general graph matching model as follows. Consider a directed network with a node for each announcement in S_A. A directed arc (i, j) between announcement i and announcement j is generated if the potential match is time feasible and has positive cost savings c_{ij} when i serves as a rider and j as a driver, and an arc (j, i) with cost savings c_{ji} if it is feasible for j to ride and i to drive. If both arcs are generated, then we retain only the one with larger cost savings c. The matching objective function again seeks to maximize the savings of selected matches over all possibilities: $\sum_{i,j} c_{ij} x_{ij}$. Then,

a single matching constraint is used to ensure that each announcement is selected to be included with no more than one proposed match: $\sum_j x_{ij} + \sum_j x_{ji} \leq 1 \quad \forall\, i \in S$. Note that this constraint considers all outbound arcs ("rider" arcs) and inbound arcs ("driver" arcs) for announcement i.

The general graph matching problem can be solved with algorithms of polynomial complexity [22]. Again, however, if we need to solve problems with requests for round-trip matching, it is necessary to add bundle constraints that then require binary integer programming software. For this case, the required bundle constraints take the same form: $\sum_j x_{i^1 j} - \sum_k x_{i^2 k} = 0, \forall i^1, i^2 \in S$, where i^1 represents the outbound trip announcement and i^2 the return trip of participant i. Note that since we only bundle outbound arcs from i^1 and i^2, this constraint only matters when participant i is selected as rider. If i is flexible and is used as a driver, he may be matched only on outbound, only on return, or for both trips. Furthermore, note that these constraints also ensure consistent role assignments within a round trip of a rider, so that if a participant is matched as a rider on the outbound he must also be matched as a rider on the return. This is necessary since a participant who shared a ride to work likely does not have access to a vehicle for the return trip home.

It is also necessary when considering round-trip matching in this case to include both arcs (i, j) and (j, i) if they are both feasible, even if one dominates the other in terms of cost savings. For example, consider a problem in which i and k can be feasibly matched for the return trip of i, and greater cost savings are generated with k serving as the rider and i as the driver. If there is another participant j, and the only feasible matches are given by arcs (i^1, j), (i^2, k), and (k, i^2), an optimal solution may be to create matches (i^1, j) and (i^2, k) even if $c_{ki^2} > c_{i^2 k}$.

3.3.2.3 Greedy Approach

To gain some understanding of the value of optimization-based approaches in ride-share matching, we will compare the matching and integer programming methods described earlier with a greedy algorithm. The greedy matching algorithm that we propose is a straightforward rule-based approach that a ride-share provider could use to match riders and drivers without requiring more sophisticated optimization software.

The greedy algorithm works as follows. First consider the case where all announcements are either rider or driver requests. Given a set S_A of active announcements, we go over all riders in the order of their announcement time. For each rider announcement r we determine the driver announcement d (if any) that represents a feasible match with the largest possible savings and fix the pair as the assignment for rider announcement r.

For round trip scheduling, we follow the same procedure but only consider riders if they have feasible drivers for both trips and store the average positive savings of the outbound and return matched trip. Finally, for the flexible role case, we use the same procedure but consider each flexible role announcement twice, once as a rider and once as a driver.

3.3.3 Benchmarks

To evaluate the performance of our ride-matching solution approaches, we propose two benchmarks that represent upper bounds on solution quality. For both benchmarks, we solve a so-called *off-line problem* that considers simultaneously the complete set S of announcements received on a particular day. Each off-line problem has advantages over reality, since announcements are essentially known in advance, and thus optimal solutions determined using a technique presented in Section 3.3.2 are upper bounds on the quality of the matches determined sequentially in time using the same technique

within the rolling horizon approach.

The two benchmarks are determined as follows. For the *a posteriori* benchmark, a driver-rider match is only considered feasible in the off-line problem if, in addition to the time feasibility and positive cost savings described earlier, the announcements could possibly be considered simultaneously within some set S_A in a rolling horizon approach, *i.e.*, if there is some overlap between the intervals between the announcement times and the implied latest departure times. The *static* benchmark provides a weaker bound, and drops this requirement for overlap; this benchmark essentially emulates a case where all participants announced their trips in advance on the day prior to traveling.

For instances in which riders and drivers announce fixed roles, each of the off-line optimization problems can be solved in reasonable computational times using CPLEX. However, when instances contain large numbers of announcements with flexible roles, it is difficult to solve the off-line problems to optimality and we therefore determine only a very good (but not provably optimal) solution using an iterative rounding procedure. In this procedure, we first solve the linear programming relaxation of the integer program, then fix certain variables x_{ij} to zero, and finally solve this restricted integer program. Specifically, we fix all outgoing arcs from the node representing participant i to zero if in the linear programming relaxation solution $\sum_j x_{ji} - \sum_j x_{ij} > 0$; this restricts this participant from being assigned as a rider, since the relaxed solution prefers him as a driver.

3.4 Numerical Experiments

We implemented the ride-share matching solution approaches detailed earlier and a simulation environment in *C++*, using CPLEX 11.1 as the linear and binary integer programming solver running on a quad-core 2.66GHz Xeon E5430 with 32GB RAM. We now detail the simulation study and its results.

3.4.1 Simulation Environment

To test the viability of dynamic ride-sharing and to study the merits of optimization for ride-share matching, we developed a simulation environment that considers work trips made in the Atlanta metropolitan region, in the U.S. state of Georgia. The Atlanta area represents a potentially interesting environment for ride-sharing since it does not have good public transport infrastructure and its freeway traffic congestion is among the most severe in the U.S. Also, many major U.S. metropolitan areas have similar urban forms, with low population density and many commercial employment hubs outside of the downtown core. It also represents a challenging test case due to its large size and the large number of automobile work trips. Dynamic ride-sharing concepts that work in Atlanta should also be likely to work in more densely populated urban environments, and perhaps more effectively.

The simulation environment is based on the 2008 travel demand model for the metropolitan Atlanta region, developed by the Atlanta Regional Commission (ARC). The ARC is the regional planning and intergovernmental coordination agency for the 10-county Atlanta area, a sprawling region with a population of approximately 5 million people occupying 6,500 square miles. The travel demand model for the region is used to generate estimates of the daily home-based work-related vehicle trips between all 2024 travel analysis zones (TAZs) within the region (see Table 2). For travel distances and times, we compute airline distances between TAZ population centroids and assume a constant average vehicle speed of 30 miles per hour. Thus, we approximate the true travel distances and times in the Atlanta region, and ignore any time-dependency in travel time caused by congestion. We also ignore any time expending during pick up or drop off of riders. We do not believe that these simplifications have a major impact on our conclusions.

We generate 5 random streams of trips for use within our simulations as follows. Each travel analysis zone is considered a possible origin and destination for

Table 2: Home-based Work Travel Information (ARC, 2008)

daily # round trips	2.96 million
daily vehicle-miles	32 million
avg. trip distance	10.8 mile
low occupancy trips	2.55 million
# o-d pairs	2.9 million
max trips per o-d	881
min trips per o-d	0.01

trips. For each origin-destination pair, we calculate an expected number of daily trip announcements by multiplying the average number of single-occupancy home-based work vehicle trips with a fixed percentage of vehicle-trips that we assume might consider participating in dynamic ride-sharing (the *participation rate*). Then, for each pair, we determine the number of trip announcements using a Poisson random variable with expected value equal to the computed expected number of trips. Each trip announcement is equally likely to be a rider announcement or a driver announcement, when roles are not flexible. Once an outbound trip announcement is generated from a to b, we assume that a return trip from b to a will occur and generate it also.

Trip timing information is also not available in the travel demand model data set. Therefore, we construct the time windows for each announcement as follows. For the outbound trip from home to work, we draw the latest departure time from a normal distribution with mean 7:30 a.m. and standard deviation 1 hour to model a typical morning peak [47], and calculate the latest arrival time by adding the direct travel distance to the latest departure time. Subsequently, we calculate the earliest departure time by subtracting a fixed time flexibility value from the latest departure time. Furthermore, the announcement time is calculated by subtracting an announcement lead time value from the earliest departure time. For the return trip from work to home, we draw a work day length value from a normal distribution with mean 9 hours

and standard deviation 0.5 hour. To construct the time window for the return trip, we add the work day length to the earliest departure time and the latest arrival time of the initial trip.

In all experiments, unless specifically stated otherwise, we generate 5 different random trip announcement streams based on a 2% participation rate, a 30 minute announcement lead-time, and a time-flexibility of 20 minutes. Each stream represents a sample day. As commonly seen in practice (see for example the system operated by zebigo.com), we specify the flexibility as an absolute value rather than a value relative to the duration of the trip; a relative flexibility, *e.g.*, 25% of trip duration, will likely underestimate the flexibility for short trips and overestimate it for longer trips. We will also use a standard re-optimization frequency of 10 minutes within the rolling horizon solution strategy, commencing the first optimization run 10 minutes after the first announcement arrival each day. Importantly, we assume that if participants are notified of a feasible ride-share arrangement, they will always accept it. It would not be too difficult to extend this research to attempt to model the accept/reject behavior of potential participants, but we have chosen to ignore this idea in this initial study.

3.4.2 Base Case Computational Results

We now provide computational results for a base case in which participants are assumed to announce their intended roles in advance, and in which all announcements are for round trips. We consider three different participation rate levels: 1%, 2%, and 4%. For each scenario, we assess the value of the optimization-based approaches for ride-share matching by comparing the quality of the solutions found by the greedy algorithm (denoted GREEDY) and the bipartite matching with bundle constraints binary integer programming approach (denoted BIPART). Each rolling horizon solution is furthermore compared to the two off-line solution quality benchmarks.

We compute the following statistics to compare the different solution approaches,

where the averages are computed over the 5 separate announcement streams:

1. *average success rate* (S): matched trip announcements divided by the number of trip announcements;

2. *average total system-wide vehicle miles savings* (M): miles saved for all announced trips versus if all individual trips were executed unmatched; and

3. *average individual cost savings per trip* (C): costs are assumed to be proportional to vehicle-miles driven, and cost savings are divided proportionally between driver and rider based on the lengths of their original trips.

Note that since we consider single-rider, single-driver ride-share matches only, $S/2$ corresponds to the percentage reduction in the number of vehicle trips among the population of announced trips.

Table 3: Base Case Solution Quality Comparison

	S (%)	M (%)	C (%)
—1%—			
GREEDY	28.2	10.5	26.2
BIPART	58.3	18.3	25.2
a posteriori	60.3	19.9	26.3
static	62.2	20.8	26.8
—2%—			
GREEDY	28.7	11.4	27.4
BIPART	67.0	22.3	27.3
a posteriori	68.7	23.8	28.3
static	70.3	24.6	28.6
—4%—			
GREEDY	28.3	12.2	29.0
BIPART	74.5	26.6	29.6
a posteriori	75.8	28.0	30.5
static	77.1	28.8	31.0

Table 3 demonstrates clearly that BIPART significantly outperforms GREEDY in terms of success rate ($28 - 36\%$) and vehicle-miles savings ($14 - 18\%$) over all three

participation rate levels. The greedy algorithm seems reasonable, but it does not yield good results in practice. Not surprisingly, the greedy approach generates good individual cost savings. It seems clear, however, that it is much more important to maximize the number of matches than the quality of the individual matches, and the integer programming technique does a much better job in this regard. Optimization-based approaches clearly appear to have much potential value in ride-share matching application. Both methods are fast and can solve even the very large off-line problems within a minute of computer time; the largest off-line problem with approximately 29,000 announcements required 78 seconds of computer time for BIPART.

Comparison to the *a posteriori* bound suggests that the rolling horizon approach is close to optimal for practical instances. This is not unexpected, since the trips of drivers and riders that can be feasibly and cost-effectively matched often have departure times that are close together and thus are likely to be considered in the same optimization run. The gap between the *a posteriori* bound and the rolling horizon approach decreases with the announcement density. A potential reason for this is that a higher announcement density leads to more feasible and cost-effective matches, thereby making the cost of committing to a less than optimal match smaller. The *static* benchmark demonstrates the further potential improvement possible given advance information from participants. If trips are announced further in advance of departure, this may allow the ride-share provider to establish matches that would otherwise be missed because compatible trips may not have been announced before the expiration time. For example, a compatible return ride may have not yet been announced by the latest departure time of the initial trip of the rider. A more rare example would be a rider who has not announced by the implied latest departure time of the driver if they were to be matched, *i.e.*, if the travel time between the driver's origin and the rider's origin is greater than the rider's announcement lead-time.

The results also demonstrate that increasing the participation rate leads to a

higher success rate, and also improves the average individual savings. That is, not only does the relative fraction of participants that find a ride-share increase, but also the individual savings from sharing the trip costs. This result quantifies the importance of density for ride-sharing, which of course is well known. Note also, however, that the relative advantage of BIPART increases with the participation rate. Thus, the optimization-based procedure provides additional advantage over simpler strategies when it considers more options during a run.

Since travel cost is assumed proportional to the travel distance, the reported system-wide vehicle-miles savings correspond to cost savings. Assuming an average per-mile direct cost of \$0.54 [3], we see daily cost savings in these scenarios that range from approximately \$27,000 (1% participation) to \$152,000 (4% participation). Even the revenue from a small fraction of these savings may provide an interesting business opportunity for a private ride-share provider. For a participation rate of 2%, the average individual savings for the matched participants is approximately \$1.90 per trip (\$3.80 per round trip) which may provide sufficient incentive for participants (who may already be motivated by travel time savings in carpool lanes or concerns about the environment). Note also that the average additional in-vehicle travel time for the drivers ranges from 5.8 minutes for the 1% participation rate to 5.2 minutes for the 4% participation rate, which seems to be an acceptably small increase according to the findings of previous ride-sharing surveys [43].

Next, we consider some additional characteristics of the solutions by examining the individual origin-destination distances of each driver-rider match. In Figure 5, we see that the rider's trip distance is typically smaller than the driver's original trip distance in a match; 78% of the matches lie below the diagonal where the driver's trip and rider's trip have the same length. This is not unexpected, since if the rider's trip is larger than the driver's trip, the additional driving distance required to accommodate the rider reduces potential distance savings for the pair. Recall that a match between

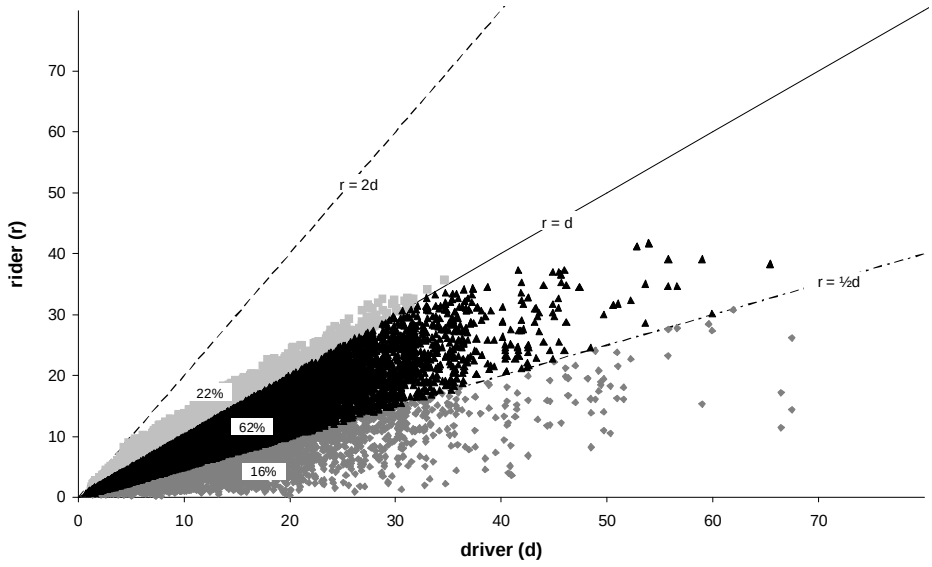

Figure 5: Original Trip Distances for Matched Participants

rider r and driver d only produces cost savings if $d_{v(d),v(r)} + d_{w(r),w(d)} < d_{v(d),w(d)}$. There is no possibility for cost savings if the length $d_{v(r),w(r)}$ of the rider's trip is more than twice the distance $d_{v(d),w(d)}$ of the driver's, which further implies that the total driving required of a driver in a ride-share match cannot exceed twice $d_{v(d),w(d)}$.

Matches in which the rider has the longer trip distance (above the diagonal in Figure 5) generally involve participants with smaller individual trip distances. The driver's time flexibility makes matches between participants with longer trips less likely. Moreover, we see relatively few matches where the rider trips are significantly shorter than the matched driver trips. To understand this, note that maximizing vehicle-mile savings coincides with maximizing the travel distance when both participants are traveling together. Thus, more savings are possible if a driver can travel with a rider who is traveling further.

