MODELING AND ANALYSIS OF AN AUTONOMOUS MOBILITY ON DEMAND SYSTEM

KATARZYNA ANNA MARCZUK
Master of Civil Engineering
awarded by Warsaw University of Technology (2012)

Supervisors: Professor Der-Horng Lee, National University of Singapore
Professor Emilio Frazzoli, Massachusetts Institute of Technology.

Examiners: Dr Ong Ghim Ping Raymond, National University of Singapore
Professor Srinivasan Dipti, National University of Singapore
Professor Gregory Washington, University of California, Irvine

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF CIVIL AND ENVIRONMENTAL ENGINEERING
NATIONAL UNIVERSITY OF SINGAPORE

2017

Declaration

I hereby declare that this thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis.

This thesis has also not been submitted for any degree in any university previously.

Katarzyna Anna Marczuk
12 January 2017

This dissertation is dedicated to my late grandmother, Halina Drabarek,
and my family.

Acknowledgments

I am grateful to many who have supported, guided, and inspired me these last few years. First and foremost, I would like to express my sincere gratitude to my advisors: Professor Der-Horng Lee and Professor Emilio Frazzoli. They have been a constant source of advice and encouragement. I would like to thank the Singapore Ministry of Education, for opportunity of pursuing my doctoral study under one of the most prestigious programs in the field. I am also grateful to my examiners, Professor Dipti Srinivasan and Dr Ong Ghim Ping Raymond from National University of Singapore and Professor Gregory Washington from University of California, Irvine, for valuable feedback.

I would like to thank all people I met at SMART, NUS and MIT, who shaped my education and personality. Completing this thesis would not be possible without people at Future Urban Mobility IRG, Singapore-MIT Alliance for Research and Technology. My sincere thanks goes to Dr. Harold Soh Soon Hong, for insightful comments and encouragement, but also for challenging my work with the hard questions which incited me to grow big. I would like to acknowledge Dr. Carlos Miguel Lima Azevedo, for many stimulating discussions and for suggesting research directions to pursue. I would like to thank SimMobility team at SMART, Prof. Moshe Ben-Akiva, Kakali Basak, Dr. Muhammad Adnan, Dr. Sebastián Raveau, Dr. Bilge Kucuk Atasoy, Neeraj Deshmunkh, Balakumar Marimuthu, Harish Loganathan, and Zhiyong Weng, for their great support in teaching me how to use SimMobility. I would also like to thank current and past Autonomous Vehicle team at SMART, Prof. Marcelo H. Ang Jr, Dr. Guo Ming James Fu, Scott Pendleton, Hans Andersen, Dr. Zhuang Jie Chong, Lucas Tetsuya Kuwae, Tawit Uthaicharoenpong, Dr. Malika Meghjani, Dr. You Hong Eng, Dr. Xiaotong Shen, Dr. Wei Liu, Marcello Scarneccia, people from LIDS at MIT, Michal Capino, Dr. Jean Gregorie, Valerio Varricchio, and many others, for all work and fun together. I am grateful to Dr. Tan Rui, for being a wonderful senior, and for guiding me through entire PhD and beyond.

My next gratitude goes to the management and administrative staff at SMART, NUS and MIT, Dr. Bingran Zuo, Andrew Tong, Cecille Loquias Maquito, Janet Long Wei Ling, Juana Bte Rawi, Maria Gotoking, Jocelyn S Sales, and Charulatha Vengadiswaran, for being helpful and supportive.

I would also like to acknowledge my friends. Nahid Afrasiyabi and Dr. Meysam

Sharifzadeh, for being my family when I first arrived in Singapore, Dr. Kamila Kamińska, for her unconditional help in many circumstances, Magdalena Buka, Pratik Chaudhari, Marzena Chodkiewicz, Dorota Januszko, Przemysław Konstańczuk, Anna Marta Pióro Vinotti, Zuzanna Rzeplińska, Anna Szrajber, Olga Waskiewicz and many others, for all life experience which we share together.

I took part in many activities at NUS and MIT, which are beyond coursework and research, with countless number of people teaching and helping me every step. I am grateful to NUS Students' Sports Club, NUS Karate Team, NUS Golf Club, NUS Wellness Center, MIT Recreation Center, for keeping me in a good shape throughout my PhD; to Polish Community in Singapore, for making my life in Singapore feeling much closer like home.

I owe everything I am and everything I have ever achieved to my parents, Alicja Aleksandra Marczuk and Mirosław Marczuk. I am also grateful to my sisters, Monika Joanna Marczuk and Magdalena Aleksandra Marczuk, for their unconditional friendship.

One last word of gratitude goes to Robert Ledziínnski, for his never-failing love and sympathy. I would not have made it that far if there were no him. Thank you for being here for me.

Executive Summary

During the post-World War II period automobiles became more widespread and with large-scale urban areas commuting trend accelerated and people became car-dependent. This mass use of motor vehicles led to some unforeseen consequences in terms of traffic congestion, pollution, reduced safety and climate change. Growth in the global transportation is likely to further exacerbate current problems, especially in densely populated urban areas. We need disruptive ideas and technologies in transportation, which could transform current mobility patterns. In this dissertation we look at the feasibility of assembling a fleet of autonomous vehicles for a mobility on demand service in order to ease congestion, increase safety, and reduce environmental impacts. We consider a new mode of urban transportation, which we refer as an Autonomous Mobility on Demand (AMOD) system. AMOD consists of a fleet of light-weight electric and self-driving vehicles, which are available through an e-hailing service application. We focus on the issues related to the fleet management of an autonomous mobility on demand system.

The main contribution of this dissertation is development of framework and related algorithms to demonstrate the role of autonomy in Mobility on Demand systems and its impact in terms of feasibility and efficiency through modeling, simulation, algorithm development and experimental demonstration. The proposed methodologies are applicable to large-scale systems in different simulations' setup.

Our methodology comprises three design and management levels differentiated by the timeframe:

(i) At the planning, or strategic, level we determine the size of operating area, estimated demand, and the number and locations of stations (depots), where we park the vehicles.

(ii) At the tactical, or long-term operational, level we determine the fleet size (number of vehicles which are required to run the system), initial locations of the vehicles, static rebalancing, and static pricing.

(iii) At the operational level we decide on short-term operational decisions, which include vehicle to customer assignment, vehicle routing, dynamic rebalancing, and dynamic pricing.

The proposed methodologies are applicable to case-studies of large-scale AMOD systems in Singapore. Our approach is simulation-based and we make use of SimMobility—an agent-based simulation platform.

In Chapter 1, we present trends in mobility on demand services and different operation strategies for managing fleets of shared-mobility systems. Next, we state what is the contribution of this dissertation and provide outline of the thesis.

In Chapter 2, we provide detailed description of our development which is a fleet management system for an autonomous mobility on demand service (AMOD Controller). We describe our methodology and systematically outline system architecture and functional organization of AMOD Controller. We evaluate different modeling approaches of analytical and simulation traffic models and justify choosing SimMobility over other available traffic simulators. Finally, we describe communication infrastructure between AMOD Controller and SimMobility.

In Chapter 3, we focus on the strategic aspect of managing AMOD systems—models for selecting the number and location of new facilities. By facilities we understand stations, or distribution centers, for autonomous vehicles. First, we provide classification of facility location problems and related work in the field. Next, we formulate four facility location models for an AMOD system: (i) distribution of stations based on the location factor rating, (ii) one station for the entire system, (iii) stations based on the set covering problem, and (iv) stations based on the maximal coverage problem. In order to assess the merits of our methods we present a simulation study based on the demand for private vehicle trips for the Central Business District in Singapore. We conclude that location of distribution centers plays an important role in the overall performance of the AMOD system. Finally, we acknowledge limitation of our models and provide future directions.

In Chapter 4, we formulate the fleet assignment and routing problem. Given a distribution (location) and capacity of available vehicles, the fleet assignment problem faced by the operator is to determine which vehicle should pick up which customer. The fleet routing is to determine the route for each assigned vehicle. We first give introduction to the assignment problem by providing definitions, terminologies and related work. Next, we describe how the assignment is implemented within AMOD Controller. We formulate two types of assignment: (i) for customers who are not willing to share a ride (called single-riders), and (ii) for customers who are willing to share a ride (multiple-riders). We extend the framework to demand management with price incentives and present results comparing different assignment algorithms. Our simulation results indicate that the use of optimization-based methods instead of simple greedy approaches substantially improves the performance of AMOD systems. Finally, we acknowledge limitation of our work and state future directions.

In Chaper 5, we describe a core-component of this dissertation, which is empty vehicle rebalancing. The rebalancing models aim in finding optimal control policies for realigning demand and supply. First, we provide terminologies, definitions, and related work in the filed of empty vehicle rebalancing. Next, we provide explanation how the AMOD Controller handles rebalancing. We propose static and dynamic formulation of the rebalancing problem. Static rebalancing solves for (i) the total number of vehicles required to operate the system, and (ii) rebalancing patterns between different locations for the course of the entire day. Dynamic rebalancing solves rebalancing problem while the system is in operation. We demonstrate the value of rebalancing policies in the simulation-based case-studies in Singapore. We conclude that the ability to meet the fluctuating demand for service is a crucial factor in the success of AMOD system.

In Chapter 6, we present a large-scale implementation of the AMOD system in the Central Business District in Singapore. We present two alternatives of introducing AMOD service in Singapore: (i) case study of the central business district in Singapore, and (ii) case study of the extended central business district in Singapore. We describe simulation setup and results for both case studies. Our results suggest that introducing AMOD service has a great potential to serve for future urban mobility.

Finally, in Chapter 7, we present conclusions of the thesis, limitations and directions for future work.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

During the post-World War II period automobiles became more widespread. With large-scale suburban areas commuting trend accelerated and people became car-dependent [Marczuk et al., 2016]. This mass use of motor vehicles led to some unforeseen consequences in terms of traffic congestion, pollution, safety and climate change, which proves that privately-owned motor vehicles are not a sustainable solution to serve for the future of personal urban mobility. Due to the congestion, most of the vehicles used in urban areas are heavily underutilized, i.e., an average private vehicle is parked for around 22 hours daily and its driving speed is usually five to ten times slower than the design speed [Pavone, 2015]. As one reaction to this fact, in Singapore, extra fees have been introduced for car ownership (Certificate of Entitlement *COE*), road usage (Electronic Road Pricing *ERP*) and car park usage. However, growing first- and middle-class, who still can afford to acquire private vehicles [Singapore Department of Statistics, 2016] will further increase urban traffic problems, which cannot be solved by fees alone. At this point, one emerging business model might be seen in selling mobility instead of cars by providing vehicles to large-scale car-sharing systems [Firnkorn and Muller, 2011]. This model has already spurred a growing interest in mobility on demand (MOD) systems—particularly a one-way vehicle sharing—as a sustainable alternative to privately owned vehicles. The challenge is therefore, to ensure flexibility of private vehicles while removing the need of car ownership. One emerging trend, which can help to tackle this challenge, is development of autonomous vehicles (AVs). Due to extensive technological developments in the field of autonomous vehicles [Google, 2016, Tesla, 2016, Pendleton et al., 2015], it is gaining more ground to deploy AVs for mobility services.

This dissertation is set to assess and demonstrate the role of autonomy in mobility on demand systems and its impact in terms of feasibility and efficiency through modeling, simulation, algorithm development and experimental demonstration.

1.1 Trends in Transportation

Traffic deaths are at fourth position among the leading causes of death for all age groups [Corwin et al., 2015]. A vast majority of the car accidents involves alcohol or human error [Hars, 2016, Frazzoli, 2014]. By the end of this century, global warming could increase the world's mean temperature by 4 degrees Celsius [Bank, 2012]. The effects would be dramatic: *unprecedented heat waves, severe drought, and major floods in many regions, with serious impacts on human systems, ecosystems, and associated services.* Road transport is responsible for about 5 billion tonnes of CO_2 annually (data: 2008) which is almost 20% of total global CO_2 emissions [Hars, 2016]. Growth in global transportation [Bank, 2012] is likely to further exacerbate current problems such as car accidents, pollution, road congestion, and parking availability. This proves that the current transportation system is not a sustainable solution for mobility, especially in densely populated urban areas. We need a disruptive ideas and technology in transportation, which could transform current patterns of mobility. Shape of the future evolution of transportation is driven by a series of industry-changing forces and mega-trends [Corwin et al., 2015]: (i) Battery and electric fuel-cell vehicles offer higher energy efficiency at lower emission levels. (ii) Lightweight materials enable automakers to reduce the weight of vehicles without reducing passengers safety. (iii) Vehicles outfitted with vehicle-to-vehicle, *V2V*, and vehicle-to-infrastructure, *V2I*, communications can gain awareness of where they are in relation to other vehicles and potential hazards so they can avoid accidents. (iv) At the same time, among young adults, the model of personal mobility is inclined toward consumption based on *pay-per-use* rather than upfront purchase of a capital asset. For them, owning a vehicle is no longer a *status symbol.* This new ecosystem of mobility promises sustainable and affordable personal mobility. Further, there are four different personal mobility futures emerging from two critical trends: vehicle control (driver versus autonomous) and vehicle ownership (private versus shared) [Corwin et al., 2015]. Based on [Corwin et al., 2015], the most conservative vision is that private and manually driven vehicles remain the norm due to the convenience, security and privacy that come with owning vehicles. The second vision anticipates continued growth of shared access to vehicles. As the cost per kilometer decreases, some people start to see shared-use mobility as a more economical, convenient, and sustainable way to get around, particularly for short point-to-point movements. The third possible trend assumes that autonomous vehicles prove to be viable, safe, convenient, and economical. However, most drivers still prefer owning their own vehicles, but with driverless functionality. The fourth vision sees a convergence of both the autonomous and vehicle-sharing trends. In this dissertation, we believe that the current state of mobility will transform in the direction of shared mobility, catalyzing the adoption of autonomous drive. This new ecosystem of mobility promises sustainable and affordable personal mobility. However, aspirations for its contributions in solving large-scale transportation problems such as pollution, road congestion and land use are still to be clarified. Therefore, this dissertation attempts to answer the question: what would change if one day there is an autonomous mobility on demand system?