Figure 6 depicts the success rate of for announced trips of different lengths, where

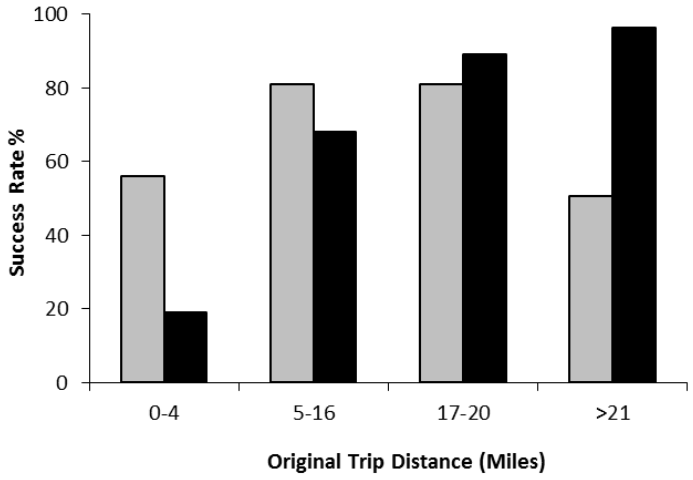

Figure 6: Success Rates for Riders (gray) and Drivers (black) by Original Trip Distance

each bucket represents roughly 25% of the daily announcements. For the driver trips, we see that the likelihood of a match increases with the length of the trip, again since longer trips correspond to more potential savings and also result in a higher likelihood of finding a compatible rider on the way. For the rider trips, we observe a trade-off between feasibility and savings with respect to trip length. Although shorter trips may easily find compatible drivers, they also represent smaller potential savings. Longer trips, on the other hand, may represent more savings but are also harder to match.

Next we focus on the likelihood of getting matched for announcements with different earliest departure times. Figure 7 shows that the highest success rates occur during the morning rush period (6 a.m. to 9 a.m.) and the evening rush period (3 p.m. to 6 p.m.). This is intuitive because these times have the highest announcement densities in our scenarios. A nice feature of dynamic ride-sharing, then, is that the high concentration of trips that leads to negative system impacts like congestion also

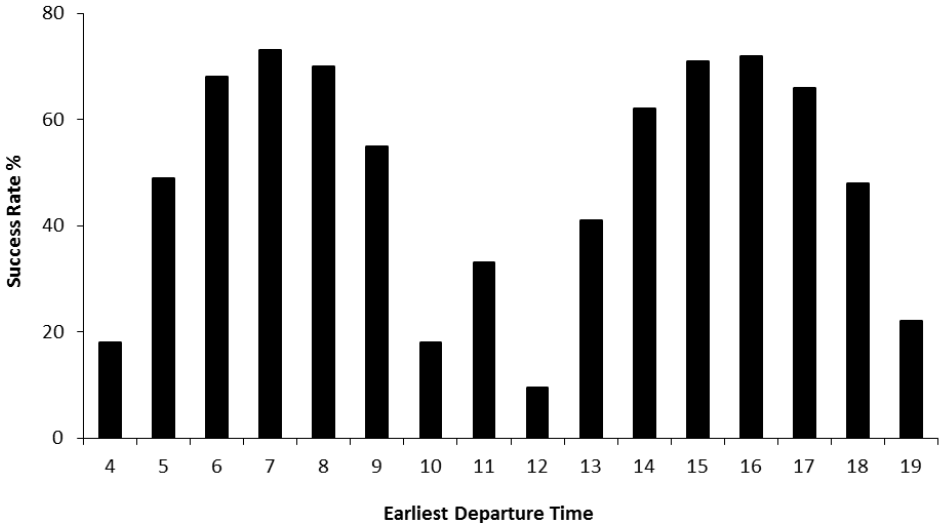

Figure 7: Success Rate by Time of Day

leads to positive impacts on the performance of ride-sharing systems.

Finally, we consider the rolling horizon strategies in more detail by examining the impact on solution quality by changing the re-optimization timing and the commitment strategy. The strategy that re-optimizes after each minute coincides with a strategy that runs an optimization each time a new announcement is made. Recall that our base case assumption is that the potential ride-share matches found via optimization are not finalized until as late as possible. Here, we also examine an alternative strategy where all proposed matches are finalized immediately after the optimization run in which they were identified.

Table 4 presents the results for the 2% participation rate announcement streams. The results demonstrate that for our test scenario assumptions regarding announcement lead time and time flexibility, systems that employ the latest commitment strategy for matches should be optimized more frequently. However, if we commit matches immediately, we observe that there are advantages of optimizing less frequently since

Table 4: Rolling Horizon Strategy Comparison

	S (%)	M (%)	C (%)
Latest commitment			
BIPART 1 min	68.5	22.9	21.9
BIPART 5 min	67.3	22.5	27.3
*BIPART 10 min	67.0	22.3	27.3
BIPART 30 min	65.5	21.2	26.6
Immediate commitment			
BIPART 1 min	62.6	14.3	15.7
BIPART 5 min	62.4	15.6	21.2
BIPART 10 min	63.0	16.9	22.5
BIPART 30 min	64.3	19.8	25.4
* base case			

it allows the accumulation of more trip announcements between optimization runs. Although not depicted in these results, it should be clear that this benefit of optimizing less frequently will eventually reverse itself. When the time between optimization runs grows too large, missed matching opportunities become more and more prevalent. For a simple example, consider a rider who announces a trip at 8:01 and a driver who expires at 8:07 (but announced before 8:01). This driver-rider match may be missed when the time between re-optimization runs is greater than 6 minutes, *e.g.*, if optimizing at 8:00 and 8:10.

3.4.3 The Advantages of Flexible Roles

The previous results assume that all participants announce trips with fixed roles, as drivers or riders. Here, we focus on the other extreme where every participant is flexible to serve as a driver or a rider for his announced trip. In this case, the optimization problem considered during each optimization run cannot necessarily be solved to optimality quickly. Therefore, we configure the optimization with two stopping criteria: a maximum solution time limit of 200 seconds, or a feasible solution found that has an objective function value guaranteed to be no worse than 1% smaller

than the optimal value (also known as 1% optimality gap in integer programming). Note then that it is possible that no feasible solution is found within the time limit; in this case, we use as the solution the proposed matches found in the previous optimization run. This time limit is not imposed when computing the *a posteriori* benchmarks, but since the problems are very difficult to solve we apply the iterative rounding procedure described earlier to find a very good feasible solution; the final integer program after variable fixing is solved to a 5% optimality gap. Since the *a posteriori* benchmark problem is not solved to provable optimality in this case, we also record the solution of its linear programming relaxation to provide an upper bound on potential cost savings.

Table 5: Ride-sharing with Flexible Roles

	S (%)	M (%)	C (%)
GREEDY	45.8	19.3	28.3
IP	85.4	31.4	30.0
a posteriori	85.6	33.6	32.1
LP-relaxation	87.0	34.3	

Table 5 summarizes results for the 2% participation rate announcement streams and shows that role flexibility yields substantial improvements: an absolute increase of approximately 15% on the success rate, and 10% on vehicle-miles savings. As in the earlier fixed role case, the optimization-based approach (denoted IP) performs much better than the greedy heuristic. However, the individual optimization problems are much harder and more time-consuming to solve. In our study, the integer programming software finds at least one integer feasible solution for each of the optimization runs for each of the 5 announcement streams within the 200 second time limit. In 15% of the runs, the time limit expires before the 1% optimality gap is attained; for these runs, the maximum gap observed was 2.9%. Note that again the rolling horizon aggregate solution has total quality not much smaller than the best integer solution

found for the *a posteriori* benchmark problem. Furthermore, the best integer solutions found for the *a posteriori* problems are quite close to the linear programming upper bound, indicating that the benchmarks are quite good and that the iterative rounding procedure is useful for solving these very large flexible role instances.

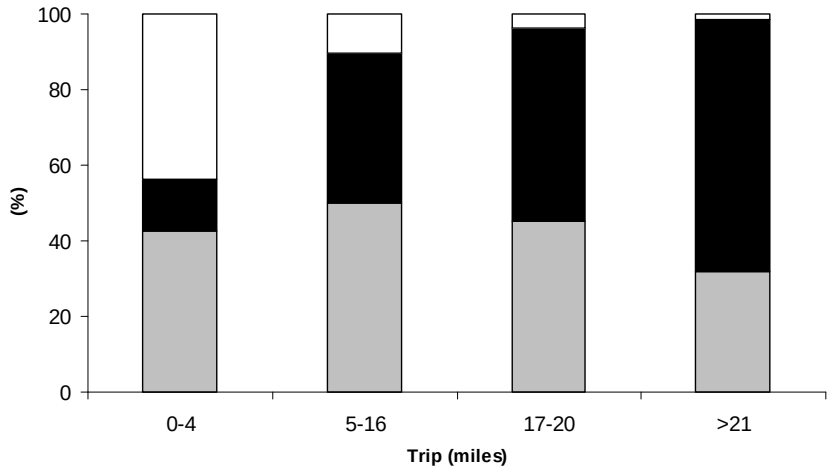

Figure 8: Matching Results for Flexible Roles Scenarios by Original Trip Length: Matched as Rider (gray), Matched as Driver (black), Not Matched (white)

Figure 8 demonstrates how flexible role problems solved using the optimization-based approach are able to find matches for most trips with longer distances. The figure breaks out trip announcements in distance buckets into three subsets: matched as rider, matched as driver, and not matched. We see that the longest trips have the highest success rate and the shorter trips have the smallest success rate. This is intuitive since ride-share matches between longer trips lead to greater vehicle-mile savings. As expected, a relatively larger number of the longer trip announcements are matched up as drivers. However, not all long (short) trips are drivers (riders) because in fact the ride-share matches that produce the largest savings involve participants with very similar trip lengths, often traveling from the same origin region to same destination region.

3.4.4 Single Trip Ride-sharing

In the experiments described earlier, we assume that all trip announcements are for round trips, and that both the outbound and return trip timing are known with certainty when announced. However, for certain round trips, it may be difficult for participants to specify the time of their return trip, and they may prefer to announce both trips separately on short notice.

To understand the system impacts that result when participants attempt to arrange their trips separately, we conduct an experiment where we consider the same 5 announcement streams for the 2% participation rate, only now return trips are announced 30 minutes before their earliest departure time instead of together with the outbound trips. Drivers are assumed to always announce two trips, but riders will not announce a return trip if they did not share a ride on their outbound trip. To prevent unmatched trip requests, we also consider using a different objective function for the optimization problems solved here, maximizing the total number of system matches instead of total system travel distance savings.

For this experiment, we compute the success rate (S) by considering the percentage of *riders* that were matched for rides on both their outbound and return trip. Moreover, we compute the percentage of *riders* (S^-) that were matched outbound, but failed to be matched on their return trip. The results are presented in Table 6. Notably, for both the round-trip announcement cases (BIPART-JOINT) and the separate announcement cases (BIPART-SEP), the objective of maximizing the number of matches rather than savings can increase the matching success rate by $4-8\%$ with only small degradation of the total vehicle-miles savings ($< 1\%$) and per-match cost savings ($3-4\%$).

Separate trip announcements without a return guarantee increase the vehicle-miles savings for both cases and success rate when maximizing matches. However,

Table 6: Maximize Savings versus Maximizing Matches

	S (%)	S^-(%)	M (%)	C (%)
maximize savings				
*BIPART-JOINT	67.0	-	22.3	27.3
BIPART-SEP	65.2	10.0	24.5	29.0
maximize matches				
BIPART-JOINT	71.1	-	21.7	25.0
BIPART-SEP	73.0	5.3	23.6	25.4
* base case				

the additional flexibility creates a risk for each rider of failing to find a return ride-share match. Not surprisingly, maximizing the number of matches seems to mitigate this risk, *i.e.*, 5.3% of the riders without a return ride compared to 10% when savings are maximized. Furthermore, it is also possible to build optimization approaches that attempt to maximize total cost savings while prioritizing matching riders that are completing round trips; of course, the risk of not finding a match for a "stranded" rider still remains. Whether such risk is acceptable depends on the situation, in particular on the availability of inexpensive alternatives such as public transport. To allow guaranteed return trips without the corresponding round trip restrictions, the ride-share provider may utilize back-up drivers, *e.g.*, by cooperating with urban commercial taxis.

3.4.5 Fixing Ride-share Pairs on Round Trips

Traditional carpooling typically involves a long-term commitment among at least two people to share rides to work on some or all of their weekly workdays. The lack of travel flexibility afforded by carpooling is often quoted as one of the major reasons people are hesitant to participate in carpooling [65]. Furthermore, irregular working hours also hinder traditional carpooling, since it may be more difficult to find compatible time schedules [26].

Dynamic ride-sharing is more flexible because daily trips can be arranged separately without requiring the same driver-rider pairs on different trips or on different days. To attempt to quantify some of the flexibility benefit of dynamic ride-sharing versus traditional carpooling, we consider a slightly less flexible ride-share scenario that requires a rider to be matched with the same driver on both his outbound and return trip on a specific day. Note that this scenario is more flexible than traditional carpooling, because it still allows different matches across days. We also choose to conduct this study using the assumptions of the static benchmark problem, where all trip announcements are known prior to the beginning of the day, and assume that announcements have fixed roles.

For this experiment, we will also vary the variability of participant departure times to understand its impact on the value of the flexibility of dynamic ride-sharing. To do so, we consider a set of scenarios in which we increase the standard deviation of morning departure time and the standard deviation of the workday duration both by 50%, and another set of scenarios where both deviations where we decrease these deviations by 50%.

Note that when we only consider ride-share matches in which the driver for each matched rider is the same on the outbound and return trips, we introduce symmetry to the optimization problem since the vehicle-mile savings on the outbound trip are equal to the savings on the inbound trip. This optimization problem can be represented using a maximum weight bipartite matching model with one node for each round-trip announcement, and an arc from a rider announcement i to a driver announcement j if both the outbound and return trip matches are feasible, with weight c_{ij} equal to twice the cost savings generated by the outbound match.

Table 7 summarizes the results of this experiment, where the lines labeled "fixed pairs" assume that riders are matched both on outbound and return trip with the same driver, while the lines labeled "flexible pairs" relax this assumption (as in the earlier

Table 7: Fixed Ride-share Pairs

	S (%)	M (%)	C (%)
fixed pairs	57.6	18.4	25.4
flexible pairs	68.7	23.8	27.3
time variability +50%			
fixed pairs	47.4	14.0	23.0
flexible pairs	65.7	22.1	27.2
time variability -50%			
fixed pairs	71.6	25.4	29.3
flexible pairs	77.1	28.6	30.9

results). Flexible pairings substantially increase the solution quality: the success rate is increased by about 10% in absolute terms, and the cost savings by about 4-5%. As expected, the benefit of flexible pairs increases with the variability of the departure times of the participants. This is because one can always keep the same ride-share pairs on both trips if the participants spend roughly the same amount of time at work between the two trips. In the absence of any time variability, of course, the flexible and fixed pairs case would yield the same solution. Since many information economy workers no longer have rigid work schedules, the flexibility benefits provided by dynamic ride-sharing over traditional carpooling are quite important to consider.

3.4.6 Varying the Participants' Flexibility

Ride-sharing asks for time sacrifices, especially from the drivers. In addition, participants may have to be somewhat flexible in their departure times to find a ride-share match. The individual benefits in terms of travel cost savings provided by ride-sharing should counterbalance these inconveniences. Therefore, financial gains less than a specified threshold may not be acceptable for participants. Moreover, more certainty regarding the potential savings may motivate participants to be more flexible in their departure times. In this experiment, we evaluate the impact of the participant's time

flexibility and a cost savings threshold on the performance of the system. The cost savings threshold (denoted τ) represents the minimum acceptable cost savings per feasible ride-share match.

The results are shown in Table 8 for the base case strategy and a 2% participation rate. As expected, there are system and individual benefits created by additional time flexibility. Furthermore, the marginal benefits decrease time flexibility increases. We observe that an increase in the cost savings threshold has a stronger negative impact on the success rate than on the system-wide vehicle miles savings. Surprisingly, in one scenario with a time flexibility of 10 minutes, setting a small threshold ($1) even leads to an increase in vehicle miles savings. In this case, the system appears less likely to commit to a match with very small cost savings while better matching opportunities are available at a later point in time. Overall, the results suggest that more time flexibility allows participants to set a higher cost savings threshold with limited impact on the performance of the system when measured by savings in vehicle miles.

Table 8: Participants' Time Flexibility and Cost Savings Threshold

	S (%)	M (%)	C (%)
τ = $0			
10 min	47.9	13.4	23.9
20 min	67.0	22.3	27.3
30 min	73.7	26.3	28.9
τ = $1			
10 min	39.8	13.5	22.0
20 min	56.7	22.0	24.1
30 min	62.8	25.9	25.3
τ = $2			
10 min	30.3	12.5	24.2
20 min	45.9	20.8	25.7
30 min	51.6	24.6	26.6

3.5 How to Achieve Critical Mass?

The experiments presented in this proposal have shown the importance of sufficient numbers of announcing participants to enable dynamic ride-share matches to be established on short notice in practice. In the startup phase of a dynamic ride-share system, it may be difficult to attract enough participants to generate good matches, and this will likely lead many potential participants to give up on the system. In this section, we attempt to develop a reasonable model for an adoption pattern of dynamic ride-sharing over time, and to determine whether dynamic ride-sharing systems may be successfully initiated and sustained.

To model the adoption of dynamic ride-sharing, we draw upon the very large body of marketing literature on the diffusion of new products and technology. The most widely accepted diffusion model is the Bass diffusion model [46]. The model assumes that the probability that an initial purchase will be made is a linear function of the number of previous buyers [9]. Due to interpersonal communications (*e.g.*, word-of-mouth), potential adopters are more likely to become aware of a certain product or service if the number of users increases. The probability $\frac{k(t)}{1-K(t)}$ of adoption, *i.e.*, starts placing announcements at time t is $p + \frac{q}{m}Y(t)$, with $k(t)$ representing the individual probability of adoption at time t and $K(t)$ its cumulative form and $Y(t)$ the total number of adopters up to time t. The constant parameters m, p, and q represent the total number of potential adopters, a coefficient of innovation and a coefficient of imitation respectively. While the coefficient of innovation represents the exogenous likelihood that a new participant joins the system, the coefficient of imitation relates to the increase in this likelihood with the number of participants that are already in the system.

While the diffusion model allows us to forecast how many new participants join the system, we also want to consider the announcement behavior of the existing participants over time. Conceptually, we may assume that participants are satisfied

if they are matched in ride-shares, and thus continue to announce trips regularly. Participants that do not receive ride-share matches may become discouraged and stop announcing new trips. To model this behavior, we assume that a participant i receives one additional positive goodwill credit s_i from each successful ride-share match, and one negative credit f_i each time a trip is announced and is not matched. As long as his net credit is positive ($s_i - f_i + g > 0$), we assume that the participant will continue to announce his ride-share trips, where we define g to be the starting goodwill credit of the participant. Once goodwill is depleted to zero, the participant never announces again. We recognize that this set of assumptions creates a system that, if simulated over very long time horizons, will eventually include no possible participants. However, we believe that the model is useful for examining system behavior over relatively short time periods. To examine longer time periods, it would not be difficult to extend the model to allow new potential adopters to enter the system over time, for example, representing new members of the labor force entering the pool of commuters.

In the following experiments, we follow the behavior of a hypothetical system for Atlanta over a two month period after startup for different diffusion parameters. Each day in the study period includes a set of round-trip announcements with fixed roles, and is solved using the rolling horizon optimization approach. Unless stated otherwise, we assume the total number of potential trip announcements (m) to be 4% of the total number of home-based trips and a goodwill g of 5. First, we determine a set of potential participant round-trips using the methods described earlier. For each potential participant, we draw a base latest departure time from a normal distribution with a mean of 7:30 a.m. and a standard deviation of 1 hour (see Section 3.4). For each subsequent day, we draw the latest departure of each active participant again using a normal distribution with his base departure time as the mean and 15 minutes as the standard deviation.

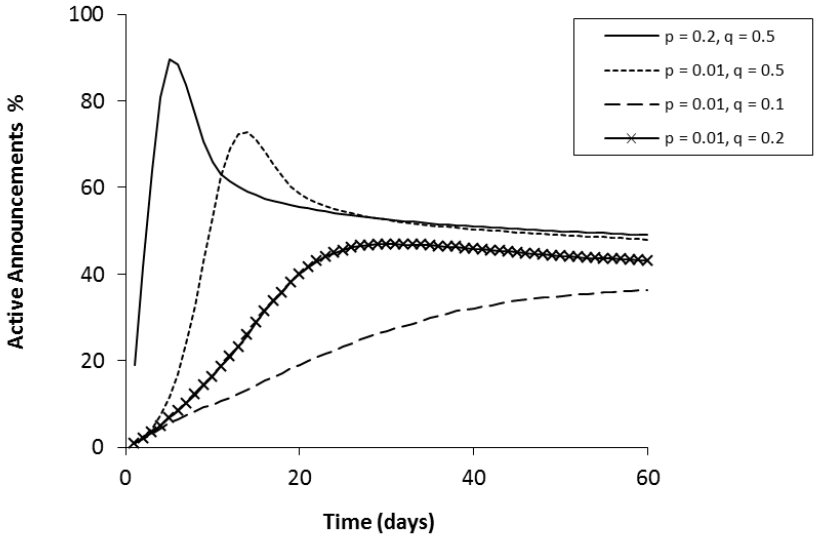

Figure 9: Ride-sharing System Sustainability for Various Diffusion Patterns

The results of these experiments are summarized in Figure 9, where the fraction of active participant announcements is plotted over time. The plots demonstrate that when the sum of the innovation and imitation rates is sufficiently high (*i.e.*, > 0.5), the system seems to converge to a steady active announcement stream in two to three weeks. Approximately 55% of the total potential trip announcements remain active, and the success rate converges to approximately 85% of announced trips. The results show that even when the total potential pool of participants is limited to a small fraction (4%) of the total home-based work round-trips, dynamic ride-sharing may still be sustainable. Participants in corridors amenable to ride-sharing will likely continue to announce given the high match rate; in this way, the ride-sharing system at least in this experiment has configured itself.

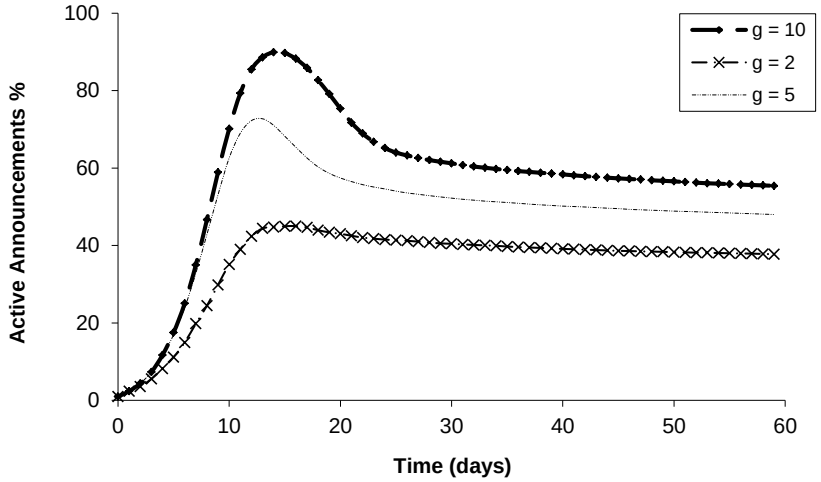

Figure 10: Sustainability of Ride-sharing Systems for Different Levels of Initial Participant Goodwill

In Figure 10, we see that the initial goodwill possessed by potential participants has a significant impact on the success and sustainability of dynamic ride-sharing systems. It seems particularly important in the startup phase that potential participants continue to place announcements even though they are not matched. It seems highly likely, therefore, that public incentives might be necessary to initiate a dynamic ride-sharing system. If participants are discouraged by not finding matches when the participant density is low, it may be quite difficult to build a sustainable participant community.

3.6 Conclusions

In this study of dynamic ride-sharing, we have seen that the use of sophisticated optimization methods substantially increases the likelihood that ride-share matches can be found for participants, and leads to ride-sharing systems that generate larger overall system travel cost savings. Furthermore, our simulation studies have shown that dynamic ride-sharing may have potential for success in large U.S. metropolitan areas, with sustainable ride-share populations forming over time even with relatively small overall participation rates and when considering only home-based work trips.

Besides travel costs savings, ride-sharing systems may provide travel time savings to participants by providing access to high occupancy lanes. Moreover, ride-sharing may help to decrease traffic congestion and thereby reduce system-wide travel times. We believe that extending ride-sharing simulation models to explicitly consider time-dependent and occupancy-dependent travel times provides a valuable area of future research.

CHAPTER IV

SINGLE-RIDER, SINGLE-DRIVER STABLE (AND NEARLY-STABLE) RIDE-SHARE MATCHING

4.1 Introduction

In Chapter 3, a fundamental assumption is that after a match-making system decides on good rider-driver matches, each participant accepts his match. This is obviously optimistic, especially since individual system participants may have motivations for participating in the system that do not completely align with system-level objectives. In this chapter, we start to more closely consider the acceptance choices of participants by attempting to build *stable* (or nearly-stable) sets of ride-share matches. We define a set of stable matches to be one where no rider and driver would prefer to be matched together for a ride-share trip than to their current matches (or, to remain unmatched). This notion of stability is analogous to that defined in the well-known stable marriage problem, which will be defined later in this chapter.

For exposition, suppose that each potential participant in a ride-sharing system has complete visibility of all other participants and all of their relevant information. Figure 2, then, graphically depicts a scenario where a system-optimal solution is not stable. A system optimal solution is to assign r_1 to d_1 and r_2 to d_2. However, r_2 and d_1 would both prefer to be matched together than to their current partners. By sharing a ride, each of these two participants would pay for the equivalent of 4.5 miles of travel if the total benefit of the match (3 miles of savings) were split equally between them. In the current matches, however, both r_2 and d_1 pay for 5 miles of travel.

A class of so-called *two-sided matching problems* potentially useful for problems

such as these was introduced by Gale and Shapley [27]. Two-sided matching problems differ from the classical assignment problem (*i.e.*, jobs are assigned to workers) since there are participants on both sides of the assignment who care about the outcome and have the power to reject or abandon the solution proposed by a central planner. These problems explicitly constrain the solutions found by a central planner so that groups of matched participants do not have incentive to make private arrangements. To do so, so-called "stability" constraints are included in formulations in addition to standard matching constraints. A discussion of the basic theory of two-sided matching is presented in [58]. In this chapter, we will use some of these ideas to introduce stability constraints into models for dynamic ride-share matching.

The remainder of this chapter is structured as follows. In Section 4.2, we introduce a related classical problem: the stable marriage problem. In Section 4.3, we define the maximum weight stable (and nearly-stable) matching problems for dynamic ride-sharing. We then design models and solution algorithms specifically for the single-rider, single-driver matching problem, considering the case where participants have fixed driver/rider roles and without round trip announcements. In Section 4.4, we model the participants accept/reject behavior in an unstable ride-sharing system and study the system performance over time. Finally, in Section 4.5, we make some concluding remarks.

4.2 Stable Marriage Problem

The stable marriage problem is a classical bipartite matching problem. The original setting considers two sets of men and women of equal size n. Each person has ranked all members of the opposite sex with a strict preference number from 1 to n, where a lower number indicates a stronger preference. The objective is to create a perfect matching of men and women such that there does not exist any pair of man and woman who prefer each other to their current partners. If man m and woman w

forms such a pair, they are called a *blocking pair*. If there are no blocking pairs in the matching solution, we call it a stable marriage assignment or stable matching.

It is shown that every instance of the stable marriage problem has a stable matching which can be found in $O(n^2)$ time [27]. Vande Vate [66] initiated the study of the stable marriage problem using a mathematical programming approach. A complete characterization of the convex hull of the stable marriage solution polytope is established. Rothblum [59] extended the polyhedral description to the case where the bipartite graph is not complete, and additionally considers scenarios where some participants would rather remain single than be matched to some participants. A simpler proof of the primary result in [66] is provided by Roth et al. [57].

When there may exist ties in preference lists, there are three stability notions: super-stability, strong stability, and weak stability. We introduce some notation before giving the definitions of these three notions. Let M be a set of men, given by $M = \{m_1, m_2, ..., m_p\}$. Let W be a set of women, given by $W = \{w_1, w_2, ..., m_q\}$. A proposed pair (m, w) in $M \times W$ is *acceptable* if m and w prefer each other to remaining single. Let A denote the set of acceptable pairs. We now define a matching:

Definition 4.2.1. Matching in Stable Marriage Problems

A *matching* is a one-to-one mapping μ from $M \cup W$ to itself, such that:

1. $\mu(m) = w$ if and only if $\mu(w) = m$, in which case m is matched to w.

2. If $\mu(m)$ is not in W, then $\mu(m) = m$, in which case m is unmatched.

3. If $\mu(w)$ is not in M, then $\mu(w) = w$, in which case w is unmatched.

We can also define some notation for preferences. The form $a >_c b$ denotes that person c prefers person a to b, and $a \geq_c b$ denotes that either $a >_c b$ or that person c is indifferent in preference toward a and b. A matching μ is called *individually rational* if no participant a prefers being single (*i.e.*, unmatched) to $\mu(a)$. An individually

rational matching is further defined to be *stable* if there is no pair $(m, w) \in A$ where $w >_m \mu(m)$ and $m >_w \mu(w)$.

Now we can define the three stability notions for problems in which the number of men p equals the number of women q. Such problems will be referred to as balanced. First, we define a stable matching in terms of blocking pairs:

Definition 4.2.2. Stable Matching

A matching μ is a *stable matching* if it contains no blocking pairs.

Definition 4.2.3. Super Stability in Balanced Problems

A stable matching is *super-stable* if a blocking pair is defined as a pair $(m, w) \in M \times W$ where $\mu(m) \neq w$, $w \geq_m \mu(m)$, and $m \geq_w \mu(w)$.

Definition 4.2.4. Strong Stability in Balanced Problems

A stable matching is *strongly-stable* if a blocking pair is defined as a pair $(x, y) \in (M \times W) \cup (W \times M)$ where $\mu(x) \neq y$, $y >_x \mu(x)$, and $x \geq_y \mu(y)$.

Definition 4.2.5. Weak Stability in Balanced Problems

A stable matching is *weakly-stable* if a blocking pair is defined as a pair $(m, w) \in M \times W$ where $\mu(m) \neq w$, $w >_m \mu(m)$, and $m >_w \mu(w)$.

For balanced stable marriage problems with potential ties in preference lists, it is not hard to see that a weakly-stable solution always exists and can be found using the well-known Gale-Shapley Algorithm. To do so, simply create a strict preference order for each participant by breaking ties arbitrarily, and apply the algorithm. Any stable matching found via this approach is weakly stable given the original (non-strict) preference lists. In contrast, there are instances that have no super-stable nor strongly-stable matching.

In real world problems, there are not always equal sizes of the bipartite sets. We can extend the stability concept accordingly.

Definition 4.2.6. Weak Stability in Unbalanced Problems

A stable matching is *weakly-stable* in unbalanced problems if a blocking pair is defined as a pair $(m, w) \in M \times W$ where $\mu(m) \neq w$, m is either unmatched or $w >_m \mu(m)$, and w is either unmatched or $m >_w \mu(w)$.

It can be shown that for unbalanced problems, there is at least one stable matching in which all the elements of the smaller set are matched. The Gale-Shapley algorithm can be applied to find a stable matching with unequal sets with some modifications.

Considering the polyhedral characteristics of stable marriage problem, it is somewhat surprising to notice that the set of stable matchings can be represented as the extreme points of the set of solutions of a simple system of linear constraints. Let M and W denote the sets of men and women as defined above. The incidence vector of a matching μ is a vector $x \in \{0,1\}^{|M|\times|W|}$ such that $x_{m,w} = 1$ if $\mu(m) = w$ and $x_{m,w} = 0$, otherwise. In the formulation, the notation $j >_m w$ is used to denote $\{j \in W : j >_m w\}$ and $i >_w m$ is used to denote $\{i \in M : i >_w m\}$. With this notation, we can characterize the stable matchings by their configurations as follows.

Theorem 4.2.1. (Roth et al. [57]) *A vector $x \in \mathbb{R}^{|M|\times|W|}$ is a stable matching if and only if it is an integer solution of*

$$\sum_{j\in W} x_{m,j} \leq 1 \qquad \forall\ m \in M \tag{1}$$

$$\sum_{i\in M} x_{i,w} \leq 1 \qquad \forall\ w \in W \tag{2}$$

$$x_{m,w} \geq 0 \qquad \forall\ (m,w) \in M \times W \tag{3}$$

$$x_{m,w} = 0 \qquad \forall\ (m,w) \in (M \times W) \setminus A \tag{4}$$

$$\sum_{j>_m w} x_{m,j} + \sum_{i>_w m} x_{i,w} + x_{m,w} \geq 1 \qquad \forall\ (m,w) \in (M \times W) \in A \tag{5}$$

Constraints (1), (2), and (3) represent matching constraints. Constraints (4) are called individual rationality constraints. Constraints (5) define the stability constraints. They ensure that for each acceptable pair (m, w), either man m marries

someone he prefers to woman w, or w marries someone she prefers to m, or m and w marry each other. Stability constraints prevent (weakly-stable) blocking pairs to appear. Constraints (1)-(5) define a set of stable fractional matchings. Next, we want to see that the extreme points of the linear constraints (1)-(5) correspond precisely to the stable matchings.

It can be shown that the constraint set (1)-(5) has integer-valued extreme points. An extreme point of a convex set C is a point $x \in C$ such that there exists no $y \in C$ and $z \in C$, both different from x, and $x = \alpha y + (1 - \alpha)z$ for some $0 < \alpha < 1$.

Theorem 4.2.2. (Roth et al. [57]) *Let C be the convex polyhedron of solutions to the linear constraints (1)-(5). Then the integer points of C are precisely its extreme points. That is, the extreme points of the linear constraints (1)-(5) correspond precisely to the stable matchings.*

In Section 4.3, we will discuss how to apply the ideas of stable marriage problem to the dynamic ride-sharing problem and tailor it to the specific characteristics found in the ride-sharing context. We will focus on the ride-sharing setting where single riders are matched with single drivers for one-way trips and roles are fixed for the entirety of the stability analysis.