1.2 Background for Mobility on Demand Services

Extensive usage of private vehicles has led to increased traffic congestion, pollution, safety issues and climate change. The challenge is therefore, to ensure flexibility of private vehicles while removing the need of car ownership. An alternative can be seen in shared-use mobility (or mobility on demand) systems. Mobility on demand can be broadly understood as a service to meet transportation needs by offering vehicles (cars, bikes, scooters, etc.) for some amount of time whenever they are needed. While, MOD was a niche market a decade ago, today the phenomenon has entered the mainstream, with a broad array of companies investing in it [Le Vine and Polak, 2015]. In this shared mobility, you do not have to be an owner of a physical item (vehicles or bicycles) to use it. The items can be accessed by multiple users via information and communications technology (ICT) on a pay-per-use basis.

The economic and social benefits of shared mobility are significant and wide-ranging, incorporating benefits of both, private and public transport. It allows door-to-door service with a flexibility of personal vehicles without the cost of owning one. Due to smaller fleet size and higher utilization rates [Marczuk et al., 2015] the MOD services lower static land consumption (requires smaller number of parking lots).

Many have also suggested that shared transportation can help reduce emissions as well as car ownership rates and household transportation costs, i.e., the literature on changes in vehicle ownership associated with car-sharing shows that between 9 and 13 cars are sold or not purchased for each shared car [Martin et al., 2010]. At the same time, however, more independent research has to be done to investigate the long-term changes associated with the shared-use mobility.

In the literature there are five (5) types of mobility on demand systems: (i) bike-sharing, (ii) car-sharing, (iii) taxi, (iv) ride-sourcing, and (v) autonomous mobility on demand, which are presented in Subsections 1.2.1, 1.2.2, 1.2.3, 1.2.4, 1.2.5, respectively. Example of shared systems are shown in Figure 1.1.

(a) Bike-sharing in New York. **(b)** Taxi in Singapore. **(c)** ZipCar One-way in Boston.

Figure 1.1: Examples of different shared mobility systems.

1.2.1 Bike-sharing

Bike-sharing (BS or *bike-sharing system*, BSS) is a service providing short-term bicycle rental and is the oldest and one of the most recognized form of mobility-on-demand systems.

The first public-use bicycles known as *White Bicycles* were introduced in Amsterdam in 1965 [Shaheen and Guzman, 2011], followed by *Yellow Bikes* in French city of La Rochelle in 1974 [Shaheen et al., 2010]. Since then, experiences have multiplied, and models have become more complex. Number of bike-sharing systems worldwide increased from 13 in 2004 to 855 systems a decade later [Fishman, 2015], as illustrated in Figure 1.2. As of January 2016, the largest number of BSSs was in China accounting for 296 systems [Paul and Russell, 2016], while the world's largest bike-sharing is in the city of Hangzhou, China, with around 78,000 bicycles in their program [Paul and Russell, 2016]. World's fleet of vehicles at the end of 2015 was 1,270,000 bicycles, while China's fleet accounted for 1,036,400 of the total number (over 80 % of the total population of the public bicycles). Other examples of massive-scale programs of public bicycles are the city of Taiyuan (China), Paris (France), Shanghai (China), and London (United Kingdom) [Paul and Russell, 2016].

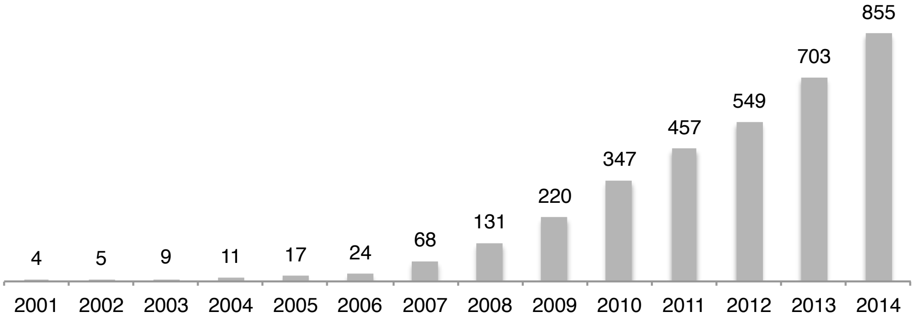


Figure 1.2: Growth of bike-sharing systems worldwide (x-axis represents years, y-axis represents the number of operating bike-sharing systems worldwide).

As of today, we recognize four (4) generations of BS schemes:

1. The first generation BSS, called white bikes (or free bikes), is a system in which bicycles were left unlocked in the city for everyone to use it freely. In this type of unregulated program the bicycles were very often stolen or damaged and the initiatives failed.

2. The second generation BSS is a response to the failure of the first generation system and known as a coin-deposit system. In this system a small deposit is needed to unlock a bicycle from a docking station. The second generation BSS consists of distinct design bicycles and docking stations, however the users are anonymous, which led to lots of thefts.

3. The third generation BSS is the information technology (IT) based system. The systems are membership-based which prevents thefts, and incorporate advanced IT

for bicycle reservations, pick-up, drop-off, and tracking. By incorporating advanced technology the third generation systems gained a worldwide popularity [Shaheen et al., 2010].

4. The fourth generation BSSs include all the main components seen in third-generation systems but they are seamlessly integrated with public transportation and other alternative modes. Its stations are conveniently located near transit hubs and a single payment smartcard creates access to all available transportation options.

Some of the major benefits of bike-sharing schemes are: (i) BSSs increase mobility options while decreasing short distance travel times. The literature consistently finds that convenience is the major factor motivating people to sign up for BSS as it eliminates the first and last mile problem and therefore supports multimodal transport connections. (ii) Bike share systems are great for the environment as they produce no emissions and they reduce traffic congestion (iii) Bike sharing provides an easy option for people to foster an active lifestyle by diverting a greater share of trips to bicycling, which may be beneficial to our health. (iv) Riding a bike can be just a fun way to see the city in a unique and personal way. (v) Using public bicycle may contribute to individual financial savings as very often public bicycles are free of charge. (vi) Current bike-sharing schemes have a potential to be integrated with electric bikes (e-bikes), GPS (global positioning system), dockless systems and public transport [Fishman, 2015]. Some researchers suggest that shared e-bikes could provide a higher level of service compared to existing BSSs with benefits such as the ability to travel longer distances and over hills with less fatigue and sweat [Ji et al., 2014]. Implicit in many of the aforementioned benefits is the assumption that a significant proportion of users are transferring from a single occupant car use to a public bicycle. Yet, a number of papers have reported that the BSS programs show little impact on reducing car use and it is quite common for the majority of bike-share trips to be substituting from sustainable modes rather than the car [Shaheen et al., 2010, Fishman et al., 2013, Fishman, 2015]. On the contrary, research on Hangzhou's bike-sharing program suggests that auto ownership may not discourage bike-sharing [Shaheen and Guzman, 2011] as its members exhibited a higher rate of auto ownership (22 %) than non-members (11 %).

Despite these considerable advantages, there are several factors which discourage people from using BSS and these includes (but may not be limited to): (i) Limited access to the bicycle stations is one of the major barriers to bike-sharing [Fishman, 2015, Fishman et al., 2014]. (ii) Another critical barrier to BS is a lack of immediate access to helmets in countries in which they are mandatory [Fishman et al., 2014]. (iii) Perceived danger from motorized traffic is another key impediment to bike-sharing. Many riders also feel uncomfortable about riding an unfamiliar bicycle. (iv) Some bikers, who are scared of cars or (and) new to the city, go dangerously slow or make careless decisions. They put themselves and everyone else on the road at risk. (v) Weather conditions are a permanent constraint to cycling, but only under extreme conditions (pouring rain or blistering heat) [Fradea and Ribeiro, 2014].

(vi) Bicycle trips are affected by the differences of slopes between origin and destination. Steep slopes can make the ascents difficult for cyclists and the descents can lead to fast speeds which might be unsafe. (vii) Another barrier for the potential users is the requirement of credit cards to sign up, which—mostly poor people—do not have. Analysis of a recent member survey by Washington, D.C.'s Capital Bike-sharing revealed that half of survey respondents reported an annual income of $100,000 or more [Parzen et al., 2016]. The story further underscored criticism that bike-sharing has largely failed to reach low-income residents [Fishman, 2015, Parzen et al., 2016].

BSSs are considered as a cheap and reliable public mobility options and that is why they do face financial challenges. The questions remain on how to profitably set up and run the system? To give an answer a cost-benefit analysis should be evaluated. An efficient, reliable and cost-effective bike-sharing system is assessed based on two critical performance metrics [Gauthier et al., 2014]: (i) average number of daily uses per public bike (ideally, four to eight daily uses per bike), and (ii) average daily trips per resident (ideally, one daily trip per twenty to forty residents). These two metrics have an inverse relationship, i.e., many systems have a high average daily use per bike because they have too few bicycles in circulation, and this means that market penetration is low. Therefore, the planning of a BSS should be carefully calibrated to ensure performance is within the optimum range for both metrics.

The lesson learned from bike-sharing operators is *go big or go home.* Most of BSS failures were due to the small number of bikes and stations, as well as the long distances between stations and limited operating hours [Gauthier et al., 2014].

1.2.2 Car-sharing

Car-sharing (CS, or car-sharing system, CSS) is defined as the organized collective use of passenger cars. It provides members access to a fleet of shared vehicles for short-term use and is developed as a sustainable solution at affordable cost to serve for personal mobility in densely populated cities [Barrios and Godier, 2014, Fan, 2013]. Users of car-sharing system gain benefits of having a private vehicle without the costs of ownership, as for many people living in urban areas, the cost of a vehicle ownership cannot be easily justified because they simply do not drive enough.

Car-sharing pioneered in Switzerland in 1987, followed by Germany in 1988, but it was only in the last decade that the concept has evolved into a mobility solution (Figure 1.3), which is a key perspective for the *automotive revolution.* The car-sharing numbers presented in Figure 1.3 demonstrate a steady growth. With development of information technology (IT), idea of sharing vehicles is gaining more ground with almost 100,000 car-sharing vehicles available worldwide in 2014. This disruptive trend of transforming the auto industry from selling cars to selling mobility, accelerates rise of new technologies and changes consumer preferences. However, compared to car rental, total fleet size and revenues for car-sharing still remain relatively small, and up to this day, car-sharing services do not replace vehicle ownership, rather they substitute for taxis and car rentals [Brown, 2015].

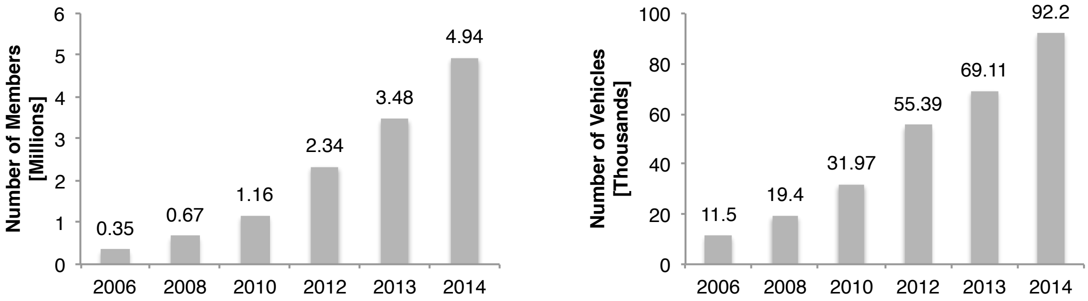


Figure 1.3: Number of members (left) and vehicles (right) in car-sharing worldwide, 2006-2014, (data based on [Le Vine et al., 2014]).

Objective of car-sharing systems is to offer drivers a sustainable transportation mode at the efficiency of public transportation and flexibility of a private automobile. Four models of the car-sharing operation have been established:

(i) Traditional station-based (round-trip) car-sharing, which restricts vehicles to be picked up and returned to the same location (station). This type of car-sharing is mainly operated by car rental companies.

(ii) One-way or point-to-point car-sharing is another form of station-based system, but it allows customers to pick up a vehicle at one location and drop it off at another. Usually the pick up and drop off locations and times have to be indicated in advance, so that the operator can manage the vehicles.

(iii) Free-floating car-sharing, which allows pick up and drop off a vehicle at any location within an operating area. They usually operate without fixed stations and do not require a booking process in advance. It offers a flexibility of hiring a vehicle but at a cost of higher uncertainty of getting one when in need. Free-floating car-sharing services are primarily offered by car-manufacturers.

(iv) Peer-to-peer (or person-to-person) car-sharing, which allows car owners to monetize the excess capacity of their vehicles by enrolling them in car-sharing programs. Existing car owners make their vehicles available for others to rent for short periods of time.

Today, most CS operators use traditional station-based system offering round-trip rental [Privat, 2015]. Traditional business model, which requires cars to be collected and returned at the same location is easier to managed by the operator. The drawback is that it becomes uneconomical for trips that require lengthy stays at the destination, because the user must also pay for the time the car is parked. For this reason point-to-point services can quickly attract three to four times the number of members of a traditional round-trip service [Brown, 2015]. But the cost of point-to-point systems is much higher since the systems requires bigger fleet size to start with [Brown, 2015] and is more challenging from the operational perspective due to rebalancing issues. An advantage of station-based, return and point-to-point services, is guaranteed parking lot, which is not guaranteed in any free-floating model.