4.3 Stable Matching Problem in Ride-Sharing

It is intuitive to relate the stable marriage problem to the stability notion in ride-sharing matching problems. In our ride-sharing problem, we have two disjoint sets of riders and drivers. Assume that they have information about the other participants and they know the total savings generated by sharing a ride with any other participant. Each participant has his/her own evaluation of the benefit created by sharing a ride with a certain partner. We assume that a participant generates his preference list based on the savings realized by pairing with each participant on the other side. Higher savings yield stronger preference. For example, r_1 can be matched with d_1, d_2

and d_3. The respective savings for r_1 are 3, 1, and 5. Then the preference list for r_1 is $(2, 1, 3)$. If there are ties in the preference list, that is, when a rider/diver can see more than one feasible potential partner that creates the same benefit, we apply weak stability only for the following analysis. Now we need to consider how to divide the savings between the participants in each pair. We make the cost division assumption that each participant in a ride-sharing pair receives half the total savings generated by this pair. This follows the Shapley value in cooperative game theory. Without participation in the game with two persons, no one can earn anything. Instead, as long as one participates, he/she can benefit.

Our analysis focuses on the scenario with home-work trips only and in which each participant fixes his/her role as a rider or a driver when sending an announcement. With return trips, the problem becomes more complicated since we need to consider unstable pairs in both trips, so we do not consider the round trip scenario. In the analysis that follows, we assume that all participants have complete system-level information. Thus, if rider i and driver j are match feasible (time feasible with positive cost savings), then rider i and driver j know about each other and will put each other in their respective ranked preference lists. Also, we assume that the complete set of announcements received for a particular day are considered simultaneously. Thus, the analysis herein will be most similar to the *a posteriori* case in Chapter 3.

4.3.1 Basic Problem

We now introduce the primary stable ride-sharing matching problem that we will consider in this chapter. This basic problem extends the matching problem in Chapter 3, maximizing total system vehicle mile savings by creating stable rider-driver matches. We call this problem the max weight stable matching problem ($MWSM$) and the formulation is defined as follows:

4.3.1.1 Basic Formulation

$$\text{Maximize} \quad \sum_{i,j \in A} c_{i,j} x_{i,j}$$

$$\text{subject to} \quad \sum_{j \in D} x_{i,j} \leq 1 \quad \forall i \in R \tag{6}$$

$$\sum_{i \in R} x_{i,j} \leq 1 \quad \forall j \in D \tag{7}$$

$$\sum_{j' \geq_i j} x_{i,j'} + \sum_{i' \geq_j i} x_{i',j} + x_{i,j} \geq 1 \quad \forall (i,j) \in A \tag{8}$$

$$x_{i,j} \in \{0,1\}, \quad \forall (i,j) \in A \tag{9}$$

Compared to the stable marriage formulation, we omit the individual rationality constraints (4). The set of acceptable pairs A is simply the set of feasible arcs in our bipartite graph. In the ride-sharing setting, there could exist ties in arc weights so we modify the stability constraints (8) to find a weakly stable matching. A minimum weight stable matching problem ($MinWSM$) can be found by changing the objective function to minimizing total vehicle mile savings.

We now present computational results for the basic maximum weight stable matching formulation and compare it with the maximum weight matching and minimum weight stable matching solutions. Again, we built a simulation exvironment that considers work trips in Atlanta. Please refer to Section 3.4 for a detailed description. Various participant rate levels are studied. For each level, we compute the following statistics: average success rate (S), average total system-wide vehicle miles savings (M), average individual cost savings per trip (C) in relative form (%) and absolute form ($|\cdot|$). $|C|$ stands for the average individual savings in dollars based on an average per-mile direct cost of \$0.54 [3]. The average here is over 5 different announcement streams.

Table 9 shows that introducing stability constraints reduces the success rate and vehicle-miles savings. However $MWSM$ gives a better individual cost savings. This is reasonable since matched participants have relatively good personal savings in stable

Table 9: Stable Matching Solution Quality Comparison

	S (%)	M (%)	C (%)	|C|
—0.5%—				
MWM	57.5	18.4	27.1	1.8
$MWSM$	54.9	17.7	27.3	1.8
$MinWSM$	54.9	17.7	27.3	1.8
—1%—				
MWM	66.0	22.5	28.9	1.9
$MWSM$	62.7	21.5	29.1	2.0
$MinWSM$	62.6	21.5	29.1	2.0
—2%—				
MWM	73.0	26.3	30.7	2.1
$MWSM$	69.2	25.1	31.1	2.1
$MinWSM$	69.1	25.1	31.1	2.1
—4%—				
MWM	78.3	29.7	32.6	2.2
$MWSM$	74.4	28.4	32.9	2.2
$MinWSM$	74.2	28.3	32.9	2.2
—5%—				
MWM	80.0	30.8	33.1	2.2
$MWSM$	76.1	29.4	33.5	2.2
$MinWSM$	75.9	29.4	33.5	2.2

matching solutions and this would lead to fewer matched participants and worse overall system total savings. Increased participation rate yields a higher success rate, system total savings and personal savings. The difference between $MWSM$ and $MinWSM$ is very small. This shows stable matching solutions yield similar total vehicle-miles savings, regardless of the objective function. This observation may be quite useful, since heuristics may be able to find feasible stable solutions quickly.

To better quantify the difference of MWM from $MWSM$ solutions, we calculate relative instability gaps (denote $Gap(MWM, MWSM)$ for each of the performance metrics). The relative gap from b to a is calculated as $(b-a)/a \times 100\%$. The results are presented in Table 10. The results show that $MWSM$ does not reduce the success rate and system-wide vehicle miles savings by more than 5%. Personal savings in $MWSM$ are no better than 2% in relative form and not better than 1% in absolute form. This shows that $MWSM$ provides a solution that is very close to the one from MWM. When varying participant rates, the absolute change values in all statistics

increase from 0.5% to 2%, and then decrease from 2% to 5%.

Table 10: Relative Gaps Between $MWSM$ and MWM Solutions For Various Performance Metrics

	S (%)	M (%)	C (%)	|C|
0.5%	-4.6%	-3.9%	0.7%	0.7%
1%	-5.0%	-4.4%	0.8%	0.6%
2%	-5.2%	-4.5%	1.2%	0.7%
4%	-5.0%	-4.4%	1.2%	0.7%
5%	-4.9%	-4.3%	1.2%	0.6%

In the worst case, the instability gap in vehicle mile savings, could be as large as 50% degradation from the MWM objective. An example is shown in Figure 11. The MWM solution assigns b and c in a match, while in $MWSM$ algorithm, a and b are in a match and c and d are matched. In this case, the relative instability gap in system vehicle mile savings is close to 50%.

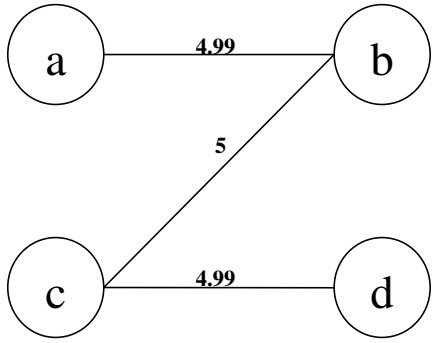

Figure 11: An Example Instance with an Instability Gap in System Vehicle Miles Savings Close to 50%

To further undertstand the stability of an MWM solution, we calculate two statistics. The first one is the proportion of matched participants who are in at least one blocking pair, and we denote it as $B1$. It is calculated as the number of unique participants in blocking pairs divided by the total number of participants who are assigned

a match. The second one is the average number of blocking pairs per participant in a blocking pair, and it is denoted as $B2$. It is obtained by dividing 2 times the number of blocking pairs by the number of unique participants in blocking pairs and it measures the blocking choice density for a blocking participant. Table 11 presents $B1$ and $B2$ values for various participant rates. The computational results are averaged over 5 simulation runs. The results demonstrate that approximately one fifth of the participants who are assigned a match are in at least one blocking pair in the max weight matching solution. This proportion does not look small. Nearly 20% matched participants may therefore have an incentive to reject the match provided by the system. The blocking choice density for each blocking participant is not high, however, does not exceed 2. This shows that on average, each participant who is involved in blocking pair(s) has no more than 2 better alternative candidates than the system's assignment.

Table 11: Blocking Pairs in Max Weight Matching

	0.5%	1%	2%	4%	5%
B1 (%)	15	19	21	23	23
B2 (%)	1.2	1.3	1.4	1.4	1.4

To understand the graph structure of the matching problems that are solved for these ride-sharing applications, we present the distribution of feasible matches in a relative frequency histogram in Figure 12. It records the number of feasible matches for each participant. A participation rate of 2% is used and the results are averaged over 5 announcement streams. We observe that each participant has a very limited number of feasible matches. With a participation rate of 2%, there are more than 10,000 drivers and riders respectively in the system, but few participants have more than 300 feasible matches. More than 40% of the participants have no more than 10 feasible options, and 28% participants have no more than 5 feasible options. Fewer than 20% of participants have more than 50 options. To explain this, remember

that to establish a feasible match, we need to satisfy two conditions: time feasible and positive cost savings. People are more likely to be matched with those who announce with similar departure times. Also, in order to create positive savings, the driver's original trip distance has to be greater than the sum of distance between driver and rider origins and the distance between driver and rider destinations. Both factors clearly limit the number of feasible matches for either a driver or a rider. On the other hand, the number of feasible matches seems to be clearly sufficient to provide alternative candidates for victims creating when constraints are introduced to prevent blocking pairs, as in the MWSM formulation. This is evidenced by the fact that we obtain a less than 5% instability gap in vehicle miles savings even though around 20% of the participants are in blocking pairs when constraints are not used to prevent them, as seen in Table 11.

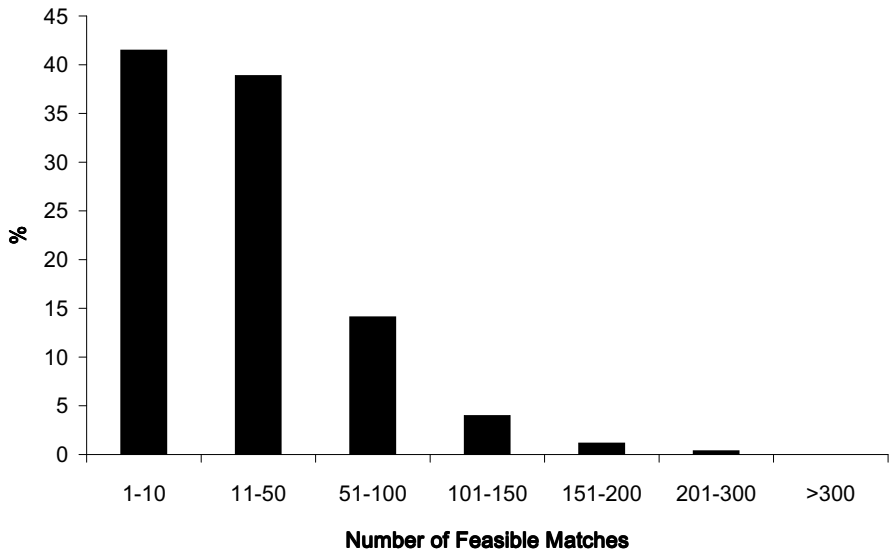

Figure 12: Distribution of Feasible Matches in A Posteriori Ride-Share Matching Problems

Next we study the impact on stable solution quality by changing the time flexibility for participants. As a reminder, a time flexibility specifies the difference between the earliest departure time and latest departure time for an announcement. By default, the time flexibility is set as 20 minutes. The instability gaps between MWM and $MWSM$ in S, M, C and $|C|$ are calculated; note that the objective is to maximize vehicle miles savings. Computational results for a 0.5% participant rate are shown in Table 12. In all the cases, the system's performance in success rate S and total vehicle mile savings M are very close. Slightly better individual savings are realized with the $MWSM$ model. With less time flexibility, participants are less likely to have many feasible matches due to the strict time constraints. The number of blocking pairs is likely to be limited in this case; thus the instability gap stays small. With greater time flexibility, the number of blocking pairs increases and a larger instability gap is obtained. But, when the time flexibility becomes very large, there are many feasible matches for each participant. More feasible choices provide more options to form a stable matching with good total savings and the instability gap could decrease.

Table 12: Instability Gaps Between $MWSM$ and MWM Solutions under Different Time Flexibility

time flex	S (%)	M (%)	C (%)	|C|
5 min	-3.3%	-2.3%	0.6%	1.0%
10 min	-4.5%	-3.7%	0.2%	0.7%
20 min	-4.6%	-3.9%	0.7%	0.7%
50 min	-4.4%	-4.0%	0.7%	0.4%
100 min	-4.5%	-4.0%	1.2%	0.5%
200 min	-4.2%	-3.7%	1.0%	0.6%

It seems that varying the time flexibility of participants does not substantially impact in the instability gap. We further consider relaxing the time feasibility constraints totally, in which case a feasible match is established if and only if matching the pair produces positive savings. In such a setting, only geographical information decides the feasible matches. We associate a probability $p \in (0, 1]$ with each feasible

match to decide the likelihood for this match to appear. Table 13 presents the instability gap in S, M, C and $|C|$ with a 0.5% participant rate. With different density of feasible matches, a similar pattern is observed as in Table 12. Instability gap in all the four statistics stays small. This shows that the small instability gap is probably independent of the timing information but due to the geographical property of our ride-sharing settings.

Table 13: MWSM Degradation without Time Feasibility Constraints

p	S (%)	M (%)	C (%)	|C|
—0.2—	-4.3%	-3.8%	1.1%	0.5%
—0.4—	-4.4%	-3.9%	1.0%	0.4%
—0.6—	-4.6%	-3.8%	1.4%	0.8%
—0.8—	-4.4%	-3.8%	1.1%	0.7%
—1.0—	-4.2%	-3.7%	1.1%	0.5%

To further investigate participants in blocking pairs, we study the incentives from leaving the system's assignment for each participant in any blocking pair. For each participant in any potential blocking pair, we calculate the saving gains needed to prevent the blocking pair from leaving the current system solution. The distribution of distance savings in vehicle-miles for 2% participant rate is plotted in Figure 13. It seems that most of the participants in blocking pairs have vehicle miles saving gains in the low range of the histogram. Specifically, nearly 50% saving gains lie in the range of no more than 0.5 miles. More than 70% blocking options have no more than 1 mile in savings. From Table 9, we know that matched participants can earn around 2 dollars on average for 2% partcipant rate. Assuming an average per-mile direct cost of \$0.54 [3], saving 0.5 miles brings around 25 cents of savings. This is approximately one eighth of the average savings and people may not consider it attractive. Thus, it is useful to study the case when a relaxed stable matching solution is considered and we solve a nearly-stable matching problem. Note also that tiny incentives to leave the system's assignment would lead to big degradation in system total savings as seen in

Figure 11. However, we observe a degradation in less than 5%. This is probably due to the fact that participants often have enough feasible candidates as backups. In the example in Figure 11, while b and c establish a matching pair, a and d are typically matched with someone else instead of being left alone.

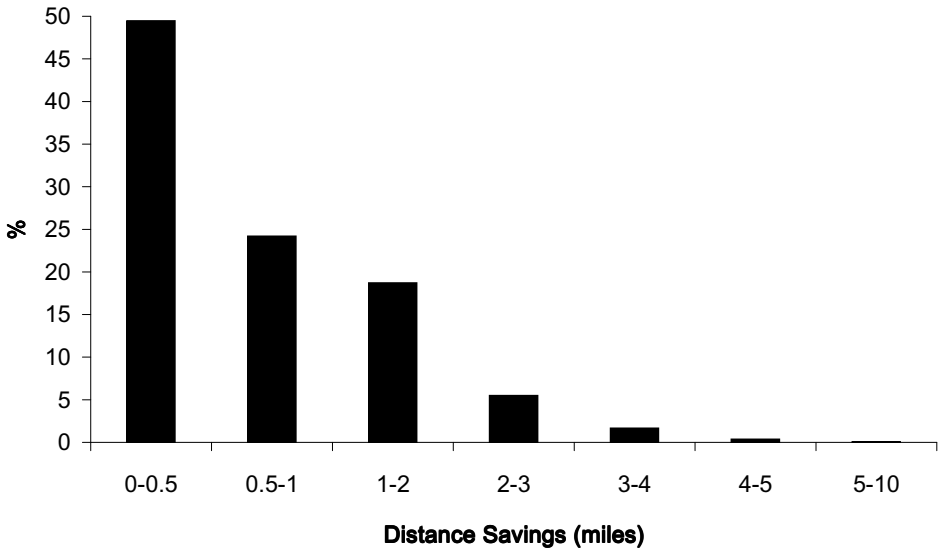

Figure 13: Benefits from Forming a Blocking Pair

4.3.2 Nearly-stable Problem

Adding stability constraints which prevent all possible blocking pairs is an extreme approach to introduce stability into ride-sharing matching formulations. It is natural to think of a method that could balance the total system benefit with the stability of the solution, which is a proxy for how acceptable the solution may be to participants.

The first method is to relax the objective to reach optimality. Instead of requiring solutions which provide the maximum total vehicle miles savings, an inferior near optimal objective value may be acceptable which minimizes also some measure of the

instability. The second method is to relax the stability constraints, and redefine the blocking pairs that the formulation prevents from existing in an optimal solution.

4.3.2.1 Near-optimal Least Unstable Formulation

In this section, we describe an approach for finding nearly-stable solutions by bounding the maximum objective function deviation from that of an unstable optimal solution. A two stage method is employed. In stage one, we solve the standard max weight matching problem and denote the objective value as O. In stage two, we minimize the total penalty $\sum_{(i \in R, j \in D)} y_{i,j}$ and set a parameter $\beta \in [0, 1]$ for controlling the maximum deviation that we allow from the unstable optimal objective function. The term y_{ij} will penalize the creation of a blocking pair for participants i and j.

The complete algorithm (denoted $MWNSM1$) is as follows:

Stage 1: Solve max weight matching problem, denote the objective value as O.

Stage 2: Solve the following integer programming problem:

$$\begin{aligned}
\text{Minimize} \quad & \sum_{i \in R, j \in D} y_{i,j} \\
\text{subject to} \quad & \sum_{j \in D} x_{i,j} \leq 1 \quad \forall i \in R && (10) \\
& \sum_{i \in R} x_{i,j} \leq 1 \quad \forall j \in D && (11) \\
& y_{i,j} \geq 1 - (\sum_{j' \geq_i j} x_{i,j'} + \sum_{i' \geq_j i} x_{i',j} + x_{i,j}) \quad \forall (i,j) \in A && (12) \\
& \sum_{i \in R, j \in D} c_{i,j} x_{i,j} \geq \beta O && (13) \\
& x_{i,j}, y_{i,j} \in \{0, 1\}, \quad \forall (i,j) \in (R \times D), \beta \in [0, 1] && (14)
\end{aligned}$$

The objective function aims to minimize total number of blocking pairs. Constraints (10) and (11) are matching constraints. Constraints (12) measure the violation of stability constraints. If i and j form a blocking pair, $y_{i,j}$ is enforced to be 1. Constraints (13) controls the lower bound of total weight of the matching solution.

It can not degrade more than (1-β) from the maximum weight. If β is 1, we want to maintain unstable optimality. We are solving a stable matching problem when β is set to 0.

Table 14: Max Weight Nearly Stable Matching $MWNSM1$ Blocking Pairs Statistics

β	1	0.99
♯ blocking pairs	2381	726
♯ participants in blocking pairs	3519	1326
B1 (%)	20	8

Table 14 presents the results with $\beta = 1$ and 0.99 for a participant rate of 2%. The results are averaged over 5 announcement streams. We can see that by compromising 1% of the optimal objective value, the number of participants in blocking pairs decreases more than 60% and the number of blocking pairs decreases even more (up to 70%) due to the fact that a participant could be in more than one pair. The proportion of participants in blocking pairs also decreases from 20% to 8% compared to the total number of matched participants. This experiment appears to indicate that most of the blocking pairs do not greatly degrade the system's total savings. Note also that the average optimal system-wide vehicle-miles savings over 5 simulation runs is approximately 66,450 miles. By 1% miles savings, we can remove about 2000 participants from blocking pairs. Thus, it is not very costly in terms of system objective function degradation to create a much more stable system.

4.3.2.2 Relax stability constraints

As indicated in Figure 13, most of the participants in blocking pairs gain less than 1 mile in savings from leaving the system's assignment and matching instead with their partner in a blocking pair. Therefore, it may be useful to consider an alternative model in which blocking pairs are redefined based on some notion of what savings gains can be reasonably perceived by participants. Instead of preventing any blocking

pairs to appear by stability constraints (8), we could allow the existence in a solution of some blocking pairs where the participants would not benefit much from leaving the system's assignments.

The modified the formulation (denoted $MWNSM2$) is described as follows:

$$\text{Maximize} \quad \sum_{i \in R, j \in D} c_{i,j} x_{i,j}$$

$$\text{subject to} \quad \sum_{j \in D} x_{i,j} \leq 1 \quad \forall i \in R \tag{15}$$

$$\sum_{i \in R} x_{i,j} \leq 1 \quad \forall j \in D \tag{16}$$

$$\sum_{j' \succeq_i^{\epsilon} j} x_{i,j'} + \sum_{i' \succeq_j^{\epsilon} i} x_{i',j} + x_{i,j} \geq 1 \quad \forall (i,j) \in A \tag{17}$$

$$x_{i,j} \in \{0,1\}, \quad \forall (i,j) \in (R \times D), \epsilon \geq 0 \tag{18}$$

Compared with $MWSM$ formulation, the difference lies in the definition of the stability constraints. Constraints (17) are called near-stability constraints and a tolerance value ϵ is added. They ensure that for each acceptable pair (i, j), either rider i is matched with someone with miles savings not ϵ worse than driver j, or driver j is matched with someone with miles savings not ϵ worse than rider i, or rider i and driver j are matched. Note that when ϵ goes to infinity, we create a MWM optimal solution and when ϵ is 0, we solve a $MWSM$ problem.

Table 15 presents the number of blocking pairs, the number of parcitipants in blocking pairs and the relative instability gap in system total vehicle miles savings. A participant rate of 2% is used and the results are averaged over 5 announcement streams. We set ϵ as 0.5 and 1.0 miles seperately. When ϵ is 0, we are solving a max weight stable matching problem. There is no blocking pair and the instability gap is -4.5%. When ϵ is infinity, we are solving a max weight matching problem in which case we generate the maximum number of blocking pairs and an instability gap of zero. By allowing 0.5 miles tolerance, more than 70% blocking pairs and participants

in blocking pairs appear in the system, and we achieve nearly 90% optimality in the system total vehicle-miles savings. If people only perceive a blocking pair when the saving is larger than 1 mile, then we create more than 90% of the total blocking pairs and the system total savings nearly reaches the max weight matching optimum. To interpret these results, note that we can say the system is essentially stable if participants cannot perceive blocking pairs with partners that generate only 1 mile of additional savings (beyond the system match). And, since only approximately 700 of the 2400 blocking pairs remain when participants cannot perceive blocking partners that yield only 0.5 miles of savings, using this parameter setting generates a nearly stable solution with very little total degradation in the objective value.

Table 15: Max Weight Nearly Stable Matching $MWNSM2$ Blocking Pairs Statistics

ϵ	0	0.5	1.0	infinity
♯ blocking pairs	0	1698	2197	2381
♯ participants in blocking pairs	0	2748	3338	3519
Gap(MWM, MWSM)	-4.5%	-1.2%	-0.3%	0

4.3.3 Greedy Algorithm

In this section, we show that a classical greedy heuristic method actually provides a stable matching solution. The greedy algorithm is a common heuristic to find a matching solution in a bipartite graph. It works as follows. Given a set S_A of active announcements, we determine for each rider announcement r the driver announcement d (if any) that represents a feasible match with the largest possible savings. Among all of these matches, we then select (r_m, d_m) with the largest savings and fix it. Requests r_m and d_m are then removed from S_A, and the process is repeated until no positive savings matches remain.

We claim that the greedy algorithm finds a stable matching solution. To understand this, assume we have a greedy matching solution s' which is not stable. Then

there must exist a blocking pair $(b1, b2)$. Suppose $b1$ is currently assigned to $b2'$ and $b2$ is assigned to $b1'$, then weight with pair $(b1, b2')$ and $(b1', b2)$ must be both smaller than that with $(b1, b2)$. This is a contradiction since otherwise $(b1, b2)$ should have been assigned before the other two pairs were assigned. Since the max weight stable matching and the minimum weight stable matching provide very close solutions as shown in Table 9, we do not provide extra computational results for the stable matching solutions provided by the greedy algorithm.

4.4 Simulation in System Dynamics with Unstable Participants

In this section, we consider a system in which the participants are able to reject an assignment from the system if it is not stable. We attempt to build a reasonable ride-sharing system over time and to realistically model participants' accept/reject interaction. As the participation days accumulate, a user could collect knowledge about who would be a good candidate to ride with based on previous ride-sharing experience. We study how a ride-sharing system evolves over time in this setting, where we assume that participants who have met before (in a ride-share) may leave the system matching if they are in blocking pairs.

For each participant i, we maintain a backup list. Whenever participant j successfully shares a ride with participant i, we put j into i's backup list. These are the participants that i knows. Now, each time i gets an assignment from the system, he/she will check his/her backup list to decide whether to accept or reject it. If i has a better candidate from his backup list who is time feasible and prefers i as well, the system's assignment is turned down.

Consider then a system that operates as follows:

- Participants make announcements.
- The system makes assignments based on max weight matching algorithm and

sends the assignments to participants involved.

- Each participant, in order of his announcement time, checks his backup list to decide whether to accept or to reject the system's assignment based on Algorithm 1.

Algorithm 1 Participants Accept/Reject Behavior Simulation

```
Sort all announcements in announcement time ascending order
Denote the set of the announcements as N_a. Announcement time of a is denoted
as T(a). If i < j, then T(i) < T(j), where 1 ≤ i, j ≤ |N_a|.
for i = 1 to |N_a| do
  if There exists a partner j assigned to i then
    if j is available then
      Check backup list of i
      if There exists b in backup list better than j then
        Finalize match i and b, mark them as unavailable
      else
        Check backup list of j
        if There does not exist b in the backup list where j prefers b to i then
          Finalize match i and j, update backup lists of i and j
        end if
      end if
    else
      Check backup list of i
      if There exists b in backup list who has i better than his current partner
      then
        Finalize match i and b, mark them as unavailable
      end if
    end if
  else
    Check backup list of i
    if There exists b in backup list who has i better than his current partner then
      Finalize match i and b, mark them as unavailable
    end if
  end if
end for
```

In Algorithm 1, we maintain a backup list for each participant. When looking through the backup list, we assume that people pick the available candidate providing the best benefits. Note that not all the candidates in the backup list are available.

The reason candidate j in i's backup list may not be available for i could be that the time is not feasible for the pair on the current day or an alternative assignment has already been finalized for j. Even if j is available, the match i and j can be established only when i and j prefer each other over current system's assignment if there is any. Note that Algorithm 1 is a simplification of an exhaustive search algorithm where i and j form a match only when they are both best available from each other's backup lists. This simplified approach matches the first blocking pair during the process of traversing the participant list. This is a reasonable setup since people may not be patient enough to wait for the best candidate, but are probably willing to investigate some alternative options if a ride-matching system creates questionable matches.

In the following experiments, we simulate a system with Atlanta data described in Chapter 3.4 over a two month period. The setup is the same as those in Chapter 4.3. Only the home-work trips are considered and each participant has fixed roles. The whole set of announcements is received before the start of the current day. A participant rate of 1% is used for the following analysis if not otherwise stated. We generate the announcements for consecutive days as follows. First, we determine a set of potential home-work trips using the methods described in Chapter 3.4. For each potential participant, a base latest departure time is drawn from a normal distribution with a mean of 7:30 a.m. and standard deviation of 1 hour. For each following day, a normal distribution with this base departure time as the mean and a standard deviation of 10 minutes is drawn for each participant.

We now provide computational results to understand the system dynamics with participants accept/reject behavior modeled as described. First, we assess the total system savings in miles each day from the max weight matching formulation, the max weight matching with participants accept/reject behavior stated in Algorithm 1 (denoted AR), and the max weight stable matching formulation.

Comparative results are summarized in Figure 14, where the relative system total

savings to the MWM upper bound is plotted over time. For day 1, participants have no candidate in the backup list so they accept the options assigned by the system and we have exactly the same total savings in AR as in MWM. From day 2, with more new people entering the backup lists of participants, people are more likely to reject the system's assignment and this makes the total system savings worsen. After about two weeks, the system with accept/reject behavior seems to reach a steady state. This is probably because people have already met nearly all the possible candidates and the system could be saturated.

Also, it is interesting that the AR line crosses with the line $MWSM$ at days 3 and 4. From day 4, the $MWSM$ line stays above. The explanation is that without enough candidates in backup list in the early days, the system's MWM assignment still dominates candidates in the backup lists and AR provides better total savings than $MWSM$. Note that in $MWSM$ solution, we establish stability over all feasible matches, not only the set of people who know each other based on their knowledge accumulated from previous days. After day 4, worse system savings result from more participants making private arrangements. The average degradation of AR from the MWM savings averaged over 60 days is 7%, while the average degradation for $MWSM$ is 5%.

On each day, it is very unlikely that a participant could have all the feasible candidates in his/her backup list due to the limited way a connection is established: by only successful previous rides. To determine the maximum degradation, then we set up a scenario when the full backup list is considered every day for each participant. That is, each participant has exactly the same visibility as the system for feasible matches. We denote this case as $AR(full)$.

Figure 15 presents relative instability gap of system total savings between accept/reject behavior setup (AR or $AR(full)$) and MWM respectively in 60 days. This is calculated as the difference between the system total savings in accept/reject

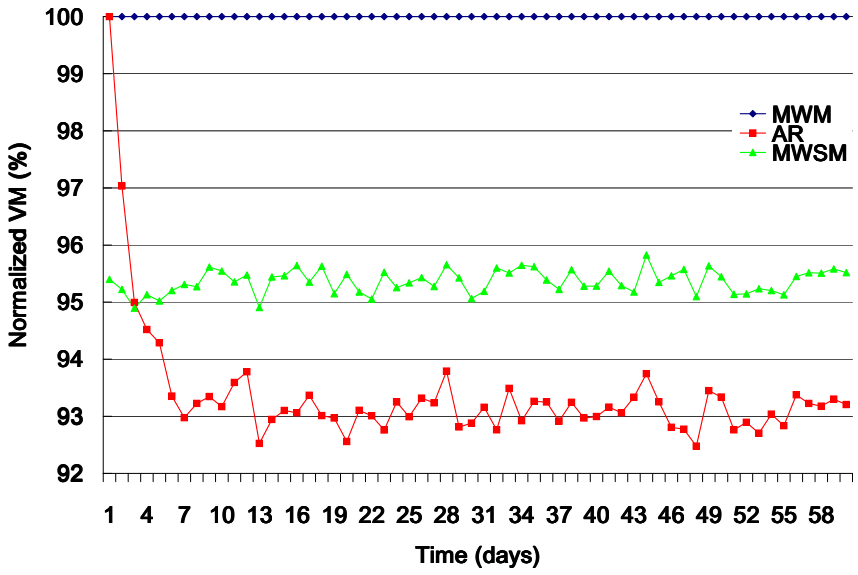

Figure 14: System Savings Dynamics with Accept/Reject Behavior Modeling

scenario (AR or $AR(full)$) and MWM divided by the savings in MWM. The AR line starts from 0 as the backup list accumulates and it is empty at day 1. However the $AR(full)$ line starts from around 8% since the backup list is constructed based on the feasible matches for each day from day 1. The average degradation from MWM for $AR(full)$ is around 8%, which is about 1% worse than AR. It shows that full visibility of feasible matches by participants would lead to worse system performance with regard to the system-wide total savings compared with limited knowledge. This is reasonable since more information about backup candidates is obtained each day, and more participant involvement could lead to a worse system performance in total savings. What is interesting, however, is that even a backup list generated sequentially between participants successfully matched together creates almost as bad degradation as the complete information backup list.

In the experiments above, we set the standard deviation to 10 minutes in the normal distribution from which we draw the latest departure time. To understand the impacts to the system when participants have different time schedules, we conduct an experiment where a range of standard deviation values are used. Denote the standard

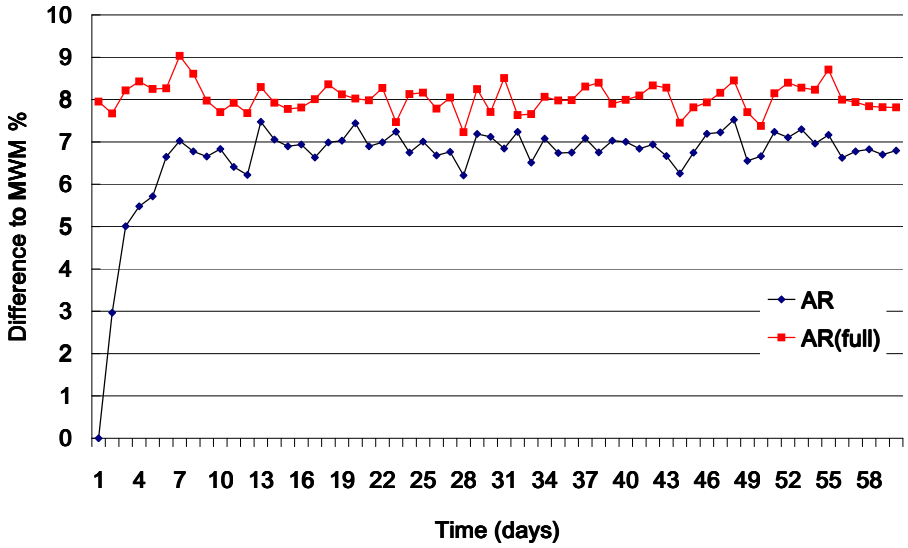

Figure 15: System Savings Dynamics in Accept/Reject Behavior Modeling with Full Backup List

deviation as *std*. We test 4 scenarios where *std* is set to 2, 5, 10 and 30 respectively. The instability gaps are calculated for each day and presented in consecutive 60 days in Figure 16.