Examples of the most recognized car-sharing programs: (i) Car2Go (a subsidiary of Daimler AG) operates free-floating systems in 30 cities with over a million members worldwide [Gibbs, 2016, Car2Go, 2016]. It is the largest car-sharing company worldwide. (ii) ZipCar (owned by Avis Budget Group) is the largest car-sharing company in the USA, which operates a station-based (return and point-to-point) car-sharing [ZipCar, 2016]. (iii) Autolib' is an electric car-sharing service which operates in Paris, Bordeaux and Lyon in France. As of 3 July 2016, Autolib' maintains a fleet of 4,000 all-electric cars and almost 6,000 charging points [Geron, 2016]. (iv) DriveNow (a joint venture between BMW and Sixt) and ReachNow (owned by BMW). In most markets, DriveNow vehicles can park the cars anywhere on the street with a special permit from the local municipality [DriveNow, 2016, ReachNow, 2016]. (v) Unite launched by Audi, which allows up to four people to literally *share* ownership of an Audi for up to two years. Group members use a smartphone app to schedule who gets the vehicle when, and a beacon, which electronically tracks personal usage (everyone's monthly payment is adjusted based on the usage) [Audi Unite, 2016]. (vi) Turo (formerly RelayRides) is a company that operates a peer-to-peer carsharing. (vii) Smove is a point-to-point car-sharing service in Singapore. Smove allows to book its cars to run trips for UBER [Smove, 2016]. and many others.

Car-sharing, similarly to other on-demand services, is helping to create more diversified and sustainable transport system. As presented in Figure 1.4, 23% of private vehicle expenses are variable, while 77% are fixed charges. This price structure forces a more direct consideration of how much each trip costs. CS therefore, gives consumers a practical alternative to owning a personal vehicle. Users of car-sharing pay minimal fixed costs but almost 5 times higher variable costs [Litman, 2015], which forces a more direct consideration of how much each trip costs. Availability of CS have been found to be a deterring factor for a future vehicle purchase [Litman, 2015, Duncan, 2011, Jorge and Correia, 2013]. Research suggests that car-sharing decreases discretionary trips and lead to more efficient travel patterns, i.e., households that join car-sharing programs typically reduce their vehicle use 40-60% [Litman, 2015]. For the above reasons, CS can be seen as cost effective replacement of owning a vehicle with neither acquisition nor maintenance costs.

Car-sharing can make living in urban areas more attractive [Duncan, 2011]. It is found to be a good alternative for short trips within the city as the vehicles are located in the city and priced by minute or hour. They are easy to check in and out. Conventional vehicle rentals may be cheaper per mile for longer trips, but much more expensive for shorter trips and not that easy to check in and out. Car-sharing usually offers a choice of vehicle types. Users choose the vehicle type that best meets their needs for a particular trip rather than upfront

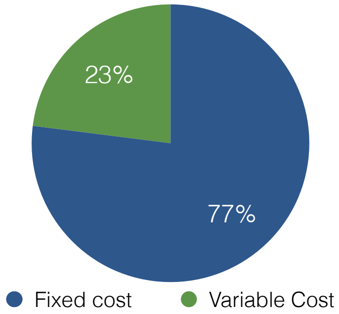


Figure 1.4: Price structure for vehicle ownership (based on [Litman, 2015]).

purchase of an over-sized vehicle to meet occasional demand. CS may also release users from the burden of vehicle maintenance and parking. One of the major concerns for substituting privately owned vehicle with a CS subscription is limited reliability of CSSs. People have proven little preference for a cooperative organization structure as they do not like to be constrained by the availability of the vehicles, which may lesser their flexibility [Litman, 2015]. Another barrier for some people to subscribe to CS is that the CS vehicle does not carry forward a *status symbol* associated with owning a car. Car-sharing is found to be the most suitable for higher density urban neighborhoods with good walking, cycling and public transit services [Litman, 2015] and it may fail in low density areas.

The trend is clear that a growing number of customers are relying on sharing services such as car-sharing, bike-sharing or ride-sourcing, rather than purchasing cars [Kuhnimhof et al., 2012, Poultney, 2015]. Research by Ford suggests that in the U.K. alone the car-sharing sector will grow by 23 per cent in the next 10 years [Poultney, 2015]. [Privat, 2015] estimates that car-sharing will attract 26 millions users worldwide by 2020. According to a report from Navigant Research, global car-sharing services revenue is expected to grow from $1.1 billion in 2015 to $6.5 billion in 2024 [Lorenz, 2016]. Car-sharing can create up to $1.5 trillion in additional revenue potential for automakers in 2030, compared with about $5.2 trillion from traditional business model, which accounts for sales and aftermarket products and services [Gao et al., 2016].

Summing up, the biggest challenge for CS operators is providing unlimited access to the vehicles to gain reliability of the service.

1.2.3 Taxi services

Taxicab service (or taxi service) is a point-to-point transport system, very often considered as part of the public transportation within cities, that provides flexible and convenient way of travelling [Rodriguez-Valencia, 2014]. A taxi can be defined as a vehicle for hire provided with a driver. Taxicabs are privately owned and operated by independent contractors within regulated environments [Rodriguez-Valencia, 2014].

Taxi was introduced in Europe in the early 17-th century and is playing a major role in urban transportation since then [Paraboschi et al., 2015]. In most cities, the taxicab industry was brought under municipal or state regulation at the beginning of XX-century [Teal and Berglund, 1987]. Those regulations, up to this day, are similar worldwide and commonly consider three factors: (i) number of taxicab in the city, (ii) minimum and maximum rates, and (iii) service standards [Teal and Berglund, 1987]. [Rayle et al., 2014] suggested that low barriers to entry in the taxi industry tended to enable over-competition, leading to aggressive and unsafe driver behavior, poor vehicle maintenance, and congestion. Secondly, lack of information on price or service quality before choosing a vehicle was a problem in street-hail and cab-stand markets. However, with the appearance of information and communications technologies (ICT), hailing a for-hire vehicle no longer requires standing on a street corner or placing a telephone call, and rating systems might resolve the lack-of-information problem. Despite the technology potential, the taxi system business model, the hailing experience, and the fare system have remained the same [Paraboschi et al., 2015].

There are different ways in which people can order a taxi and there are different possible combinations in which a driver can look for a new passenger. Most places allow a taxi to be *hailed* on the the road as it is approaching. Another option is to be picked up from a taxi stand. Passengers also commonly call a central dispatch office to book taxi. The newest method to hire is through E-hailing apps [Paraboschi et al., 2015].

Taxis fill a critical gap by providing transportation when driving or other public transit modes are not possible [Rayle et al., 2014]. As mentioned before, they are relatively convenient. Unlike public transportation, taxi services are very flexible and can be customized according to the passenger's need. Taxis are commonly used for airport access and during the nighttime [Rodriguez-Valencia, 2014]. They provide 24 hour service 7 days a week. Thanks to the regulations imposed on the taxi companies, they employ professional and well experienced drivers, which can provide a sense of safety and reliability. The same applies for the vehicles. Although there are thousands of taxi companies all over the world, managing millions of taxicabs altogether, the problem of online taxi dispatching has not been explored thoroughly [Maciejewski and Nagel, 2014]. So much of freedom in getting a cab results in low efficiency of the service and a massive amount of taxicabs on the streets, about half of which wanders in search of new passengers [Paraboschi et al., 2015]. Aligning supply with the demand is the most challenging task for the taxi operators. This topic is studied in details in Chapter 5.

1.2.4 Ride-sourcing Services

Ride-sourcing services (RSS or ride-sourcing, RS, also known as Transportation Network Companies, TNCs) use online platforms to connect passengers with drivers who use personal, non-commercial, vehicles. Growth of RS services was possible thanks to advances in information and communication technology. Ride-sourcing trips replicate taxi trips, so they are usually one-way, with similar usage patterns to the trips on point-to-point car-sharing services. Passengers of RS services request a ride through the mobile application, which then assigns a driver to the request. The assignment is usually done based on the estimated shortest arrival time or distance. A driver only learns the passenger's destination when the the customer is picked up, which takes care of the problem of being refused access to a taxi when traveling to undesirable parts of town [Jalloh, 2014]. Because the passenger's credit card is linked to the E-Hail account, then upon arriving at the destination, the passenger can simply walk out the car and the payment will be processed independently [Jalloh, 2014]. For the driver, he does not need to worry about unpaid fares and robbery. The fare is usually charged based on the trip distance and time, approximately 80% of which goes to the driver, with the remaining to the ride-sourcing service. Many of these applications maintain a rating system that allows mutual ratings after the trip is completed.

Ride-sourcing services target similar transportation gap as taxi services. Survey presented in [Rayle et al., 2014] shows that ride-sourcing serves a similar demand to taxis, where a substantial portion of sampled ride-sourcing trips is spatially and temporally not well served by public transit. In addition, the analysis in [Rayle et al., 2014] shows that ride-sourcing users also appear to be less likely to own an automobile. One of the most recognized TNC is Uber, which is available worldwide [Uber, 2016]. The mobile application for Uber has been launched in 2009 in San Francisco linking commuters with drivers 1.5. Most of Uber drivers use personal, non-commercial, vehicles. There are also companies (e.g., Smove in Singapore), who are renting vehicles to ride for Uber. Another big players are Lyft in the USA, Easy Taxi in Hong Kong, Didi in China and Grab in South East Asia.

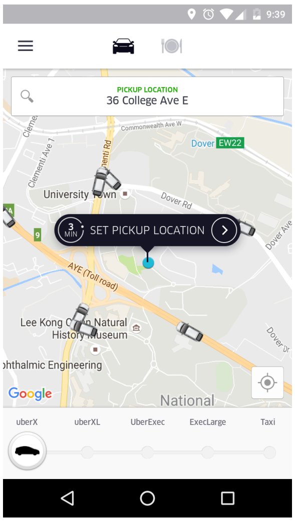


Figure 1.5: A screen-shot of Uber's e-hailing app (passenger's side). Before the customer makes the booking, he or she can see available vehicles around his or her location and an approximated waiting time to be picked up.

RSS have made it possible to secure a car or taxi from a smartphone from any location. Anyone can provide the RSS, what increases the number of cars available [Jalloh, 2014]. Although RSSs have a positive effect by increasing the supply of drivers, the drivers might not be motivated to reach high standards of professionalism and safety, simply because the service is deregulated (easy access for everyone to the E-Hail network as service provider). Critics, claim that RSSs unfairly flout existing regulations, compete with public transit, increase congestion at peak times, mislead consumers through opaque pricing practices, and endanger public safety [Rayle et al., 2014]. This new services have created a competition that has reduced the market share of traditional taxi services and lowered the overall profits of drivers [Jalloh, 2014]. E-Hail services are engaged in an intense battle to provide the cheapest service and gain more customers. With such cheap prices and wide availability of cars, customers may get into the habit of taking a car even for short-distance trips. On the other hand, low prices negatively impact drivers' earnings, who have to work longer to compensate lower fares.

The popularity and growth potential for ride-sharing services is tremendous. A challenge for RSSs is how to expand its supply scale without recruiting private drivers as its service suppliers. One of the promising approaches to tackle this problem might be introducing an automated mobility on demand (AMOD) system with robotic shared vehicles, which is introduced in Section 1.2.5.

1.2.5 Autonomous Vehicles for Mobility on Demand

Automated mobility on demand (AMOD) systems attempt to provide a one-way car-sharing with self-driving electric vehicles. AMOD systems rely on high levels of vehicle-automation, which are now entering the commercial marketplace [Pendleton et al., 2015, Martinez et al., 2014]. Road vehicles capable of operating independently of real-time human control (so called autonomous vehicles, AVs) will likely become more widely available in the near future [Le Vine et al., 2015].

Autonomous vehicles Vehicles that drive themselves are no longer just fantasies. The initial developments of autonomous vehicles can be traced back to the 80's [Thorpe et al., 1988]. It is currently considered a key research effort in many car manufacturer and mobility/robotics research centers [Fagnant and Kockelman, 2015, Azevedo et al., 2016] and has recently started to be marketed for personal use [Google, 2016, Tesla, 2016, Pendleton et al., 2015, Martinez et al., 2014]. Industry experts believe that highly automated vehicles would become viable by 2020 and fully autonomous ones could be common by 2030 [Sun et al., 2014]. However, aspirations for its contributions in solving large scale transportation problems such as pollution, road congestion and land use are also high [Fagnant and Kockelman, 2015]. Such vehicles are expected to bring massive economic benefits relative to human-driver vehicle control. Vehicle automation can increase safety as over 90% of car accidents are caused by the human error [Pavone, 2015, Ozimek, 2014]. In 2014 in the USA alone, there were 29,989 fatal car crashes [National Highway Traffic Safety Administration, 2016]. According to the Department of Transportation the official value of a statistical life is US$9.2 million, so if self-driving cars can save 30,000 lives a year this is a yearly benefit of US$276 billion. In addition to the lost lives, the deadly crashes in 2005 also lead to $41 billion in medical and work loss costs [Ozimek, 2014]. So the total cost of deadly crashes per year is around $317 billion. Then there are the non-deadly crashes and crashes where nobody is hurt, which account for additional crash cost of US$226 billion [Ozimek, 2014]. The societal cost of congestion is approximately worth US$ 100B/year [Frazzoli, 2014]. The health cost of congestion is approximately US$50 B/year [Levy et al., 2010]. The next major source of savings is that passengers will become fully disengaged from vehicle operation and free to perform non-driving tasks (which translates to increased productivity), which is estimated at savings of US$1.2T/year [Frazzoli, 2014]. Autonomous vehicles hold great promise for mobility on demand systems because they can cooperate with each other and rebalance themselves. When we enable a widespread car-sharing with autonomous vehicles, we gain another US$1.8T/year of benefits [Frazzoli, 2014]. From the financial perspective, in the US alone, the autonomy vehicles can contribute to savings of more than US$4T per year. This number still misses a lot of important benefits like senior citizens who currently can't drive, and the cost savings to households who gain much cheaper access to cars by renting them by the hour [Ozimek, 2014].