Our first observation is the slope differences of the lines at the beginning of the simulation days. The slope shows the system performance degradation speed. A standard deviation of 2 minutes has a smaller slope compared to the case in standard deviation of 5 minutes, which is smaller than that of 10 minutes. However, when the time window size becomes too large where standard deviation is 30 minutes, the slope becomes the smallest among all the four cases. The reason that the system worsens slowly in $std = 2$ case is that with small variation of announcement timing, participants gradually meet other participants. Thus, they are less likely to have enough candidates from their backup list to possibly reject the system's assignment. When people have larger variation in announcement timing ($std = 5$ and $std = 10$), they recognize more new people sooner and the system's MWM solution would more likely be rejected due to the participants' abundant backup candidates. Once the

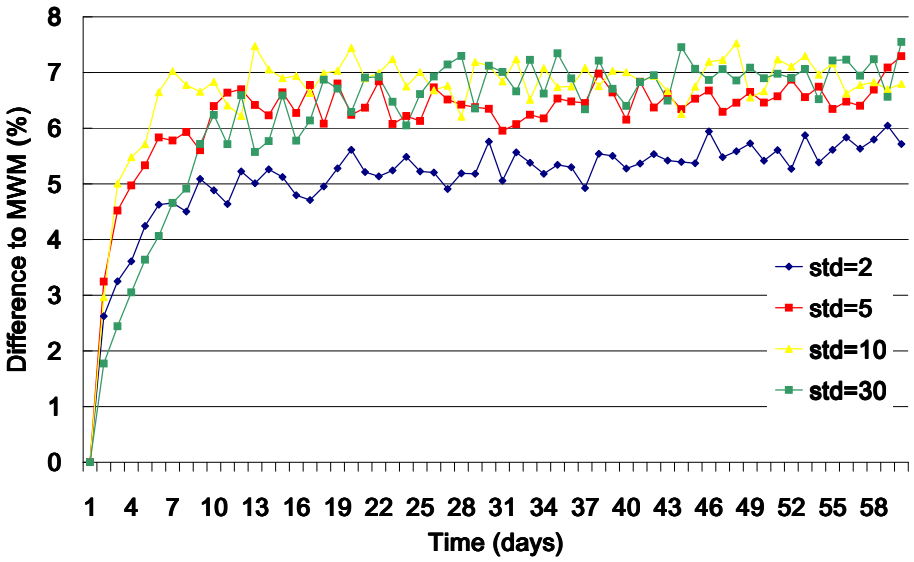

Figure 16: System Savings Dynamics with Accept/Reject Behavior Modeling in Different Timing Variation

time window grows too large, participants can not make use of their backup list since their time schedules jump unpredictably and it is very likely that they miss those candidates due to time infeasibility.

Our second observation is the height of the lines in the time span. With $std = 2$, the line is lower than those in larger standard deviations. This is probably due to the limited candidates met with the small announcement timing variation, which leads to limited damage to the system. With more time flexibility, a worse system total savings could result as people have more of candidates in their backup lists. This shows that if people have rigid time schedules, even if they have the ability to reject the system's assignment when there is a blocking pair, the system's performance can not deteriorate much due to people's limited access to available matching choices.

Participants may not want to reject the system's max weight matching assignment if there is a blocking pair since the saving differences in the two choices might be very close, as seen in Section 4.3.2.2. Also, rejecting the system guaranteed choice would bring extra time and effort costs since people need to find alternative candidates by

themselves. It is reasonable to consider the situation when participants only regard benefits of constructing a blocking pair above a certain threshold as perceivable. We can associate a tolerance value when comparing the savings from the system and the blocking pair options. We define such tolerance in miles denoted as tol. Rider r considers driver $d1$ better than $d2$ if and only if the savings in miles with $d1$ is tol better than the one with $d2$.

Figure 17 presents the $MWSM$ and AR in the case when the tolerance is 0 and 0.5 respectively. By relaxing the blocking pair definition in this way, the system total mile savings are less impacted. Averaged over 60 days, the instability gap to MWM for $MWSM$ decreases to 1% from 5%, and AR decreases from 7% to 3%.

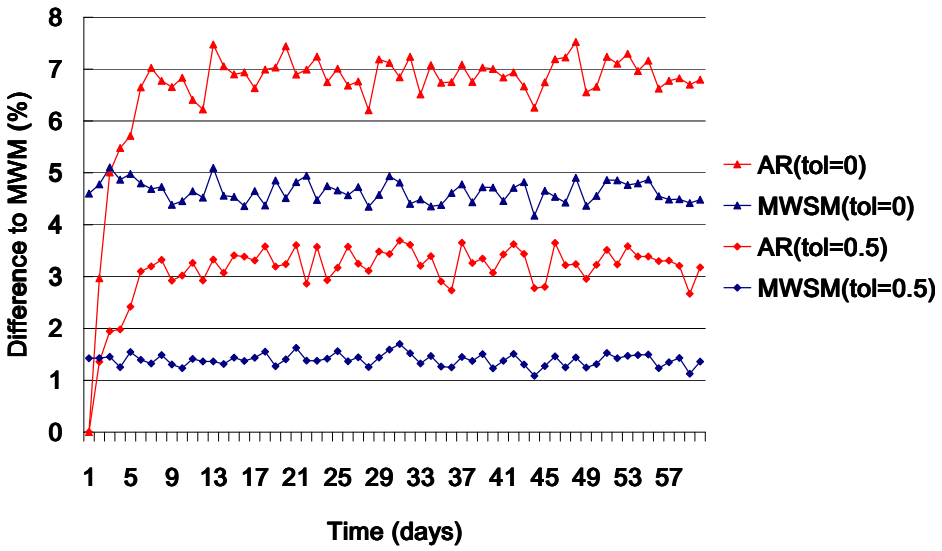

Figure 17: System Savings Comparison with Tolerance in Savings

4.5 Conclusions

In this study of ride-sharing problem, we analyze the stable matching problems based on home-based work trips with fixed rider/driver roles. With the simulation of Atlanta travel data, we observe a noticeable deterioration in ride-sharing system performance with unstable matching assignments. The system total savings could be degraded by

up to 5% with max weight stable matching model in Atlanta travel data simulation. A larger instability gap has not been observed which may due to the geographical structure of our ride-sharing problem. Each participant's feasible matches are constrained within local groups of neighbours. Due to the abundance of local feasible alternatives, blocking pairs would not bring fatal damage to the system's total benefits.

We further study the system dynamics over a number of simulation days with participants who could reject the matching assignments if they are not stable. The system's total benefits deteriorate around 7% from max weight matching optimal objective values, which is worse than the performance provided by the max weight stable matching algorithm. When more feasible match information is revealed to participants, they may make decisions that move the system into an even worse situation. This shows that applying a stable matching algorithm is beneficial to prevent blocking pairs. If people are assigned only stable matchings by the system in which case no private match making would bring more benefits, the ride-sharing system could be more trustworthy and sustainable.

Moreover, we observe that participants' time schedule flexibility plays an important role in influencing the system dynamics. Note also that people may not want to announce everyday and it is interesting to study the system dynamics with respect to the participants' announcing frequency.

With only home-based work trips with fixed rider/driver roles, it seems not costly to find a stable matching solution. When considering work-home trips and participants' flexibility in roles, the problem could become more complex. We believe that incorporating round trip constraints and flexible participants' roles into the stable matching setting could be an interesting future research area.

CHAPTER V

MECHANISM DESIGN IN CHOICE ASSIGNMENT

5.1 Introduction

In this chapter, we consider another set of sophisticated ride-share settings where participants are not assumed to accept each match to which they are assigned. In such settings, we present users with a menu of possible ride-share matches from which they can choose. This is basically the idea of mechanism design, i.e., designing the operational rules of the system to induce desirable behaviour from individual participants even if they follow selfish motives. The choices in the menu presented to potential matching participants are different in vehicle-miles savings. The objective of the system decision maker is to maximize the system's total vehicle-miles savings while each user wants to maximize their own savings when making a choice. In order to simulate participants' reaction to a menu of matching choices, two scenarios are considered. One is to assume that each participant chooses the option with highest vehicle-miles savings. In this scenario, people only care about the savings they can realize. The other one is to associate a probability for each option from the menu. The assumption is that people also consider other facts to establish a match besides economic benefits. We derive mathematical models and test the computational results on a complete bipartite graph.

The remainder of this chapter is structured as follows. In Section 5.2 we formally define the problem. Section 5.3 present mathematical models with solution approaches. Concluding remarks are given in Section 5.4.

5.2 Problem Statement

It is not necessary to assign every participant a menu of choices. We define those users who receive a choice menu as *active* users. Users who are assumed to accept a match if assigned are defined as *inactive* users.

The decision process can be divided into 3 stages:

- Stage 1: the system assigns a menu of matching choices to active users;
- Stage 2: each active user picks one choice from the menu;
- Stage 3: the system collects the selected choices from active users at Stage 2, based on which matching assignments are finalized with the objective of maximizing the system-wide vehicle-miles.

Specifically, we consider a setting with an equal number of rider and driver trips, and any pair of (rider, driver) trips creates a feasible match. This is equivalent to a complete $m \times m$ bipartite graph G whose bipartition is the set of rider trips I and the set of driver trips J. We generate the weight for an arc as an integer uniformly drawn from 1 to 100, where the weight represents the savings created by matching the pair. For each left hand side node (active user), no two feasible matches will yield the same total savings for simplicity. To allocate the cost savings within a pair, we divide the total system's savings equally between the pair of participants. That is, for a pair (i, j), where $i \in I$ and $j \in J$, if the system's benefits by matching the pair is s, then we assume that user i and j can receive $s/2$ savings each.

At Stage 1, we assume that the active users are all the rider trips I. Since we have a complete bipartite graph, each rider would have at most m feasible match choices. The system determines the subset of choices to present to each rider.

Users' behavior in Stage 2 plays a key role in determining the system outcome. One realistic assumption is that people always pick the choice with best savings. We

denote this scenario as *choice with certainty*. In such a setting, people only care about the economic benefits. We can also assume that people take into account other factors besides the economic benefits and have a certain likelihood to pick each choice. This is denoted as the *choice with uncertainty* scenario. It is reasonable to assign an equal number of choices to each active user, which we denote by ξ. Specifically, we consider scenarios where $\xi = 2$.

In choice with certainty scenario, we solve a deterministic problem and the objective is to maximize the system's total vehicle-miles savings. In choice with uncertainty scenario, a stochastic optimization problem is considered and the objective is to maximize the expected system's total savings.

5.3 Mathematical Model and Solution Approaches

5.3.1 Choice With Certainty

We now present a model for providing a menu of choices to a user where we assume that each user will select the presented choice with largest savings. In this scenario, the maximum objective function value can be attained when each active participant chooses such an option that leads to a maximum weight matching solution on G. Such a solution is not always attainable. If the minimum savings option of rider i with driver j is in maximum weight matching solution, then pair i, j will never exist in the final matched solution for this scenario. It is still relatively easy to solve this problem. We claim that Algorithm 2 finds an optimal solution in choice with certainty scenario.

Denote the arc set by A and the weight on arc $(i, j) \in A$ by $c_{i,j}$. For each $i \in I$, denote the arc connected to i with smallest weight as sm_i, that is $sm_i = \{(i, j) \in A | c_{i,j} \leq c_{i,k}, \forall (i, k) \in A\}$, where $i \in I$. Denote $|I| = |J| = m$. The algorithm is described in Algorithm 2:

Claim 5.3.1. *Algorithm 2 provides an optimal solution.*

Algorithm 2 Polynomial Algorithm for Optimal Solution in Choice With Certainty Scenario

Find set $Q1$, where $Q1 = \{sm_i, \forall i \in I\}$. $G' = (V, A')$, where $A' = A \backslash Q1$.
Solve max weight matching problem on G'. Denote the set of arcs in optimal solution by $Q2$.
if $|Q2| = m$ **then**
  Return $Q1 \cup Q2$
end if
if $|Q2| < m$ **then**
  Let the set of left hand side nodes not matched be denoted set P. For each node in P, choose any adjacent arc not in $Q1$ and save those arcs in $Q3$. Return $Q1 \cup Q2 \cup Q3$
end if

Proof. It is easy to see that arcs in $Q1$ cannot appear in any optimal solution since every participant $i \in I$ always chooses the larger weight arc option. Thus, to solve the problem on graph G is equivalent to solve the problem on graph G'. If we obtain a perfect matching from G', then options in $Q1$ combined with options in $Q2$ provide an optimal solution. If a perfect matching is not obtained from G', we claim that $Q1 \cup Q2 \cup Q3$ is an optimal solution. To prove the claim, we only need to prove that to solve the max weight matching problem on G' is equivalent to solve the max weight matching problem on $Q2 \cup Q3$. This is true since $Q2 \cup Q3$ is a subset of the arc set of G', and $Q2$ is an optimal matching on G'. □

5.3.1.1 Extension to > 2 Choices

When participants are given more than two choices, we can create an integer programming formulation $P1$ for the choice with certainty problem as follows:

$$x_{ij} = \begin{cases} 1 & \text{if arc } (i,j) \text{ is chosen in Stage 1} \\ 0 & \text{otherwise} \end{cases}$$

$$y_{ij} = \begin{cases} 1 & \text{if arc } (i,j) \text{ is chosen in Stage 2} \\ 0 & \text{otherwise} \end{cases}$$

$$z_{ij} = \begin{cases} 1 & \text{if arc } (i,j) \text{ is chosen in Stage 3} \\ 0 & \text{otherwise} \end{cases}$$

$\forall i \in I, j \in J$

$$\text{Maximize} \quad \sum_{i \in I, j \in J} c_{ij} z_{ij} \tag{19}$$

$$\text{subject to} \quad \sum_{j \in J} x_{ij} = \xi \quad \forall i \in I \tag{20}$$

$$y_{ij} \leq x_{ij} \quad \forall j \in J, i \in I \tag{21}$$

$$\sum_{j \in J} c_{ij} y_{ij} \geq c_{ij} x_{ij} \quad \forall j \in J, i \in I \tag{22}$$

$$\sum_{j \in J} y_{ij} = 1 \quad \forall i \in I \tag{23}$$

$$z_{ij} \leq y_{ij} \quad \forall i \in I, j \in J \tag{24}$$

$$\sum_{j \in J} z_{ij} \leq 1 \quad \forall i \in I \tag{25}$$

$$\sum_{i \in I} z_{ij} \leq 1 \quad \forall j \in J \tag{26}$$

$$x, y, z \text{ binary} \tag{27}$$

The objective in choice with certainty scenario (19) is to maximize total savings created by the Stage 3 decision variables. Constraint (20) ensures each participant in I is assigned ξ options at Stage 1. Constraints (21),(22) and (23) enforce that at Stage 2 each active participant chooses the matching option with largest savings from the ξ options. Constraints (25) and (26) are the matching constraints for Stage 3.

Note that when $\xi = 1$, the formulation is equivalent to solving a maximum weight matching problem on G. This is an upper bound (denoted UB) on system-wide vehicle-miles savings. When $\xi = m$, where $m > 1$, the system will perform worse since users have the most flexibility to make choice. We test graph sizes from 3 to 12 and vary the parameter ξ from 1 to m. If not stated otherwise, the following computational results are averaged over 50 random graph samples. As a reminder,

the weight for an arc is generated as an integer uniformly drawn from 1 to 100, where the weight represents the savings created by matching the pair. For each active user, no two feasible matches will yield the same total savings. The ratio (in %) between the objective value of model (19) and the maximum weight matching UB is calculated for each parameter setting. The results are presented in Table 16.

Table 16: Solution Quality under Different Choice Menu Sizes

ξ	1	2	3	4	5	6	7	8	9	10	11	12
$m = 3$	100	97.9	82.9									
$m = 4$	100	99.5	96.0	74.2								
$m = 5$	100	99.9	99.6	95.1	77.9							
$m = 6$	100	100.0	99.9	98.2	94.8	74.5						
$m = 7$	100	100	100.0	99.9	99.2	92.2	70.6					
$m = 8$	100	100	100	100.0	99.5	97.8	90.9	71.1				
$m = 9$	100	100.0	100.0	100.0	99.8	99.2	97.7	91.5	71.5			
$m = 10$	100	100	100	100	100.0	99.9	99.1	97.4	89.2	69.8		
$m = 11$	100	100	100	100	100	99.9	99.8	99.4	96.8	90.0	68.4	
$m = 12$	100	100	100	100	100.0	99.9	99.8	99.6	99.2	97.6	89.5	69.0

When $\xi = 1$, the objective value is exactly equal to UB and the ratio is 100%. When ξ is small, the quality of the solution remains very close to that of the max weight matching upper bound solution. As ξ gets larger, the gap from the UB increases. Also note that when dealing with larger graphs with more choices per participant, the flexibility of the participant decision makers does not lead to large degradation from UB. For example, when $m = 10$, by assigning up to 4 choices to each left hand side node, we could still achieve the upper bound value.

5.3.2 Choice With Uncertainty

Assuming that participants will always select the maximum value matching option from a menu is naive. Not only are participants motivated by factors and constraints other than economic reward, but it is also possible that they will begin to look for ways to game the selection choice if they find that they are frequently not matched at all when they select the maximum value option. In this section, we initiate a

study of a system where participants may have some probability of not selecting the maximum value matching option presented to them. In such settings, the objective of the system is to present choices to participants that optimize the expected total savings. We formulate the problem as a two-stage stochastic optimization model.

Stochastic optimization provides a means to deal with uncertainty in optimization problems. Uncertainty is modeled by assuming probability distributions on the parameters of the problem. For an introduction, please refer to [13]. It has important application in transportation models, logistics, network design, etc. A widely-used model is a two-stage recourse model. In such a model, the decision maker must fix so-called first stage decisions now given uncertainty about the second stage; in the second stage, additional variables may model decisions that are made after uncertainty is revealed. The objective then is to choose the first-stage actions in order to minimize the expected total cost incurred, given the recourse decisions (and their costs) that are made under each realization of uncertainty. Stochastic optimization problems are often computationally very difficult, and often more difficult than their deterministic counterparts. The computational difficulty often arises from the fact that the distribution assigns a non-zero probability to an exponential number of scenarios, which leads to a huge increase in the problem complexity.

Recently there has been a growing interest in two-stage stochastic combinatorial optimization problems. Kong and Schaefer ([37]) introduces a two-stage stochastic max weight matching problem. In their problem, each edge has two weights, a first-stage weight and a discretely distributed second stage weight. The decision in first-stage is to choose a matching in the graph. Then, a scenario of the second-stage edge weight is revealed. The second-stage decision is to choose a matching over those vertices not matched by the first-stage matching. The objective is to maximize the total expected edge weight. They proved this problem is NP-hard and provided a factor $\frac{1}{2}$ approximation algorithm. Escoffier ([24]) improves the approximation ratio

and tackles the max weight spanning tree problem in the two-stage setting.

The first-stage decision of our problem is to pick η options to form a menu for each active user. For problems with $\eta = 2$, as long as the two options are decided, if we assume that we know the probability associated with each of the choices then we have 2^m scenarios at stage 2. Our objective is to maximize this expected savings over all the two option assignments in Stage 1.

5.3.2.1 Model Formulation

Consider a problem of assigning $\eta = 2$ choices to each participant in a $m \times m$ bipartite matching scenario. Let S be the set of all possible scenarios revealed after the two options are fixed for Stage 1. Since every left hand side node decision maker has two options, $|S| = 2^m$. Now denote 1 as the case that the larger savings option is chosen and 0 as the case that the smaller savings option is chosen. For example, when $m = 3$, S={(0,0,0), (0,0,1), (0,1,0), (0,1,1), (1,0,0), (1,0,1), (1,1,0), (1,1,1)}. We now consider an important simplification. Suppose that independent of the actual savings values provided by the two options, we assume that the probability of choosing the larger of the two options is fixed and known for each participant. Then, for the two options provided, we associate a probability η for a active user to choose the larger savings option and 1-η to choose the one with smaller savings.

In Stage 2, we solve $|S| = 8$ subgraphs each by a max weight matching algorithm to determine the system outcome under each of the participant choice scenarios. Combined with the probability to reach each subgraph, that is the choice outcome of the participants, we can calculate the expectation of the max weight matching objective value. The objective is to maximize this expectation value over all possible $\eta = 2$ assignments in Stage 1.

To incorporate the scenario that smaller weight arc could be chosen, we partition the set I into two subsets defined as below:

I_+^s: index set of left hand side nodes that have chosen higher weight arc in scenario s, $s \in S$.

I_-^s: index set of left hand side nodes that have chosen lower weight arc in scenario s, $s \in S$.

For example, if $s = (0, 0, 1)$, then $I_+^s = \{3\}$, meaning left hand side node 3 is in set I_+^s, $I_-^s = \{1, 2\}$, which indicates that left hand side nodes 1 and 2 are in set I_-^s. Note that the union of set I_+^s and I_-^s denotes the a complete set of left hand nodes. Mathematically, define

$$s(i) = \begin{cases} 1 & \text{if the } i\text{-th element of scenario vector } s \text{ is 1, } s \in S,\ i \in I \\ 0 & \text{if the } i\text{-th element of scenario vector } s \text{ is 0, } s \in S,\ i \in I \end{cases}$$

$I_+^s = \{i \in I | s(i) = 1, s \in S\}$

$I_-^s = \{i \in I | s(i) = 0, s \in S\}$

The two-stage stochastic optimization model $P2$ is then described as follows:

$$\text{Maximize} \quad \sum_{s\in S} p_s \sum_{i\in I, j\in J} c_{ij} z_{ij}^s \tag{28}$$

$$\text{subject to} \quad \sum_{j\in J} x_{ij} = 2 \quad \forall i \in I \tag{29}$$

$$y_{ij}^s \le x_{ij} \quad \forall j \in J, i \in I, s \in S \tag{30}$$

$$\sum_j c_{ij} y_{ij}^s \ge c_{ij} x_{ij} \quad \forall j \in J, i \in I_+^s, s \in S \tag{31}$$

$$\sum_j c_{ij} y_{ij}^s \le c_{ij} x_{ij} + M(1 - x_{ij}) \quad \forall j \in J, i \in I_-^s, s \in S \tag{32}$$

$$\sum_{j\in J} y_{ij}^s = 1 \quad \forall i \in I, s \in S \tag{33}$$

$$z_{ij}^s \le y_{ij}^s \quad \forall i \in I, j \in J, s \in S \tag{34}$$

$$\sum_{j\in J} z_{ij}^s \le 1 \quad \forall i \in I, s \in S \tag{35}$$

$$\sum_{i\in I} z_{ij}^s \le 1 \quad \forall j \in J, s \in S \tag{36}$$

$$p_s = \eta^{|I_+^s|}(1-\eta)^{|I_-^s|} \tag{37}$$

$$x, y, z \text{ binary, M is a big number} \tag{38}$$

The objective is to maximize the expected max weight matching objective values over 2^m scenarios, in which p_s is the probability that scenario s appears. For example, if $s = (1, 0, 1)$, then $p_s = \eta^2(1-\eta)$. Constraints are similar to those in $P1$. The differences lie in constraints (32). They represent the cases when users choose the smaller savings option. Note that there are exponential number of terms in objective function and constraints since there are 2^m choice scenarios.. To solve the problem directly is hard and approximation methods are needed.

5.3.2.2 Solution Approach

The fact the model $P2$ has an exponential number of terms in the objective and constraints makes solving it directly a hard task. A Monte Carlo simulation based approach can be applied to solve an approximation problem. The basic idea is to

generate a random sample and approximate the expected value function by the corresponding sample average function. This is denoted as SAA [36]. We also derive a series of construction methods to find a two choice assignment based on the graph structure. All the computational results are averaged over 50 random graph instances. The averaged computational time in seconds is denoted as t. An upper bound is the max weight matching objective value, which is denoted as $\tilde{v}$.

Solve $P2$ directly Due to the exponential number of terms in $P2$, the complexity makes solving $P2$ not easy in practice so we only report results for the smallest of graphs, with m ranging from 3 to 7 participants on each side of the bipartition. We denote the average objective value of $P2$ as v^* over 50 graph instances. Table 17 presents v^* relative to $\tilde{v}$ in percentages with $\eta = 0.9$ and 0.6. We observe that $\eta = 0.9$ case has v^* closer to the upper bound compared with $\eta = 0.6$ case for each graph instance. This is reasonable since there is a higher probability to choose the higher savings choice in $\eta = 0.9$ case and this leads to a higher expected value in $P2$. We can also find that the averag elapsed time in $\eta = 0.9$ case is shorter than that for $\eta = 0.6$ case. This shows that the mixed integer programming model is more complicated in $\eta = 0.6$ case due to the fact that a larger number of scenarios have probabilities of occurrence that are influential. The computational time increases fast while the graph size grows. For $m = 7$, it takes more than 5 minutes on average to get v^*. For large graph input, we employ SAA and construction methods to find a good solution.

P2-SAA One popular approximation approach is to sample a certain number of scenarios from the distribution, and solve the two-stage problem determined by this approximate distribution. This is known as sample average approximation (SAA) method. The basic idea is that a random sample is generated and the expected value function is approximated by the corresponding sample average of the function over the sample scenarios. Instead of considering all possible scenarios, we draw a fixed

Table 17: Optimal Solutions for $P2$

m	3	4	5	6	7
$\eta = 0.9$					
r(v^*, $\tilde{v}$)	89.7%	91.1%	91.2%	91.3%	91.4%
t	0	0	1	8	49
$\eta = 0.6$					
r(v^*, $\tilde{v}$)	72.8%	74.8%	74.6%	75.2%	75.7%
t	0	0	2	31	331

number of samples from binomial distribution $B(m, \eta)$ to mimic the choice made for each user decision maker out of the two options. For example, when $m = 3$, $\eta = 0.9$, we draw samples in which each user has probability 0.9 to choose the higher savings choice and probability 0.1 to choose the lower savings one. Given m user decision makers, we could use a vector of size $|m|$ with binary numbers to present the choice behavior of each user. We use 1 to stand for the scenario when the higher savings option is chosen and 0 for the scenario when the lower savings option is chosen. The vector $(1, 0, 1)$ represents a sample realization for the case when the first and third user decision makers choose higher savings option, and second decision maker chooses lower savings option. We record all sampled scenarios in set $\tilde{S}$. Note that the set $\tilde{S}$ may contain some repeated patterns. For example, $\tilde{s} = \{(0, 1, 1), (0, 0, 1), (0, 1, 1)\}$ is possible.

Given $\tilde{S}$, we can construct an optimization model with objective to maximize sample mean. The model $P2'$ is formulated as follows:

$$\text{Maximize} \quad (1/|\tilde{S}|)\sum_{s\in\tilde{S}}\sum_{i\in I, j\in J} c_{ij}z_{ij}^s \tag{39}$$

$$\text{subject to} \quad \sum_{j\in J} x_{ij} = 2 \quad \forall i \in I \tag{40}$$

$$y_{ij}^s \le x_{ij} \quad \forall j \in J, i \in I, s \in \tilde{S} \tag{41}$$

$$\sum_j c_{ij}y_{ij}^s \ge c_{ij}x_{ij} \quad \forall j \in J, i \in I_+^s, s \in \tilde{S} \tag{42}$$

$$\sum_j c_{ij}y_{ij}^s \le c_{ij}x_{ij} + M(1 - x_{ij}) \quad \forall j \in J, i \in I_-^s, s \in \tilde{S} \tag{43}$$

$$\sum_{j\in J} y_{ij}^s = 1 \quad \forall i \in I, s \in \tilde{S} \tag{44}$$

$$z_{ij}^s \le y_{ij}^s \quad \forall i \in I, j \in J, s \in \tilde{S} \tag{45}$$

$$\sum_{j\in J} z_{ij}^s \le 1 \quad \forall i \in I, s \in \tilde{S} \tag{46}$$

$$\sum_{i\in I} z_{ij}^s \le 1 \quad \forall j \in J, s \in \tilde{S} \tag{47}$$

$$x, y, z \text{ binary, M is a big number} \tag{48}$$

Compared with $P2$, the objective is sample average function instead of the expected value function. We denote the objective value as $\hat{v^s}$. Note that the constraints are similar to those in model $P2$. The only difference is that S is replaced by $\tilde{S}$. The set S is all the 2^m scenarios while $\tilde{S}$ includes the random samples drawn as designed. We run a certain number of replications and pick the one with best expectation. See Algorithm 3 for a general outline of the procedure. P is the number of replications to draw samples of size N. Among all the P replications, we pick the solution that provides the best objective value of $P2$.

In our experiment, we test three sample sizes: $N = 5$, 15, and 25. The replication number P is 20. Table 18 shows the computational results for $m = 6$ with probability $\eta = 0.9$ and 0.6. r($\hat{v^s}$, $\tilde{v}$) represents the sample average objective value relative to max weight matching objective upper bound averaged over 50 graph instances. t stands for the elapsed time in seconds. Compared with the results presented in Table 17,

Algorithm 3 P2-SAA Algorithm

for $p = 1$ to P **do**
  Generate a sample of size N and solve problem P2-SAA with optimal solution $\hat{x}^p$
  Calculate expectation $f(\hat{x}^p)$
end for
Find $\hat{x}^j$, where $f(\hat{x}^j) \geq f(\hat{x}^i), \forall i \in \{1, 2, ..., P\}$

SAA provides good solutions that are close to optimality in the case where m=6. A larger sample size yields a better solution but takes longer to solve. Solutions in the case $\eta = 0.9$ do not have much difference with different sample sizes inputs. While in the case $\eta = 0.6$, a larger sample size produces a better solution. This is reasonable since with $\eta = 0.9$ case, preferable scenarios have higher probability to appear and although we draw few samples, the scenarios captured are more representative of the real behavior of users. On the other hand, in the $\eta = 0.6$ case, all scenarios occur with nearly equal probability. Thus, without enough sample sizes drawn, the real users' behavior is hard to imitate. The above reason also leads to greater complexity of the integer program in the $\eta = 0.6$ case, and the elapsed time for the $\eta = 0.9$ case is shorter compared with the $\eta = 0.6$ case.

Table 18: P2-SAA, $m = 6$

N	5	15	25
$\eta = 0.9$			
r($\hat{v^s}$, $\tilde{v}$)	91.2	91.3	91.3
t	1	3	4
$\eta = 0.6$			
r($\hat{v^s}$, $\tilde{v}$)	74.2	75.0	75.1
t	3	19	55

When $m = 12$ in the case $\eta = 0.9$, on average it takes more than 6 minutes with sample size 15 and more than 16 minutes with sample size 25. It takes a longer time in the case $\eta = 0.6$. For the experiment with sample size 5, it takes more than 11 minutes on average. With large graph input, SAA method may not be an efficient

method. Thus, we derive some construction methods in the following section which are more efficient.

Construction Methods In this section, we construct two options for each active decision maker by several strategies. Since max weight matching solution provides an upper bound for the system's total savings and the arcs chosen in this solution would have relatively good weights, it is reasonable to fix one of the two options as the choice from solving the max weight matching problem. Denote set of arcs in max weight matching solution by $S1$. To determine the second option, it is beneficial to add an arc with smaller savings because we assume that people are more likely to choose the option with higher savings. Note that there is no guarantee that such a choice exists for each active decision maker. In this case, we need to define some backup choices if there does not exist a smaller savings choice. $S1 + ScLc$ and $S1 + ScLf$ are two methods designed to satisfy this goal. To generate a good expected value in the objective function of $P2$, it is reasonable to have solutions that contribute greatly to total system savings. Thus we have developed the $S1 + S2$ method.

Related terminology is defined as follows:

$S1$: the set of arcs in the max weight matching solution on original complete bipartite graph G

$S2$: the set of arcs in the max weight perfect matching solution on graph $G \backslash S1$.

$S_{small}(i)$: the set of arcs with weights smaller than the weights of arcs in $S1$ for active user i, $i \in I$.

$S_{large}(i)$: the set of arcs with weights greater or equal to the weights of arcs in $S1$ for active user i, $i \in I$.

$ScLc$: the neighbors of arcs in $S1$ chosen such that for each active user i, if $S_{small}(i)$ exists, choose the arc with largest weight in $S_{small}(i)$; otherwise, choose the arc with smallest weight in $S_{large}(i)$.

$ScLf$: the neighbors of arcs in $S1$ chosen such that for each active user i, if $S_{small}(i)$ exists, choose the arc with largest weight in $S_{small}(i)$; otherwise, choose the arc with largest weight in $S_{large}(i)$.

Set of construction methods are:

$S1 + S2$: two options are the union of $S1$ and $S2$.

$S1 + ScLc$: two options are the union of $S1$ and $ScLc$.

$S1 + ScLf$: two options are the union of $S1$ and $ScLf$.

Solutions from construction methods are shown in Table 19 with $m = 6$. The objective value in $P1$ given the solution provided by construction method is denoted as $\hat{v^c}$. Construction method $S1 + S2$ performs better in $\eta = 0.6$ case but worse in $\eta = 0.9$ case compared with methods $S1 + ScLc$ and $S1 + ScLf$. Note that $S1 + S2$ method considers to have a good total savings while $S1 + ScLc$ and $S1 + ScLf$ methods are designed to lead each active decision maker to choose the solutions from $S1$. Under such designs, $S1 + S2$ method yield a good solution in $\eta = 0.6$ case where the probability of appearance of each scenario is more evenly distributed. But those methods do not provide a good solution for $\eta = 0.9$ case since the appearance of scenarios displays an uneven pattern.