Autonomous mobility on demand AMOD can be considered as an element of a complete integrated door-to-door future urban mobility, where efficient and reliable mass transit is still needed. AMOD system does not require drivers to operate vehicles and it can be designed in such way to maximize the number of demand served. There are already few deployments of AV for mobility on demand services [Balea, 2016, Chafkin, 2016]. The world's first self-driving taxis developed by *NuTonomy* is picking up passengers in Singapore [Balea, 2016], while a giant *Uber* allows customers in downtown Pittsburgh to get self-driving cars for their Uber rides [Chafkin, 2016]. With these advancements it became realistic to see autonomous taxis on the streets.

AMOD can make inter-modal transportation easier providing solution for the first- and last-mile problem. AMOD vehicles are demand-responsive, which means that they do not operate on a regular schedule like buses or trains, but rather only run when there is a request for the service. This allows for a long-term environmental sustainability and potential cost savings for customers, while relieving people from the burden of driving. There is a believe that, in the long run, prices will fall so low that the per-mile cost of travel, even for long trips in rural areas, will be cheaper in a driverless car than in a private car [Chafkin, 2016]. For example, according to [Spieser et al., 2014], the estimated cost to own a mid-sized car in

Singapore is approx. S$ 18,162/year. To reduce this cost, a pay-as-you-drive transportation service would be a good alternative (according to the same study, estimated cost of mobility on demand service in Singapore with autonomous vehicles is S$12,563/year). Another benefit of using driverless technology is that the AMOD vehicles can provide mobility to people who cannot, should not, or prefer not to drive such as elderly, youth or disabled and increase productivity. It may also make the rural communities more attractive because of availability of shared journeys to nearby cities becomes affordable and will not lead to loss of productive time. Vehicles can also platoon together maintaining constant and close separation between each other and at the same time reducing an overall occupancy of the road infrastructure. Through the system-level coordination, autonomous vehicles can use existing roads more efficiently e.g., by routing vehicles via not heavily congested roads [Spieser et al., 2014]. Shared autonomous vehicles could increase urban space by 15 to 20 percent [Spieser et al., 2014], largely through the elimination of parking spaces. Freeing up this space would make the cities greener and increase quality of life. To the consumer, AMOD offers an alternative transportation mode to the private vehicles.

Despite these prominent advantages, an inadequate and unbalanced fleet of shared vehicles can result in service unavailability problems, particularly during periods of high demand. This topic is brought up in Section 1.2.6.

1.2.6 Fleet Management for Mobility on Demand Systems

Fleet management systems, *FMS*, have been adopted in a number of areas such as freight logistics, railway industry, airlines, material handling systems or shared mobility services [Sayarshad et al., 2012]. FMS can include a range of functions, such as operations planning, fleet (vehicles and drivers) scheduling, vehicle tracking and diagnostics, speed management, fuel management and health and safety management. It allows companies which rely on transportation to minimize their cost associated with vehicle investment, while improving efficiency, productivity and reducing their overall transportation and staff costs. Attractiveness of mobility on demand systems is highly depended on availability of vehicles to the customers, which translates to efficiency of the fleet management system. If the system does not respond *well* to the demand, e.g., due to shortage of the supply some customers are rejected to be served, it will become less reliable and in a long-run will not provide a sustainable alternative to the ownership. The most fundamental aspect of all MOD systems is therefore determining the appropriate number of vehicles to serve for the requested trips. This can be derived from travel patterns using the simulation-based approach. In general, the fleet size of any transportation system largely depends on: (a) the size of the operating area (which is related to the distance of trips), (b) the average user demand, (c) routing policies, (d) the level of service that the system provider wants to achieve, (e) the assignment and ride-sharing policies, and (f) the amount of redistribution.

Operation of Bike-sharing System BSSs are designed to provide a transit mode for short commuting trips within the city. As introduced in Section 1.2.1, BSSs have a tremendous potential as an *on-demand* service. A first strategic decision in designing a bike-sharing system is the planning of the number of stations, their locations and sizes. These issues are addressed in [Garcia-Palomares et al., 2012, Lin and Yang, 2011, Martinez-Barbera and Herrero-Perez, 2012]. [Garcia-Palomares et al., 2012] proposes a GIS-based method to calculate the spatial distribution of the potential demand for bike-sharing trips and locate stations using location allocation models. Authors show that with this analysis,relatively isolated stations can be eliminated. [Lin and Yang, 2011] determine the number and locations of bike stations, the network structure of bike paths connected between the stations, and the travel paths for users between each pair of origins and destinations. They illustrate the proposed model with a small network example. [Martinez-Barbera and Herrero-Perez, 2012] present a mixed integer linear program, that simultaneously optimizes the location of bike-sharing stations, the fleet dimension and the relocation activities required in a regular day. The authors present the design and deployment of their model for the BSS in Lisbon. Other long-term operation decisions involve static pricing and fixing the number of bikes in the system [Sayarshad et al., 2012]. At a more tactical level, pricing and reservation policies should be determined by the operator [Ji et al., 2014]. [Sayarshad et al., 2012] presents a mathematical model to optimize a bike-sharing system by determining the minimum required bike fleet size that minimizes unmet demand, underutilized bikes, and the rebalancing trips. The proposed approach is applied to an example problem, which proves that the model gives an effective solution. [Ji et al., 2014] describe the operational concepts and system requirements of a fully automated electric bike-sharing system demonstrated through a pilot project at the University of Tennessee. Based on the results of the simulation, the authors present a cost constrained e-bike sharing system design that can maintain a high level of system reliability (e-bike and battery availability) through optimal battery charging and distribution management. Finally, the day-to-day operation of the system requires rebalancing the system in order to meet the demand for bicycles and vacant lockers at the location and times desired by the users. Rebalancing issue arises from the fact that the bicycles are not meant for performing long recreational trips. To maintain this objective in most bike-sharing systems 30 minutes of usage is free of charge. After the free minutes, charges increase exponentially forcing people to keep trip length short. These factors lead to short one-way trips that cause imbalances in bike distribution [Caggiani and Ottomanelli, 2012, Caggiani and Ottomanelli, 2013, Ciari et al., 2014a, Farahani et al., 2014, Forma et al., 2015, Garcia-Palomares et al., 2012, Martinez-Barbera and Herrero-Perez, 2012, Raviv and Kolka, 2013, Raviv et al., 2013, Sayarshad et al., 2012, Vogel and Mattfeld, 2010], which implies that bike-sharing systems should be efficiently rebalanced to provide a quality of service for the customers. The relocation problem for the BSS is twofold: (i) to ensure that the users can find an available bicycles at their origins, and (ii) to provide enough vacant lockers to allow users to return bicycles at their destinations. In order to balance the difference between supply and demand at each station, the BSS operator has to operate staffed trucks for manual relocation of bicycles or/and offer incentives to encourage customers to use certain stations, i.e., to dock bicycles at empty docking locations [Shaheen et al., 2010, Vogel and Mattfeld, 2010, Fricker and Gast, 2014, Zhang and Pavone, 2015]. This rebalancing approach differs from the rebalancing of a fleet of vehicles as vehicles cannot be loaded and unloaded from the trucks easily.

Operation of Taxi System To solve the problem of operation of taxi system several simulation-based studies have been carried out [Alshamsi et al., 2009, Cheng and Nguyen, 2011, Horn, 2002, Lee et al., 2004, Maciejewski and Nagel, 2014, Maciejewski, 2014]. [Alshamsi et al., 2009] apply a multiagent self-organization technique to the taxi dispatching problem. The authors compare performance with actual data and show that the proposed technique decreases total waiting time by up to 25% in comparison with the real system and increases taxi utilization by 20%. [Cheng and Nguyen, 2011] propose a multi-agent simulation platform to investigate interactions among taxis and to evaluate impact of implementing certain management policies. The major contribution of their work is incorporation of the analysis on the real-world driver's behaviors. [Horn, 2002] describe a software system designed to manage the deployment of a taxis system. The scheduling system includes automated vehicle dispatching procedures designed to achieve a favorable combination of customer service and efficiency of a vehicle deployment. Simulation tests indicate that improvement procedures yield substantial efficiencies and that the system will be effective in real-time applications. [Lee et al., 2004] propose a simple dispatch method based on the Global Positioning System. The system assigns the nearest, in terms of the shortest straight-line distance, taxi for each booking location. Data from the simulations of a real network of Singapore show that the proposed method is capable of being over 50% more efficient in the reductions in passenger pickup times and average travel distances as compared to the current taxi dispatcher in Singapore. [Maciejewski and Nagel, 2014] and [Maciejewski, 2014] present an optimization toolbox for the dynamic vehicle routing problem embedded into a microscopic, behavior-based traffic simulation *MATSim*. The authors show the off-line and on-line taxi dispatching problems, and present three different dispatching strategies that are then evaluated on a realistic scenario. The computational results show that a strategy of assigning the nearest empty taxi to each incoming request works well as long as the system is not overloaded.

Operation of Car-sharing System The findings summarized in the literature review presented in [Jorge and Correia, 2013] show that the major issue in development of a one-way car-sharing is how to balance the demand and supply [Jorge and Correia, 2013]. To estimate the minimum required fleet size, many researchers have focused on rebalancing strategies for both station-based and free-floating carsharing system [Alshamsi et al., 2009, Barrios and Godier, 2014, Barth and Todd, 1999, Brownell and Kornhauser, 2014, Ciari et al., 2014b, Correia and Antunes, 2012, Fan et al., 2008, Horn, 2002, Jorge et al., 2014, Kek et al., 2009, Maciejewski, 2014, Nourinejad and Roorda, 2014, Nourinejad and Roorda, 2015, Pavone et al., 2011, Smith et al., 2013]. [Barth and Todd, 1999] develop a simulation model of car-sharing operations and conclude that a sufficient fleet size for satisfying customers is 3-6 vehicles for every 100 trips but that 18-24 vehicles per 100 trips are required to minimize relocation costs. This conclusion shows the importance of robustness of FMS. [Fan et al., 2008] propose a multi-stage stochastic linear integer model which attempts to capture system uncertainties such as car-sharing demand variation. The objective function of their model maximizes revenues from servicing customers while minimizing the cost of vehicle relocation. [Kek et al., 2009] design a mixed integer linear program to determine a set of near-optimal manpower and operating parameters for the vehicle relocation problem. Simulation tests, based on a set of commercially operational data for the city of Singapore, indicate that optimization of manpower can reduce staff expenses by up to 50% and zero vehicle time (duration of vehicle shortage at parking stations) by up to 13%. The model however cannot be applied for a real-time operation. [Pavone et al., 2011] and [Smith et al., 2013] show a theoretical solution to fleet sizing by introducing rebalancing assignments that minimize the number of empty vehicles traveling in the network and the number of rebalancing drivers needed, while ensuring stability. They introduce a rebalancing policy based on a fluidic model. Using both theoretical and simulation results, the authors determined the minimum number of vehicles required to maintain system stability. [Correia and Antunes, 2012], on the other hand, focus on a depot location in one-way car-sharing systems where vehicle stock imbalance issues are addressed. Considering all decision variables (depot locations, satisfying demand and relocation), the authors present a mixed integer optimization approach to maximize revenue of the car-sharing operator while minimizing operating cost such as vehicle maintenance, parking provision, vehicle depreciation, and vehicle relocation. Based on the case study of Lisbon, their results show that the depot location and trip selection schemes have an impact on the profitability of such systems. [Jorge et al., 2014] builds their work on [Correia and Antunes, 2012] and focuses on optimal rebalancing strategies for one-way car-sharing systems. The authors applied their static relocations algorithms to network of stations in Lisbon and their results show that relocating vehicles produces significant increases in profit. [Nourinejad and Roorda, 2014] propose a dynamic optimization-simulation model for one-way car-sharing. The authors analyze the trade-offs between the vehicle relocation and fleet size using the Autoshare—a car-sharing company in Toronto—data. To perform the analysis they run a dynamic vehicle relocation optimization, which was embedded in a discrete event-based simulation. They came to two (2) conclusions: (i) a bigger fleet requires less relocation hours to service the same demand and vice versa and (ii) if we extend the reservation time (meaning allowed wait time of customers), then we observe a significant reduction in the fleet size. [Nourinejad and Roorda, 2015] present an extension to model in [Nourinejad and Roorda, 2014]. Their model consists of two (2) sequential optimization problems:(i) fleet size problem, which finds the optimal fleet size when all demand is answered, and (ii) profit maximization model, which maximizes car-sharing profits when the fleet size is constant. They introduce a hybrid car-sharing system containing features of both one-way and two-way systems and compare three of them against each other. Their results suggest that the one-way system requires the smallest fleet size but highest vehicle relocation hours. On the other hand, [Bruglieri et al., 2014] explores methods to forecast the balancing trips of a one-way car-sharing system of electric vehicles and applies them to the city of Milan, Italy. The authors solve a mixed integer program to redistribute the vehicles. Their numerical results on a test bed of 30 vehicles show that two workers with a duty time of 5 hours are sufficient to satisfy about 86% of the relocation requests. [Barrios and Godier, 2014] presents three different redistribution strategies (zero, periodic and continuous redistribution) for station-based and free-floating car-sharing using an agent-based simulation approach. The authors showed that without changes in percentage of satisfactorily served demand, continuous redistribution of vehicles results in a reduction in the required fleet size as compared to zero-redistribution and periodical redistribution strategies. [Brownell and Kornhauser, 2014] evaluated fleet sizing for an autonomous Taxi system. The authors evaluated two models: (a) personal rapid transit, in which customers were served by the same vehicle if they arrived at a station within a time window and their origin and destination stations were the same, and (b) smart paratransit, where vehicles were re-routed to pick-up additional customers. For both models, stations were established in a grid. The authors presented upper and lower bounds for the fleet size required for both models.