Table 19: Construction Methods, $m = 6$

	$S1 + S2$	$S1 + ScLc$	$S1 + ScLf$
$\eta = 0.9$			
r($\hat{v^c}$, $\tilde{v}$)	80.3	91.0	91.0
$\eta = 0.6$			
r($\hat{v^c}$, $\tilde{v}$)	74.5	71.9	71.9

In order to improve the solution quality, we employ 1− exchange local search method based on solutions from the above construction methods. During local search, we replace one of the two-option assignment by another arc associated with the same active user each time. A best improvement rule is used and we stop until we reach

the local optimal solution. Table 20 shows the 1− exchange local search method for solutions in Table 19. After local search, the solutions are very close to the optimal solution in various construction methods referring to the results in Table 17.

Table 20: 1 - exchange Local Search on Construction Methods, $m = 6$

	$S1 + S2$	$S1 + ScLc$	$S1 + ScLf$
$\eta = 0.9$			
$\mathrm{r}(\hat{v^c}, \tilde{v})$	90.5	91.3	91.3
$\eta = 0.6$			
$\mathrm{r}(\hat{v^c}, \tilde{v})$	75.0	74.8	74.8

Note that the above construction methods are polynomial solvable and we would be able to find a feasible solution in a reasonable time. Considering the number of participants in the dynamic ride-sharing problem, construction methods might be a practical way to find a feasible and good choice menu in timely manner.

5.4 Summary and Future Research

In this chapter, we design a menu of matching choices provided to participants with the objective to maximize total system benefits, which would be intentionally used for dynamic ride-sharing problem. We perform the analysis on a complete m by m bipartite graph with η choices designed for users represented by left hand side nodes. Two scenario assumptions are considered. One is to assume that users always prefer the choice with better savings and the other one is to assume that users are likely to choose any of the choice provided. In order to maximize the system's total benefits after users' behavior in choice making, we are trying to induce users to choose the option that is close to a max weight matching solution which is an upper bound. Under the first scenario assumption, a polynomial algorithm is derived. The other scenario assumes that people are more likely to choose the option with higher savings but still possible to choose the other option. We formulate this scenario into

a two stage stochastic optimization model and maximize the expected system's total savings. Solving the model directly is a hard task and we employ the sample average approximation method for an estimated solution. The complexity is simplified since an integer program with a much smaller size is solved. We could derive a fast algorithm integrating brach-and-cut method during the SAA procedure since our problem has objective function piecewise linear and convex in first stage variables [67]. To avoid solving an integer programming, we derive a set of construction methods which are polynomial solvable. Local search heuristics can improve the solution further more.

Our analysis on the complete bipartite graph could support further analysis on the dynamic ride-sharing problem. In real time ride-sharing settings, we may no longer have a complete bipartite graph. A menu with zero or two choices would be provided. Instead of analyzing the whole graph, we could solve a subgraph with enough feasible matches by the similar methods derived for the complete bipartite graph. Our future research will focus on applying the mechanism design for multiple choice to the dynamic ride-sharing system to construct a more user-friendly system.

REFERENCES

[1] "Indicator: Occupancy rates of passenger vehicles," tech. rep., European Environmental Agency, 2005.

[2] "International energy outlook 2009," tech. rep., Energy Information Adminstration, 2009.

[3] AAA, "Your driving costs: how much are you really paying to drive?," tech. rep., AAA Association Communication, 2009.

[4] Agatz, N. A. H., Erera, A., Savelsbergh, M. P. W., and Wang, X., "Dynamic ride-sharing: A simulation study in metro atlanta," *Transportation Research Part B*, 2011.

[5] Ajtai, M., Aspnes, J., Naor, M. Rabani, Y., Schulman, L. J., and Waarts, O., "Fairness in scheduling," *Journal of Algorithms*, vol. 29, no. 2, pp. 306–357, 1998.

[6] Alumur, S. and Kara, B. Y., "Network hub location problems: The state of the art," *European Journal of Operational Research*, vol. 190, pp. 1–21, 2008.

[7] Amey, A., "Proposed Methodology for Estimating Rideshare Viability Within an Organization: Application to the MIT Community," *Transportation Research Board Annual Meeting 2011*, vol. Paper 11-2585, 2011.

[8] Baldacci, R., Maniezzo, V., and Mingozzi, A., "An exact method for the car pooling problem based on lagrangean column generation," *Operations Research*, vol. 52, no. 3, pp. 422–439, 2004.

[9] Bass, F. M., "A new product growth model for consumer durables," *Management Science*, vol. 15, no. 5, pp. 215–227, 1969.

[10] Bell, J., "Two startups harness facebook's power to connect riders to rides - social networking web site hopes to reverse sagging ride-share rates," *ABC news*, 2007.

[11] Berbeglia, G., Cordeau, J. F., , and Laporte, G., "Dynamic pickup and delivery problems," *European Journal of Operational Research*, vol. 202, no. 1, pp. 8–15, 2010.

[12] Berbeglia, G., Cordeau, J. F., Gribkovskaia, I., and Laporte, G., "Static pickup and delivery problems: a classification scheme and survey," *Top*, vol. 15, no. 1, pp. 1–31, 2007.

[13] Birge, J. R. and Louveaux, F. V., "Introduction to stochastic programming," *Springer, New York*, 1997.

[14] Brunekreef, B. and Holgate, S. T., "Air pollution and health," *Lancet*, vol. 360, no. 9341, pp. 1233–1242, 2002.

[15] Calvo, R. W., de Luigi, F., Haastrup, P., and Maniezzo, V., "A distributed geographic information system for the daily car pooling problem," *Computers & Operations Research*, vol. 31, no. 13, pp. 2263–2278, 2004.

[16] Campbell, J. F., "Strategic network design for motor carriers.," in *Logistics Systems: Design and Optimization* (Langevin, A. and D., R., eds.), pp. 245–278, New York: Springer-Verlag, 2005.

[17] Cordeau, J. F. and Laporte, G., "The dial-a-ride problem: models and algorithms," *Annals of Operations Research*, vol. 153, no. 1, pp. 29–46, 2007.

[18] Cordeau, J. F., Laporte, G., Potvin, J. Y., and Savelsbergh, M. W. P., "Transportation on demand," in *Transportation* (Barnhart, C. and Laporte, G., eds.), Handbooks in Operations Research and Management Science, pp. 429 – 466, Amsterdam: North Holland: Elsevier, 2007.

[19] Cortes, C. E., Matamala, M., and Contardo, C., "The pickup and delivery problem with transfers: Formulation and a branch-and-cut solution method," *European Journal of Operational Research*, vol. 200, no. 3, pp. 711–724, 2010.

[20] Dailey, D. J., Loseff, D., and Meyers, D., "Seattle smart traveler: dynamic ridematching on the world wide web," *Transportation Research Part C-Emerging Technologies*, vol. 7, no. 1, pp. 17–32, 1999.

[21] Dial, R. B., "Autonomous dial-a-ride transit introductory overview," *Transportation Research Part C-Emerging Technologies*, vol. 3, no. 5, pp. 261–275, 1995.

[22] Edmonds, J. and Johnson, E., "Matching: A well-solved class of integer linear programs," in *Combinatorial Optimization Eureka, You Shrink!* (M. Jnger and G. Reinelt and G. Rinaldi, ed.), vol. 2570 of *Lecture Notes in Computer Science*, pp. 27–30, Springer, 2003.

[23] Emmerink, R. H. M. and vanBeek, P., "Empirical analysis of work schedule flexibility: Implications for road pricing and driver information systems," *Urban Studies*, vol. 34, no. 2, pp. 217–234, 1997.

[24] Escoffier, B., Gourves, L., Monnot, J., and Spanjaard, O., "Two-stage stochastic matching and spanning tree problems: Polynomial instances and approximation," *European Journal of Operational Research*, vol. 205, pp. 19–30, 2010.

[25] Fagin, R. and Williams, J. H., "A fair carpool ccheduling algorithm," *IBM Journal of Research and Development*, vol. 27, no. 2, pp. 133–139, 1983.

[26] Ferguson, E., "The influence of employer ridesharing programs on employee mode choice," *Transportation*, vol. 17, no. 2, pp. 179–207, 1990.

[27] Gale, D. and Shapley, L. S., "College admissions and the stability of marriage," *American Mathematical Monthly*, no. 69, pp. 9–15, 1962.

[28] Ghoseiri, K., Haghani, A., and Hamedi, M., "Real-time rideshare matching problem," *University of Maryland, Department of Civil and Environmental Engineering*, vol. UMD-2009-05, 2011.

[29] Gruebele, P. A., "Interactive system for real time dynamic multi-hop carpooling," 2008.

[30] Hall, R. W. and Qureshi, A., "Dynamic ride-sharing: Theory and practice," *Journal of Transportation Engineering*, vol. 123, no. 4, pp. 308–315, 1997.

[31] Hensher, D. A., "Climate change, enhanced greenhouse gas emissions and passenger transport what can we do to make a difference?," *Transportation Research Part D: Transport and Environment*, vol. 13, no. 2, pp. 95–111, 2008.

[32] Herbawi, W. and Weber, M., "Evolutionary multiobjective route planning in dynamic multi-hop ridesharing.," in *EvoCOP'11*, pp. 84–95, 2011.

[33] Horn, M. E. T., "Fleet scheduling and dispatching for demand-responsive passenger services," *Transportation Research Part C*, vol. 10, pp. 35–63, 2002.

[34] Kelley, K. L., "Casual carpooling enhanced," *Journal of Public Transportation*, vol. 10, no. 4, pp. 119–130, 2007.

[35] Kleiner, A., Nebel, B., and Ziparo, V., "A mechanism for dynamic ride sharing based on parallel auctions," in *Proc. of the 22th International Joint Conference on Artificial Intelligence (IJCAI)*, (Barcelona, Spain), 2011. To appear.

[36] Kleywegt, A. J., Shapiro, A., and De Mello, T. H., "The sample average approximation method for stochastic discrete optimization," *SIAM Journal on Optimization*, vol. 12, pp. 479–502, 2002.

[37] Kong, N. and Schaefer, A. J., "A factor 1/2 approximation algorithm for two-stage stochastic matching problems," *European Journal of Operational Research*, vol. 172, pp. 740–746, 2006.

[38] Kunzli, N., Kaiser, R., Medina, S., Studnicka, M., Chanel, O., Filliger, P., Herry, M., Horak, F., Puybonnieux-Texier, V., Quenel, P., Schneider, J., Seethaler, R., Vergnaud, J. C., and Sommer, H., "Public-health impact of outdoor and traffic-related air pollution: a european assessment," *Lancet*, vol. 356, no. 9232, pp. 795–801, 2000.

[39] Lee, D. H., Wang, H., Cheu, R. L., and Teo, S. H., "Taxi dispatch system based on current demands and real-time traffic conditions," in *Transportation Network Modeling 2004*, Transportation Research Record, pp. 193–200, 2004.

[40] Lee, K. T., Lin, D. J., and Wu, P. J., "Planning and design of a taxipooling dispatching system," in *Transit: Bus, Rural Public and Intercity, and Paratransit*, Transportation Research Record, pp. 86–95, 2005.

[41] Levin, I. P., Mosell, M. K., Lamka, C. M., Savage, B. E., and Gray, M. J., "Measurement of psychological factors and their role in travel behavior," *Transportation Research Record*, no. 649, 1977.

[42] Levy, S., "Baby, you can drive in my car - via web," *Newsweek*, 2007.

[43] Li, J., Embry, P., Mattingly, S. P., Sadabadi, K. F., Rasmidatta, I., and Burris, M. W., "Who chooses to carpool and why? examination of texas carpoolers," *Transportation Research Record*, no. 2021, pp. 110–117, 2007.

[44] Li, X. and Quadrifoglio, L., "Feeder transit services: Choosing between fixed and demand responsive policy," *Transportation Research Part C*, vol. 18, pp. 770–780, 2010.

[45] Liaw, C. F., White, C. C., and Bander, J., "A decision support system for the bimodal dial-a-ride problem," *Ieee Transactions on Systems Man and Cybernetics Part a-Systems and Humans*, vol. 26, no. 5, pp. 552–565, 1996.

[46] Mahajan, V., Muller, E., and Bass, F., "Diffusion of new products: Empirical generalizations and managerial uses," *Marketing Science*, vol. 14, no. 3, pp. G79–G88, 1995.

[47] McGuckin, N. and Srinivasan, N., "Journey to work trends in the united states and its major metropolitan areas 1960 - 2000," tech. rep., US Department of Transportation Federal Highway Administration, 2003.

[48] Naor, M., "On fairness in the carpool problem," *Journal of Algorithms*, vol. 55, no. 1, pp. 93–98, 2005.

[49] Nielsen Company, "2006 panorama study," tech. rep., The Nielsen Company, 2007.

[50] Paquette, J., Cordeau, J. F., and Laporte, G., "Quality of service in dial-a-ride operations," *Computers & Industrial Engineering*, vol. 56, no. 4, pp. 1721–1734, 2009.

[51] Pentico, D., "Assignment problems: A golden anniversary survey," *European Journal of Operational Research*, vol. 176, pp. 774–793, 2007.

[52] Powell, W. B., "A stochastic formulation of the dynamic assigment problem, with an application to truckload motor carriers," *Transportation Science*, vol. 30, no. 3, pp. 195–219, 1996.

[53] Powell, W. B., Bouzanene-Ayari, B., and Simpo, H. P., "Dynamic models for freight transportation," in *Transportation* (Barnhart, C. and Laporte, G., eds.), vol. 14 of *Handbooks in Operations, Research and Management Science*, pp. 285–365, Elsevier, 2007.

[54] Powell, W. B., Towns, M. T., and A., M., "On the value of optimal myopic solutions for dynamic routing and scheduling problems in the presence of user noncompliance," *Transportation Science*, vol. 34, no. 1, pp. 67–85, 2000.

[55] Psaraftis, H. N., "A dynamic-programming solution to the single vehicle many-to-many immediate request dial-a-ride problem," *Transportation Science*, vol. 14, no. 2, pp. 130 – 154, 1980.

[56] Quadrifoglio, L., Dessouky, M. M., and Ordonez, F., "Mobility allowance shuttle transit (mast) services: Mip formulation and strengthening with logic constraints," *European Journal of Operational Research*, vol. 185, no. 2, pp. 481–494, 2008.

[57] Roth, A. E., Rothblum, U. G., and Vande Vate, J. H., "Stable matchings, optimal assignments and linear programming," *Mathematics of Operations Research*, vol. 18, no. 4, pp. 803–828, 1993.

[58] Roth, A. E. and Sotomayor, M., "Two sided matching: A study in game-theoretic modelling and analysis," *Cambridge University Press, Cambridge. Econometric Society Monograph Series*, 1990.

[59] Rothblum, U. G., "Characterization of stable matchings as extreme points of a polytope," *Mathematical Programming*, no. 54, pp. 57–67, 1992.

[60] Santos, A., McGuckin, N., Nakamoto, H. Y., Gray, D., and Liss, S., "Summary of travel trends: 2009 national household travel survey," tech. rep., U.S. Department of Transportation Federal Highway Administration, 2011.

[61] Saranow, J., "Carpooling for grown-ups — high gas prices, new services give ride-sharing a boost; rating your fellow rider," 2006.

[62] Savelsbergh, M. W. P. and Sol, M., "The general pickup and delivery problem," *Transportation Science*, vol. 29, no. 1, pp. 17–29, 1995.

[63] Schrank, D. and Lomax, T., "2007 urban mobility report," tech. rep., Texas Transportation Institute, 2007.

[64] Spivey, M. Z. and Powell, W. B., "The dynamic assignment problem," *Transportation Science*, vol. 38, no. 4, pp. 399 – 419, 2004.

[65] Tsao, H. and Lin, D.-J., "Spatial and temporal factors in estimating the potential of ride-sharing for demand reduction," tech. rep., Institute of Transportation Studies, University of California, Berkeley, 1999.

[66] Vande Vate, J. H., "Linear programming brings marital bliss," *Operations Research Letters*, vol. 8, pp. 147–153, 1989.

[67] Verweij, B., Ahmed, S., Kleywegt, A. J., Nemhauser, G., and Shapiro, A., "The sample average approximation method applied to stochastic routing problem: A computational study," *COMPUTATIONAL OPTIMIZATION AND APPLICATIONS*, vol. 24, pp. 289–333, 2003.

[68] Walters, H., "Hitching a ride for earth's sake," *Business Week*, 2007.

[69] Wiedenkeller, P., "Suddenly, sharing a ride looks good," 2008.

[70] Winter, S. and Nittel, S., "Ad-hoc shared-ride trip planning by mobile geosensor networks," *International Journal of Geographic Information Science*, vol. 00, no. 00, pp. 1–21, 2006.

[71] Xing, X., Warden, T., Nicolai, T., and Herzog, O., "Smize: a spontaneous ride-sharing system for individual urban transit," in *Proceedings of the 7th German conference on Multiagent system technologies*, MATES'09, (Berlin, Heidelberg), pp. 165–176, Springer-Verlag, 2009.

[72] Yang, J., Jaillet, P., and Mahmassani, H., "Real-time multivehicle truckload pickup and delivery problems," *Transportation Science*, vol. 38, no. 2, pp. 135 – 148, 2004.

[73] Zhao, J. M. and Dessouky, M., "Service capacity design problems for mobility allowance shuttle transit systems," *Transportation Research Part B-Methodological*, vol. 42, no. 2, pp. 135–146, 2008.