1.3 Contribution

A main motivation for development of Autonomous Mobility on Demand systems is a sustainable urban transportation. However, no standard methodology has been established to accurately and consistently design and evaluate this new service. Existing methods for operating as well as modeling urban transport require extensions to incorporate AMOD systems as benefits analysis requires controlled experiments that compare transportation behavior with and without the new mode. To the best of our knowledge, most of the existing literature treats problems of depot location, fleet size, assignment and redistribution, separately. In this dissertation we attempt to correlate and merge together these different problems of fleet management of autonomous mobility on demand systems. This work aims to look at the feasibility of assembling a fleet of autonomous vehicles for a mobility on demand service in order to ease congestion, increase safety, and reduce environmental impact in the city of Singapore. The key part of this undertaking relies on the agent-based simulations of the estimated traffic.

We set up this dissertation to assess and demonstrate the role of autonomy in mobility on demand systems and its impact in terms of feasibility and efficiency through modeling, algorithms development, simulations, and experimental demonstrations.

Based on the literature review, which is presented in four chapters (Chapter 1, 3, 4, 5), first novel contribution presented in this thesis detailed literature review of more than 100 publications on fleet management systems for flexible mobility services.

Second contribution is development of the controller for the fleet of autonomous taxis and analysis of a potential implementation of such system. To the best of our knowledge, there is no such contribution in the current literature. The fleet controller presented in this dissertation is simulator-independent and can also be used as a manager of a fleet of real autonomous vehicles.

Third contribution of the thesis is development of algorithms related to the operation of autonomous vehicles for mobility on demand systems. Although the algorithms may be similar to those for human-driven taxis, the AV-specific implementation gives us full control over the vehicles, which inherently changes the perspective of the operator. There is growing number of publications in the current literature related to operation of autonomous taxis, however non of the existing work tackles the problem as a whole: from designing where to operate (long term decisions), to simulating second-to-second operation of the system. To the best of our knowledge, there is no such development in the literature which combines together all operational perspectives of designing an AMOD system: problem of depot location, fleet size, assignment and redistribution, separately.

Fourth contribution is a unique approach to rebalancing of autonomous vehicles, which consists of static and dynamic models. Static model gives us a rough idea where and when we need vehicles, which is estimated based on taxi travel pattern, while the dynamic model serves as an adjustment (final tune) to the static model.

In this dissertation we identify the following three design and management levels of introducing AMOD system:

(i) At the planning level we decide on strategic decisions, such as the location and size of operating area and the number of stations (zones). These decisions are considered as a long-term planning decisions and they cannot be easily changed after the system is implemented.

(ii) At the tactical level we decide on long-term operation decisions. These decisions include the number of vehicles which are required to run the system, *home* location of the vehicles at the beginning of the day and static pricing.

(iii) Operational level (short-term operation decisions) include assignment and rebalancing methods, and dynamic pricing. This decisions can vary between different days and within days.

Performing a sequential analysis of each level, based on the case study of the city of Singapore, we answer the following questions:

(i) Where to operate the system and where to locate depots (stations) for AMOD vehicles?

(ii) What fleet sizes are optimal to serve demand? Most of the current studies assume a fixed rate of trips which are served by MOD (car-sharing, taxi or aTaxi) systems. This fundamental aspect of determining the appropriate number of vehicles is derived from the travel patterns using the simulation-based approach.

(iii) What vehicles assignment and route choice policies are likely to improve system performance? In order to manage an extensive fleet size of vehicles in real-time, we designed a robust dispatcher and route planner for AMOD system.

(iv) How to perform rebalancing of the vehicles? AMOD systems offer a convenience of boarding and alighting at any location, which may cause a deficit of vehicles in some areas and surplus in other. To effectively move vehicles between different locations, while minimizing the demand loss, a rebalancing mechanism is required. Our algorithm performs two types of rebalancing: offline, which distributes vehicles once a day (overnight) and online, which makes a real-time adjustments to to the offline solution.

(v) What ride-sharing policies will improve the system? Ride-sharing is a more economical, convenient, and sustainable way to get around, particularly for short point-to-point movements. We have developed a static ride-sharing algorithm, which shows a decrease in the total vehicle-kilometer-traveled.

(vi) How the dynamic pricing will influence the mode choice of customers? A simple choice model and dynamic pricing model have been developed to assist in managing the demand and supply.

Our approach is simulation-based and we make use of SimMobility—an agent-based simulation platform. The proposed methodologies are applicable to a large-scale AMOD systems for the extended Central Business District in Singapore. Experimental results are presented and discussed.

1.4 Thesis Outline

This thesis is organized as presented in Figure 1.6.

In Chapter 1, we present trends in mobility on demand services and how the fleets of shared-mobility systems are managed, which set up background for our work. Next, we state what is the contribution of this dissertation and provide outline of the thesis.

In Chapter 2, we provide detailed description of our development which is fleet management systems for autonomous mobility on demand service (we call it AMOD Controller). We

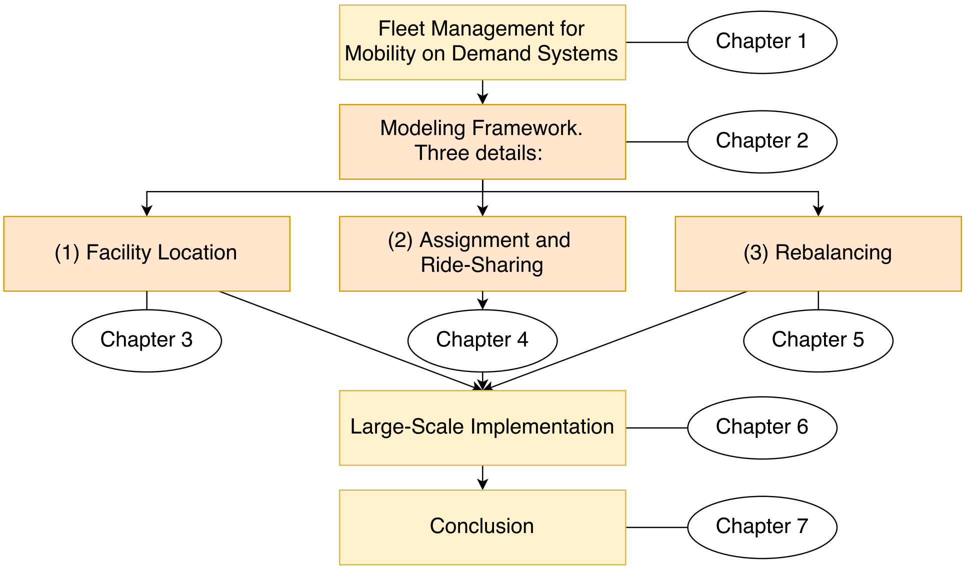

Fleet Management for Mobility on Demand Systems
Chapter 1
Modeling Framework. Three details:
Chapter 2
(1) Facility Location
(2) Assignment and Ride-Sharing
(3) Rebalancing
Chapter 3
Chapter 4
Chapter 5
Large-Scale Implementation
Chapter 6
Conclusion
Chapter 7

Figure 1.6: Organization of the thesis. Note that Chapter 2 presents an overview of the AMOD Controller, while Chapter 3, 4, 5 show detailed representation of the models introduced in Chapter 2.

evaluate different modeling approaches of analytical and simulation models. We present paradigms in simulation modelings and review traffic simulators, which serves us as a justification of choosing SimMobility over other available software. Finally, we introduce the reader to simulation platforms and AMOD Controller. We provide justification to use SimMobility as a tool to simulate autonomous vehicles and provide implementation details of extending SimMobility with AMOD Controller.

In Chapter 3, we focus on the strategic aspect of management of AMOD systems—models for selecting the number and location of new facilities. By facilities we understand stations (distribution centers or car parks) for autonomous vehicles. In this chapter we provide classification of facility location problems followed by related work in the field. We formulate facility location models for an AMOD system as (i) distribution of stations based on location factor rating, (ii) one station for the entire system, (iii) stations based on the set covering problem, and (iv) stations based on maximal coverage problem. We evaluate our models by providing results and discussion. Finally, we acknowledge limitation of our models by stating future directions.

In Chapter 4, we formulate the fleet assignment and routing problem. Given a distribution (location) and capacity of available vehicles, the fleet assignment problem faced by the operator is to determine which vehicle should pick up which customer, while the fleet routing is to determine the route for each assigned vehicle. We first give introduction to the assignment problem by providing definition, terminology and related work. Next, we describe how the assignment is implemented within AMOD Controller and we describe implemented algorithms for assignment with and without ride-sharing. We extend the framework to demand management with price incentives and present results comparing different assignment algorithms. We discuss the results and acknowledge limitation of our work by stating future directions.

In Chapter 5, we show our formulation of empty vehicles rebalancing problem, which aims in finding optimal control policies for realigning demand and supply. We provide terminology, definitions, and related work in the filed of empty vehicle rebalancing. Next, we provide explanation how the AMOD Controller handles rebalancing and what algorithms are used for static and dynamic methods of redistributing the vehicles. We present comparison of rebalancing policies, discussion and directions for the future work.

In Chapter 6, we present a large-scale implementation of the AMOD system in the Central Business District in Singapore. We present two alternatives of introducing AMOD service in Singapore: (i) case study of the central business district in Singapore, and (ii) case study of the extended central business district in Singapore. We describe simulation setup, results and discussion for each case study. We summaries our results and provide directions for future work.

Finally, in Chapter 7, we present conclusions of the thesis and directions for future work.

Chapter 2

Autonomous Mobility on Demand (AMOD) Controller

This dissertation addresses the problem of design and analysis of an autonomous mobility on demand system. More specifically, we focus on the strategic and operational decisions which are required to run the AMOD service. Our approach is simulation-based, with some parts of the problem solved analytically. The simulation framework is modular and each module is responsible for a different task, e.g., facility location, assignment, rebalancing. This formulation encompasses several decisions under one framework by integrating the key strategic and operational issues that may emerge during regular operation of the system. To the best of our knowledge, most of the existing literature treats problems of depot location, fleet size, assignment and redistribution, separately. In this dissertation we attempt to correlate and merge together these different problems of autonomous vehicles' fleet management systems. If operational decisions are taken into account when considering strategic decisions, this may lead to a more efficient management of the system [Boyaci et al., 2015]. For this reason, we develop AMOD Controller—an experimental research platform to test models and algorithms for (i) planning, (ii) strategic, and (iii) operational decisions of an autonomous mobility on demand system.

AMOD Controller is designed as a hybrid software framework, including both analytical approach and event-driven discrete time-step simulation. It is platform-independent framework for managing a fleet of autonomous vehicles.

This chapter describes framework of AMOD Controller and how it is integrated with simulation platforms. The chapter is organized as follows. Modeling approaches are presented in Section 2.1. Introduction to SimMobility Platform is covered in Section 2.2. Section 2.3 and Section 2.4 present framework and software design of AMOD Controller. Run-time operation of AMOD Controller is described in Section 2.5.

2.1 Modeling Approaches

Road traffic becomes a challenge for urban planners. To better predict outcomes of introducing new solutions for the transportation needs, we model it first to check the robustness of the proposed implementations. In this section we describe the modeling approaches used within AMOD Controller.

2.1.1 Analytical and Simulation Models

We distinguish analytical and simulation models. Understanding capabilities of these two approaches is of great practical interest to system modelers. In analytical, or static, model we identify system variables (parameters) and evaluate them based on sets of equations relating these variables. However, analytical solution does not always exist, or may be very hard to find [Borshchev and Filippov, 2004]. Alternatively, simulation, or dynamic, modeling may be considered. Simulation can be defined as a set of rules (e.g. equations, flowcharts, state machines, cellular automata) that define how the system will change in the future, given its present state [Borshchev and Filippov, 2004]. Simulation executes the model through (discrete or continuous) state changes over time. An example of traffic simulation can be defined as a dynamic representation of the real world through the application of computer software [Pursula, 1999]. For complex problems where time dynamics is important, simulation modeling is a better answer, because it can study models too complicated for analytical or numerical analysis alone.

Abstraction levels [Traffic] simulation software can be classified as microscopic, mesoscopic and macroscopic depending on the simulated level of details and timescale [Pursula, 1999].

(i) *Micro-simulation* is a so called *physical* modeling where individual objects with exact sizes, distances, velocities and timings matter [Borshchev and Filippov, 2004]. It represents a highly detailed model of activities such as individuals' movement and their interactions between each other. This simulation technique, when applied to traffic simulations, relates directly to traffic flow theory and utilizes equations governing driver behavior such as gap acceptance, lane changing or car following models [Espada et al., 2010]. The driving pattern of each individual car is generally influenced by the surrounding cars in the model and the rules for modeling this impact can be chosen by the operator. Due to its level of details, this technique is the most resource intensive [Espada et al., 2010].

(ii) *Mesoscopic simulations* are based on aggregate traffic movements, where vehicles are represented as a traffic stream or a network density map. To reduce modeling complexity no single driver's behavior is directly modeled, but incorporated in to the

Table 2.1: Types of simulations in transportation

Time	State	Space		
		Continuous	Discrete	N/A
Cont.	Discrete	Real transportation systems, e.g., dynamic traffic assignment		Discrete event systems, e.g., queuing
	Cont.	Traffic flow models based on partial differential equations		Open dynamics engine methods, e.g., vehicle motion, fluid approx. for queuing
Discrete	Discrete		Cellular automata	Discrete event simulation
	Cont.	Car following models Microscopic traffic flow models	Numerical methods based on Partial diff. equations	Numerical open dynamics engine methods
N/A	Discrete or Cont.	Monte Carlo method		Econometric models, e.g., trip generation, traffic assignment

model as a consideration of an overall interacting traffic. Results are provided as graphic visualization of speed, flow, density, queue-length for each lane and movements in intersections.

(iii) *Macro-simulations* represent the traffic at the most aggregate level of details. Macro-simulations are typically understood in terms of aggregate values, global feedbacks and trends [Borshchev and Filippov, 2004]. System analysis based on macroscopic assignment provides a *big picture* and allows simulating a high-level decisions such as land use planning.

Simulations rely on the concept of system state, which describes the evolution of the system over time. System state can be either discrete or continuous. Traffic simulation models are classified according to discrete and continuous time, state, and space. Basic division of the simulation types is presented in Table 2.1.

2.1.2 Paradigms in Simulation Modeling

The major paradigms in simulation modeling are: System Dynamics (SDM), Dynamic Systems (DSM), Discrete Event (DEM) and Agent Based (ABM) Modeling [Borshchev and Filippov, 2004].

(i) *System Dynamics Modeling* simulates the real-world processes in terms of stocks (e.g., material, people, money), flows between these stocks, and information that determines the values of the flows. To approach the problem in SDM style one has to describe the system behavior as a number of interacting feedback loops, typically in the form of differential equations [Borshchev and Filippov, 2004].

(ii) *Dynamic Systems Modeling* is used in technical engineering disciplines as a standard part of the design process. The mathematical models of a dynamic system consist of state variables and algebraic differential equations over these variables. Opposite to the SDM, the variables in DSM have direct *physical* meaning: location, velocity, acceleration, pressure, etc., they are inherently continuous, and are not aggregates of any entities [Borshchev and Filippov, 2004].

(iii) *Discrete Event Modeling* is based on the concept of entities, resources and block charts describing entity flow and resource sharing [Borshchev and Filippov, 2004]. Entities are passive objects representing objects such as people, documents, tasks. They travel through the flowchart where they are processed.

(iv) In *Agent-Based Modeling*, a system is modeled as a collection of autonomous decision-making entities called agents [Helbing, 2012, Bonabeau, 2002]. The crucial feature of an agent in ABM is being decentralized. Compared to SDM or DEM, in AB model we do not declare any global system behavior (dynamics). Instead, we define behavior at the individual level, and the global behavior emerges as a result of many individuals, each following its own behavior rules while living together in some environment [Borshchev and Filippov, 2004]. These dynamically interacting rule-based agents can create a system with real-world-like complexity. In some ways, agent-based models complement traditional analytic methods. Where analytic methods enable humans to characterize the equilibria of a system, agent-based models allow the possibility of generating those equilibria. This generative contribution may be the most mainstream of the potential benefits of agent-based modeling.

In terms of time space, SDM and DSM work mostly as continuous processes, while DEM and ABM as discrete time, i.e. jump from one event to another.

Our approach Based on our expertise, ABM allows us to capture real life phenomena better than other approaches do. Therefore, a core-component of AMOD Controller is design such that it can interact with agent-based simulators. AMOD Controller contains active objects such as customers and vehicles, who are characterized by timing, event ordering and individual behavior as in the ABM framework. Additionally, we make use of the analytical methods to estimate demand for simulated case-studies (as presented in Chapter 6).

In the next section (Section 2.1.3), we review open-source and commercial agent-based microscopic simulation platforms and present our findings.

2.1.3 Review of Traffic Simulators

In order to keep track of the AMOD vehicles, it is necessary to simulate them individually. For performing empirical tests on individuals' level, a micro-scale simulator is of great interest to modelers. Although a variety of transportation simulation tools do exist, e.g., Aimsun, Coresim, MATSim, Paramix, PTV Vissim, SimMobility, SimTraffic, Transims, only a few allow taking into account some specific requirements [Maciejewski and Nagel, 2014, Garcia-Palomares et al., 2014]. AimSun [Aimsun, 2016], is a commercial microsimulation software developed by TSS-Transport Simulation Systems. It is built as a hybrid simulator providing simultaneous a more detailed time-sliced microscopic model and an event-based mesoscopic model [Aimsun, 2016]. This allows to model large areas while zooming in on all areas that require a finer level of detail. MATSim, *Multi-Agent Transport Simulation* [MATSim, 2016], is an agent-based system for transport simulation with the primary focus on transportation planning. It allows disaggregated activity-based modeling that consists of three main phases which are run iteratively: planning, traffic flow simulation, and scoring [Maciejewski and Nagel, 2014]. MITSimLab, *MIcroscopic Traffic SIMulation Laboratory* [Mitsim, 2016] is an open-source application developed at MIT. Its core models have been written in C++. It consists on three main components: Microscopic Traffic Simulator, Traffic Management Simulator and Graphical User Interface. All three modules interact with each other. PTV Vissim, *Verkehr In Stadten–SIMulations modell* [Vissim, 2016] is a commercial microscopic agent-based multi-modal traffic flow simulation software. VISSIM is one of the most widely used simulation tool to validate new transportation policies and control systems [Garcia-Palomares et al., 2014, Marczuk, 2010]. It can model integrated roadway networks, as well as different modes such as buses, light rail, heavy rail, trucks, pedestrians, and bicycles. The software may be used to create detailed computational results or 3D animations for different scenarios. Paramics, *PARAllel MICroscopic Simulation* [Paramix, 2016] is a commercial software integrating a microscopic traffic modeled dedicated for the simulation of congested traffic networks. Paramics is designed to handle scenarios ranging from single intersection up to an entire city traffic system [Paramix, 2016]. SimMobility [Adnan et al., 2016] is a multi-scale simulator that considers land-use, transportation and communication networks along with individual choices and decisions at different levels of resolutions: from detailed traveler movements to day-to-day and year-to-year travel decisions. SimMobility is under ongoing development and it is an open-source software based on a distributed C++ implementation.

SUMO, *Simulation of Urban MObility* [Sumo, 2016], is an open source, microscopic and continuous traffic simulation package developed at the German Aerospace Center. It is designed to handle large road networks. SUMO allows modeling of inter-modal traffic systems including road vehicles, public transport and pedestrians.

Our approach Our preferred choice of simulator is an open-source platform. It is important for our research to know how the software is implemented and configured. The open source programs are, in general, more flexible for the user as compared to commercial counterparts. Users of open-source platforms can easily examine the source code and make the necessary alterations to the code based on their desired behavior of the system. This also allows us to potentially identify any problems in the system and to make our fixes to the software to rectify the problem. In the commercial software the source code is usually not available for the user, in which case we may not be aware of what exactly affects the performance. Among the non-commercial simulation platforms, we have simulators such as SUMO, MATSim, MITSIMlab and SimMobility.

Another consideration is the level of details of the simulator. Our study evaluates the fleet management of a new transportation service, which is an autonomous mobility on demand. In short, the fleet management system consists of the reservation procedures (related to assignment), routing and rebalancing. Therefore, to evaluate the impact of different implementation policies the level of details may play an important role. Based on our experience, MATSim simulates the system at a mesoscopic level. It does not consider lanes and detailed interactions between vehicles. SUMO due to its simplicity is neither distributed nor parallel so the simulation with a sophisticated fleet management of thousands vehicles may be prohibitively time consuming. SimMobility is built on MATSIMLab and extends it. It is an agent-based and multi-modal platform with parallel and distributed architecture. Therefore, for this work, *SimMobility* is selected and used. A general overview of SimMobility is presented in the following section (Section 2.2).

2.2 SimMobility Platform

SimMobility is a multi-scale simulator that considers land-use, transportation and communication networks along with individual choices and decisions at different levels of resolutions: from detailed traveler movements to a day-to-day and a year-to-year travel decisions. SimMobility is under ongoing development and it is an open-source software based on a distributed C++ implementation. SimMobility is an agent-based simulator. The individual travel behavior is modeled under an activity-based formulation, where each agent's daily activities and its impact on the transportation systems are simulated [Lu et al., 2015]. Each individual person is represented as an agent in the model, which helps in simulating how people will react in the uncertain future. To support multi-modality SimMobility explicitly simulates private traffic, public transit, pedestrian traffic as well as freight transportation, and allows agents to switch between these modes over the course of a given day.

SimMobility integrates various mobility-sensitive behavioral models within a multiple time-scale structure, comprised of three simulation levels which are differentiated by the time-frame.

The high-level design of SimMobility is shown in Figure 2.1.

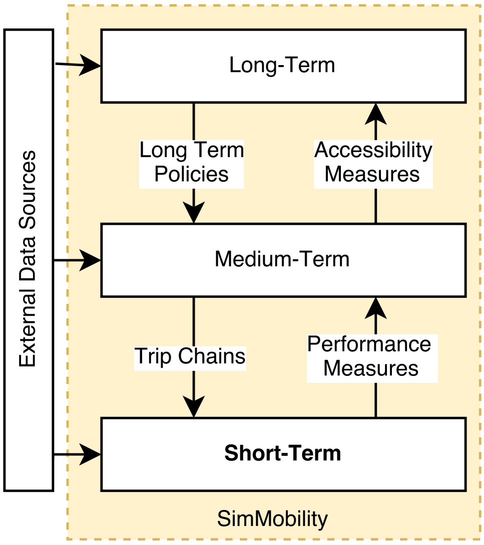


Figure 2.1: High-level design of SimMobility. SimMobility comprises three primary modules differentiated by the time-frame.

The long-term simulator models year-to-year changes. It captures land use and economic activity, with special emphasis on accessibility. It predicts evolution of a land use and property development, determines associated life cycle decisions of agents, and accounts for interactions among individuals and firms. **The mid-term** simulator models day-to-day changes. It handles transportation demand for passengers and goods. It simulates activity and travel patterns of agents. The mid term represents moving vehicles in aggregate, and routes are generated by behavior-based demand models. **The short-term** simulator (SimMobility ST) operates at the operational level. It simulates movement of agents at a microscopic granularity. It synthesizes driving and travel behavior in detail and also interacts with a communication simulator that models the impact of a device to device communication on these behaviors. Similarly, the code structure and functions are shared by the three levels, assuring consistency among sub-models. Short-term level shares several mobility decisions with the mid-term level, as described in![Lu et al., 2015].

The three levels can work together as well as independently. When they are run independently they only require a prior knowledge from different level, e.g., if an update on accessibility is required for the long-term simulator, it calls the mid-term level. The key to multi-scale integration in SimMobility is a single database model that is shared across all levels [Adnan et al., 2016]. Every agent exists across all levels and in this way the impact from one level is propagated to the others. SimMobility is entirely developed in C++, using boost threads library, for parallelization, and MPI (message passing interface) library, for distribution [Adnan et al., 2016]. SimMobility takes advantage of state-of-the-art computational efficiency tools to increase scalability, i.e., it allows network decomposition with MPI distribution, and it is able to do runtime load balancing by taking advantage of individual agent's context, e.g., agent of similar type can be grouped together.

Model validation and calibration within SimMobility SimMobility consists of three pillars: long-term, mid-term, and short-term, each of them simulating the same agents at different spacial and temporal resolutions.

Demand estimation and calibration for AMOD Controller takes place within SimMobility Mid-Term (MT). The MT simulator takes multi-modal network and population, which may come from SimMobility Long-Term (LT) simulator or other population data sets, as an input. Population data set contains detailed characteristics of each agent (Appendix A.1.1). As an output, it passes accessibility measures (in the form of Logsums) from the pre-day component of MT simulator to the LT simulator. The MT simulator provides the ST simulator with trip chains as input demand to simulate smaller network regions in more detail (Appendix **??**). SimMobility ST applies several design heuristics to make modeling and development easier for a heterogeneous user base. As we have described in [Azevedo et al., 2017]:

1. Entities are isolated from each other, and can only interact through properties that are shared among them, which is a typical agent-based simulation framework.
2. The simulator is location-agnostic regarding agents. In other words, an agent's interface is not affected by the agent's location on the network.
3. Time step within SimMobility is indivisible and agents are assumed to all tick forward at once.
4. SimMobility is hierarchical and provides sensible defaults, e.g., trip-chains can be filled in with more information as the agent's trip progresses.

SimMobility uses an activity-based demand formulation in the form of activity-schedules rather than the traditional Origin-Destination matrix definition. In such approach, trip chains are generated by individuals' daily schedules instead of aggregated traffic specific matrices. Demand parameters are calibrated through tuning of the OD flows by calibrating one parameter for each activity schedule [Azevedo et al., 2017]. Then the updated OD parameters are converted into trip chains by dis-aggregating through so-called *killing-cloning process* for each iteration. There are also parameters related to the route choice, which are calibrated as described in [Azevedo et al., 2017, Adnan et al., 2016]. Further details on SimMobility, in terms of modeling details, integration, and calibration can be found in [Adnan et al., 2016, Azevedo et al., 2016, Azevedo et al., 2017], Appendix D, and Appendix E.

In this dissertation we make use of SimMobility ST as a tool to simulate autonomous vehicles' traffic, which is presented in the following section.

SimMobility Short-Term SimMobility Short-Term (SimMobility ST) is an agent-based, multi-modal microscopic simulator [Adnan et al., 2016]. It captures agents' movements at resolution of up to 100 milliseconds. SimMobility ST comprises three main components as shown in Figure 2.2. The core traffic simulation model of SimMobilityST (in Figure 2.2 Microscopic Traffic Simulator) is based on the microscopic simulation tool MITSIM [Yang et al., 2000]. The structure of the Microscopic Traffic Simulator is detailed in Figure 2.12. Traffic simulator simulates high level decisions, e.g., route choice, and low level decisions, e.g., movement decisions, such as car following and lane-changing. Control and Operation Systems simulates the control centers, e.g., traffic and parking controller, bus, rail, and autonomous vehicle controllers. The controllers influence the movement simulator. At the current state of SimMobility, the bus controller, traffic signal controller and autonomous vehicle controller are operational [Adnan et al., 2016]. The autonomous vehicle controller is a contribution presented in this dissertation. The third component of SimMobilityST is the Communication Network Simulator, which simulates an agent-to-agent communication, e.g., via mobile phones or a vehicle-to-vehicle infrastructure.

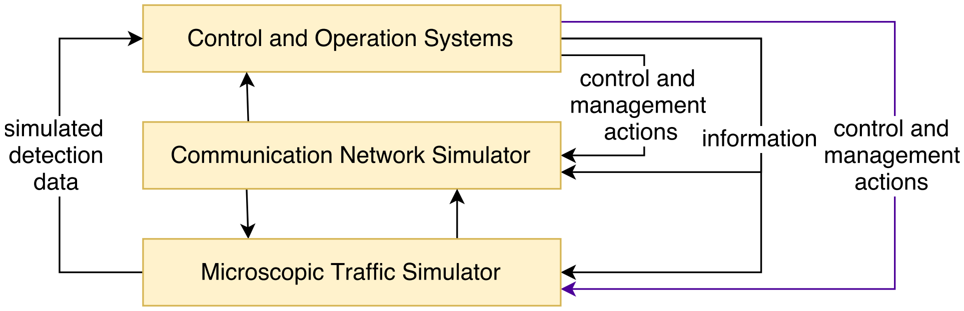

Control and Operation Systems
Communication Network Simulator
Microscopic Traffic Simulator
simulated detection data
control and management actions
information
control and management actions

Figure 2.2: Framework of SimMobility Short-Term. SimMobility ST comprises three primary components: (i) Microscopic Traffic Simulator, (ii) Control and Operation Module, and (iii) Communication Network Simulator.

2.3 General Framework of AMOD Controller

The AMOD Controller framework consists of a core component and communication infrastructure (Figure 2.3). The core component is responsible for the fleet management of AVs, e.g., assignment, routing, rebalancing, and has been implemented in *C++* with some parts written in Matlab. The communication component handles connections with the simulators and is written in *C++*.

AMOD Controller is a detachable component which is responsible for the fleet management of

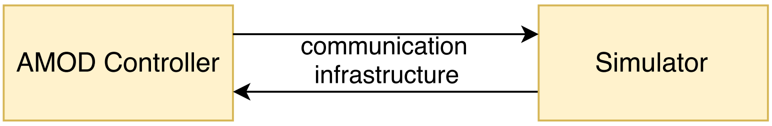


Figure 2.3: Design of AMOD Controller. The AMOD Controller design consists of a core component and communication infrastructure. It is a detachable component which is responsible for the fleet management of autonomous mobility on demand system. This design allows us to test the controller within different simulation frameworks. AMOD Controller sends dispatches to a microscopic simulator, which handles vehicle movements.

autonomous mobility on demand system. This design allows us to test the controller within different simulation frameworks. The basic idea is that AMOD Controller sends dispatches to a microscopic simulator, which handles vehicle movements and provides tracking information for all vehicles in the system. The objective of this approach is to design a versatile manager which can be used as an extension of any system (simulator). It aims to provide a tool which helps us to understand how different system setups of AMOD system impact its efficiency.

The overall architecture of AMOD Controller is presented in Figure 2.4. The controller's framework is modular and each module is responsible for different task, e.g., facility location,

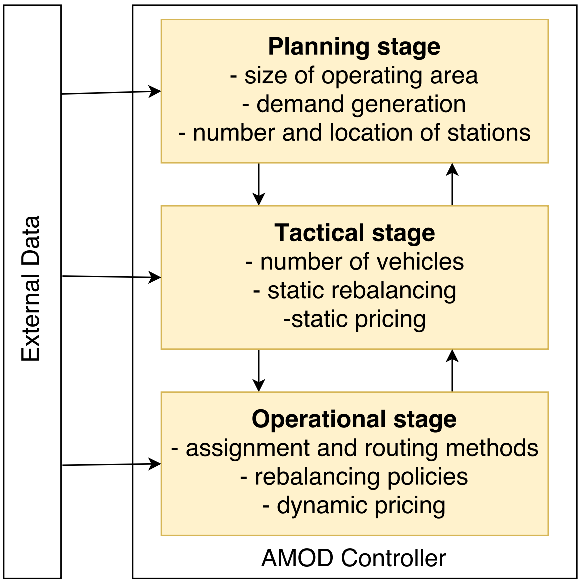


Figure 2.4: General framework of AMOD Controller. AMOD Controller is run at three (3) stages. Each stage requires external data, such as the network and demand information to be fed into the controller. At the planning (strategic) stage we determine the size of operating area, estimate the demand and the number of stations (depots), where we park the vehicles. At the tactical (long-term operational) stage we focus on the number of vehicles, which are required to run the system, and static pricing. At the last stage we decide on short-term operational decisions, which include methods for assignment, routing, rebalancing, and dynamic pricing.

assignment, rebalancing. AMOD Controller has been designed to run at three stages. At each stage demand information are fed into the controller. At the planning (strategic) stage we determine the size of operating area, estimated demand, and the number and locations of stations (depots), where we park the vehicles. This stage is evaluated first and the solution is passed to the tactical stage. At the tactical stage (equivalent to long-term operation decisions) we focus on the number of vehicles which are required to run the system, static rebalancing, and pricing. At the last stage we decide on short-term operational decisions, which include dynamic pricing, assignment, routing, and dynamic rebalancing policies. The performance measures from the operational stage are fed back to the tactical level and the planning level. Based on the output from the operational level we make adjustments to the tactical and planning level and repeat the loop. In this framework we do not treat each decision level separately as the strategic and tactical decisions influence directly operational decisions. When considering strategic and tactical decisions, we take into account operational matters. Mathematical formulations of each component are presented in Chapter 3, Chapter 4, and Chapter 5.

A vast majority of AMOD Controller has been implemented in *C++* with some components written in Matlab. There are few benefits of using *C++* environment. One of them is high portability of *C++* language, which allows to develop software irrespective of hardware and operating system. It compiles into highly optimized CPU-specific machine code with little or no runtime overhead. Finally, as an object-oriented programming language, it provides a clear modular structure, which is easy to maintain and modify.

2.4 System Architecture

Here, we describe the system architecture of AMOD Controller. AMOD Controller is designed in a modular manner. The modularity in designing AMOD Controller allows us to easily test different scenarios and management algorithms by simply replacing necessary sub-modules of the controller. Additionally, AMOD Controller is a detachable component and can be easily tested within different simulation frameworks.

In the next paragraphs, we describe architecture of different management levels within AMOD Controller: planning, tactical and operational.

Planning Level At the planning level of AMOD Controller, *AMOD Controller PL*, we decide on (i) the area where we want to deploy AMOD system), and (ii) the station locations. The architecture of AMOD Controller at the planning level is presented in Figure 2.5. To successfully run AMOD Controller at the planning level we require external sources of information, such as network information and trips data. Based on the input we can decide on the region where we can potentially implement the AMOD service. Then the demand

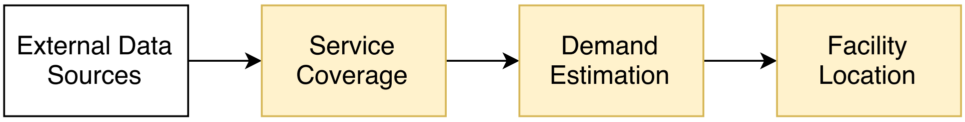


Figure 2.5: Architecture of AMOD Controller at the planning level. The network information and trips data (predicted demand for AMOD system) are fed to the Controller. Based on the input, we decide on the operating area and the number and location of stations (zones) for AMOD vehicles.

generation for the region is performed and based on the demand, we estimate the number of zones and location of the distribution centers (facilities). The zones are not necessarily equal in size, but we assume that each zone *hosts* exactly one station for the AMOD vehicles. The stations are understood as distribution centers, where parking and charging facilities for the vehicles are provided. One can understand the function of facilities for AMOD vehicles similarly to the function of bus interchanges for the public transport buses. The service coverage and demand generation are inter-correlated in our model and the detailed description of the method is provided separately for each case study in Chapter 6. The facility location models are more generic and explained in Chapter 3.

Tactical Level At the tactical level of AMOD Controller, *AMOD Controller TL*, we estimate (i) the number of vehicles required to efficiently move people around, and (ii) counts for the static rebalancing. The architecture of AMOD Controller at the tactical level is presented in Figure 2.6. The number of vehicles required to run the service is optimized based on the network data, the estimated demand and the location of stations. Counts for static rebalancing between the stations are obtained from the model presented in Section 5.3).

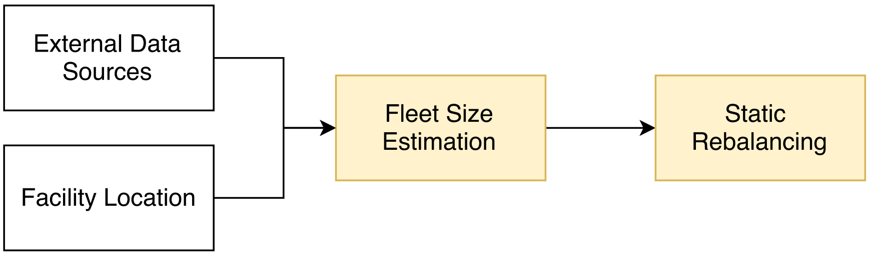


Figure 2.6: Architecture of AMOD Controller at the tactical level. The network information and trips data are fed to the Controller based on which service coverage and the number and location of zones are decided.

Operational Level The models and algorithms within the operational level of AMOD Controller, *AMOD Controller OL*, are organized into three main components: initialization, online fleet management and vehicle tracking modules (Fig. 2.7). During the initialization

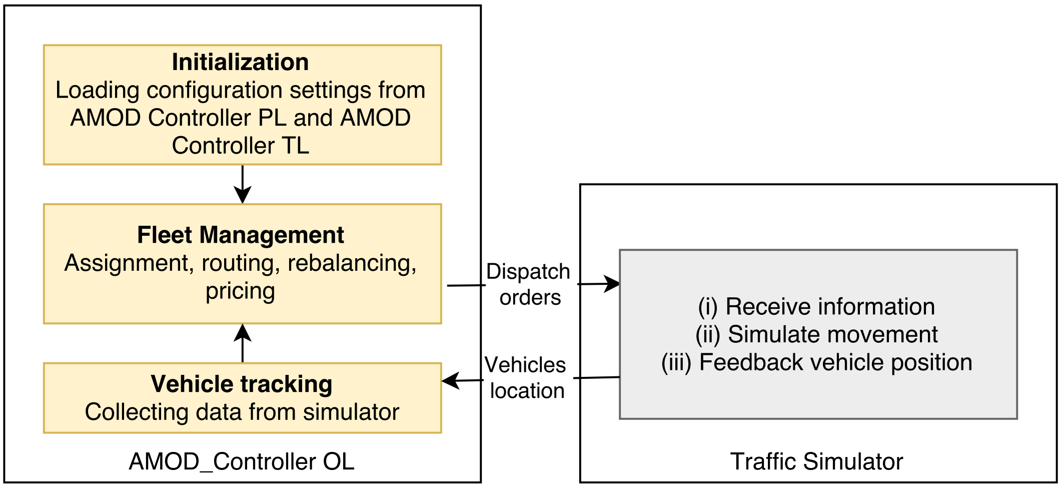

Initialization
Loading configuration settings from AMOD Controller PL and AMOD Controller TL
Fleet Management
Assignment, routing, rebalancing, pricing
Vehicle tracking
Collecting data from simulator
AMOD_Controller OL
Dispatch orders
Vehicles location
(i) Receive information
(ii) Simulate movement
(iii) Feedback vehicle position
Traffic Simulator

Figure 2.7: Architecture of AMOD Controller at the operational level. AMOD Controller OL comprises of three main components: initialization, online fleet management and vehicle tracking. It dispatches vehicles to the traffic simulator and keeps track of their location and statuses.

phase the configuration parameters from AMOD Controller Pl and AMOD Controller TL are loaded and the connection to the database with the booking requests is established. The online fleet management component establishes connection with Traffic Simulator and if the connection is successful, it sends dispatches, e.g., requests Traffic Simulator to simulate vehicle movement from its origin to the pick up point and from the pick up to the destination. The online fleet management module is a core component of AMOD Controller and is responsible for assignment, routing, rebalancing and pricing. AMOD Controller dispatches orders to the simulator, which performs a simulation of the vehicles and returns vehicular information (e.g., speed and location) to the vehicle tracking component that captures and logs the results.
Next section describes functional organization of AMOD Controller at the Operational Level.

Functional Organization of AMOD Controller at the Operational Level AMOD Controller OL is a centralized system, where the entire communication goes through a server as presented in Figure 2.8. As presented in Figure 2.8 the ride requests (bookings made through an app) are stored in the database. Each booking request consists of a time stamp, customer id, origin and destination location. The AMOD server constantly pulls booking information from the database. If there are no vehicles available, the booking is stored in a booking queue (for a limited time). If the booking is not served within a predefined time, it is discarded and logged as a *not served trip*. If there are vehicles available, then the assignment algorithm (Chapter 4) is executed. Assigned vehicles are sent to the dispatcher, which communicates the trips to the traffic simulator. To manage the AMOD system, the server

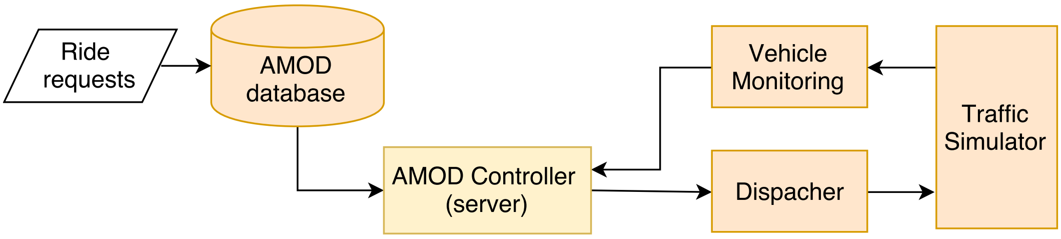


Figure 2.8: Centralized communication of AMOD system. Ride requests (bookings made through an app) are stored in the database. The AMOD server constantly pulls booking information from the database and assigns vehicles to customers. Assigned vehicles are sent to the dispatcher, which communicates the trip to the traffic simulator. To manage the AMOD system, the server keeps track of the fleet of vehicles through the vehicle monitoring module.

keeps track of the fleet of vehicles through the vehicle monitoring module. Additionally, AMOD Controller OR dispatches vehicles for the rebalancing trips and for the battery charging.

Class structure of AMOD Controller The manager is composed of different physical elements such as AMOD vehicles, stations, and customers. The class structure of the physical elements is presented in Figure 2.9. Physical elements belong to super class *Entity* and

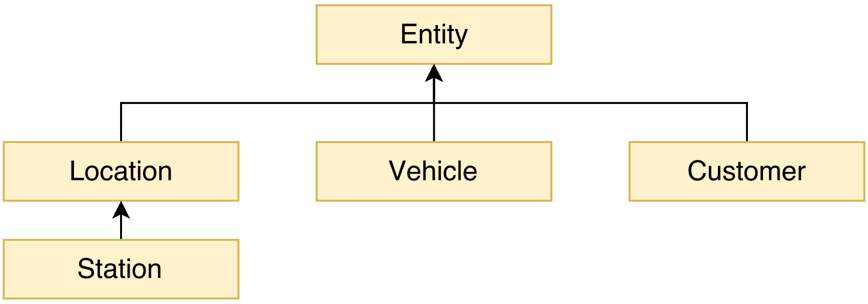


Figure 2.9: Class diagram for the physical elements of AMOD system: super class Entity, child classes Location, Vehicle, Customer and Station. A more detailed diagram is presented in Appendix C.

inherit functionality from it. The child classes of Entity are: Location (or Node), Vehicle and Customer. Station is a child class of Location. On top of the inherited functionality, each element adds new functionality as presented in Appendix in Figure C.1. During simulation the physical elements cannot be modified, e.g., the number of vehicles remains constant over the simulation time. Attributes of these elements, which describe their current occupation, change as the simulation progresses, e.g., a vehicle can be available at the beginning of the simulation, however when it picks up a customer, then its status changes.

Other classes and structures describe actions and movements of customers and vehicles.

They are presented in Figure 2.10.

Figure 2.10: Classes and structures describing actions within AMOD system.

Maps maintaining vehicles and customers flow Bookings, customers and vehicles are stored in *C++* map structures, which allows us for an easy access of the elements by the keys, e.g., map of bookings contains keys and values, where key is the booking ID and value is the Booking object. If we want to retrieve a particular booking we can find the entire object by the ID. There are 3 basic maps at which AMOD Controller operates:

(i) available vehicles

(ii) booking requests

(iii) pick-ups

(iv) drop-offs

2.4.1 Framework for Autonomous Vehicles in SimMobility

As mentioned, the behavioral models in SimMobility operate in different temporal resolutions. For the purposes of this dissertation, we focus primarily on the SimMobility Short-Term (SimMobility ST), which simulates individual decisions and transportation network at the sub-second level. AMOD Controller is an integrated, but detachable, component that imbues SimMobility ST with the capability to simulate an AMOD system. The communication framework of AMOD Controller and SimMobility ST is presented in Figure 2.11.

The models and algorithms in AMOD Controller are organized into three main components: initialization, fleet management and vehicle tracking modules. The principal component is fleet management, which assigns, dispatches and routes vehicles. This component is typically reconfigured depending on the model being evaluated. In general, as presented in Figure 2.11, initialization is triggered by AMOD Controller, which queries network information and trip data from SimMobility. Based on the response from SimMobility, AMOD Controller selects location of the facilities (stations for AMOD vehicles). The network structure and location of the facilities are passed to the fleet management module, which performs real-time operation of AMOD system. To perform assignment and rebalancing, the fleet management module queries SimMobility ST for the current travel time on the network. Results of the assignment and rebalancing are dispatched to SimMobility ST, which simulates vehicle movement and provides a continues update on vehicles' location and status. Vehicle tracking module receives information from SimMobility, logs events and feedback the fleet management module.

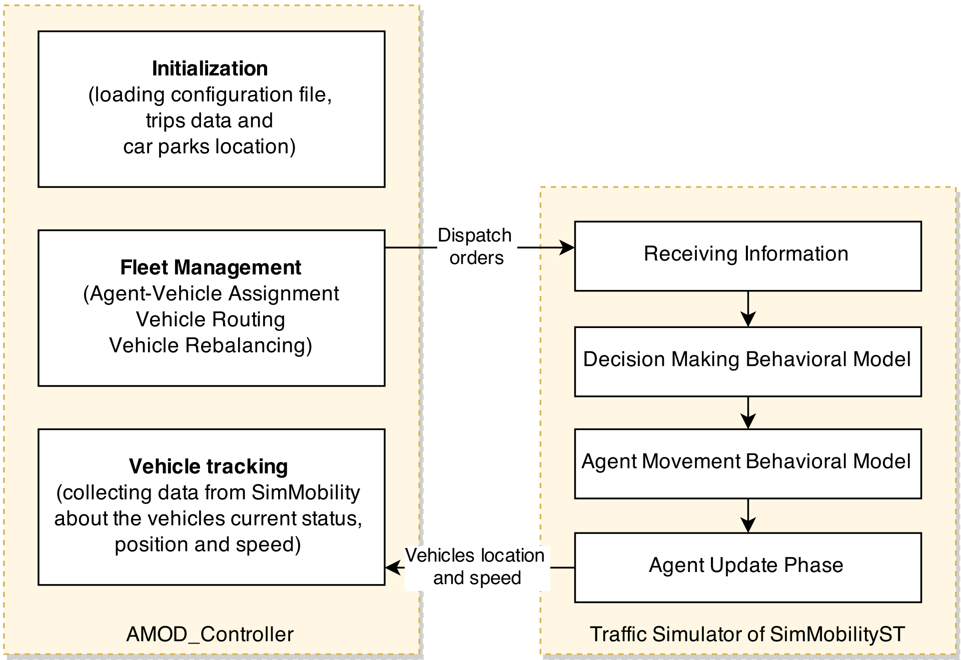


Figure 2.11: Framework enhancing SimMobility with dedicated AMOD Controller to manage and simulate autonomous vehicles.

More detailed information about input and output for each stage of AMOD Controller is presented in Appendix A.1.

2.4.2 Communication of AMOD Controller with SimMobility

To run AMOD Controller at the operational level, input from the planning and tactical level are required.

SimMobility MT and AMOD Controller Planning Stage At the planning level, we feed network and demand information from the simulator, i.e., SimMobility Mid-Term (*SimMobility MT*), into the controller as presented in Figure 2.13. The network information passed by SimMobility to Planning Stage consists on the list of nodes and edges. Nodes are described by x and y coordinates in WGD84 (zone 48N) coordinate system. Edges, e.g., road lanes and connectors are described by start and end node and the length. Demand information accepted by AMOD Controller Planning Stage is the daily activity schedule (DAS) generated by SimMobility MT. Each row of DAS consists of the following structure: `person id` is a unique id for the person; `tour_no` describes sequence number of the trip for the person; `tour_type` describes purpose of the trip, e.g., work-related trip; `stop_no`, `stop_type`, `stop_location`, which is understood as trip destination; `stop_zone`, `stop_mode`, which is understood as trip mode; `primary_stop`, `arrival_time`, `departure_time`, `prev_stop_location`, which

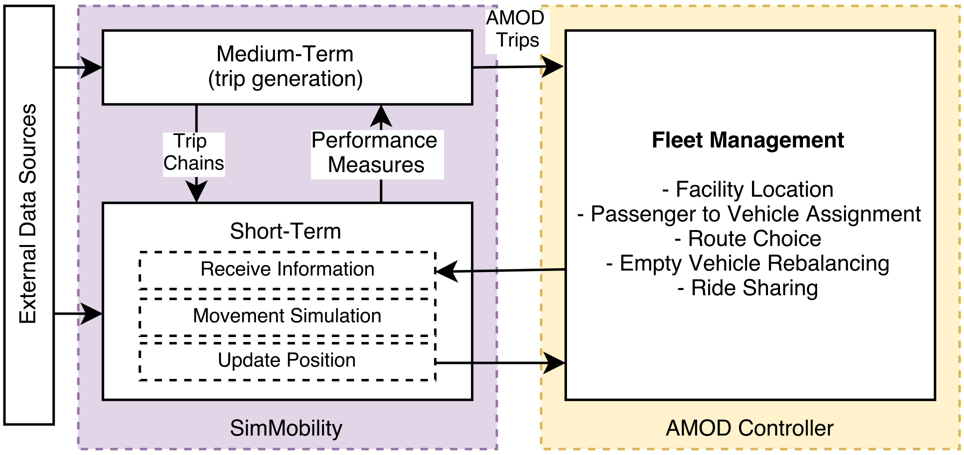


Figure 2.12: The AMOD Controller handles fleet management of autonomous vehicles and consists of five main components responsible for: (i) facility location, (ii) passenger to vehicle assignment, (iii) routing, and (iv) empty vehicle rebalancing. AMOD Controller dispatches orders to SimMobilityST, which performs a 0.1 second scale simulation of the vehicles and returns vehicular information (e.g., speed and location) back to the fleet management module.

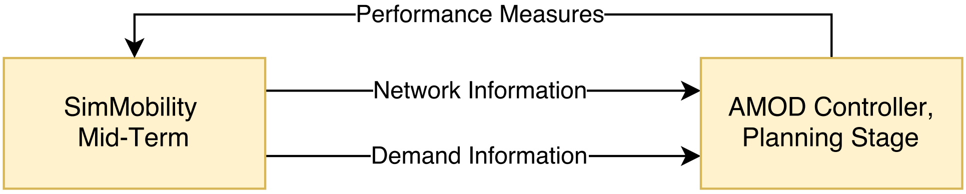


Figure 2.13: Integration of SimMobility and the Planning Stage of AMOD Controller. AMOD Controller gets data, i.e., network and demand information, from SimMobility Mid-Term Simulator.

is understood as trip origin; `prev_stop_zone`, `prev_stop_departure_time`, which is understood as trip tart time. Based on the input from SimMobility MT, Planning Stage estimates number and pattern of AMOD trips within the analyzed region.

AMOD Controller Planning Stage receives feedback (performance indicators) from AMOD Controller Tactical Stage so that the ASPS estimates can be adjusted based on the feedback.

AMOD Controller Planning Stage does not send any direct feedback to SimMobility MT. Indirectly, AMOD Controller Operational Stage communicates with SimMobility ST and SimMobility ST feeds back the performance measures to SimMobility MT (compare Section 2.1.

More details on structure and algorithms used by AMOD Controller Planning Stage are presented in Section 2.4.

SimMobility MT and AMOD Controller Tactical Stage AMOD Controller Tactical Stage is set to estimate

(i) Number of vehicles required in the system, and

(ii) General travel pattern to optimize static counts for vehicle rebalancing.

Tactical Stage uses information provided by the Planning Stage:

(i) Location of stations,

(ii) Processed information from SimMobility MT in Planning Stage, i.e., network and estimated demand as presented in Figure 2.14

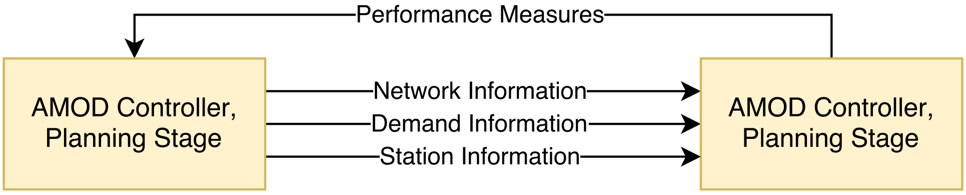


Figure 2.14: Integration of SimMobility and the Tactical Stage of AMOD Controller. AMOD Controller Tactical Stage gets data from the Planning Stage, i.e., network information, estimated demand, and location of stations. Note that network and demand data are indirectly received from SimMobility MT.

SimMobility ST and AMOD Controller Operational Stage Integration of SimMobility and the Operational Stage of AMOD Controller is presented in Figure 2.15. AMOD

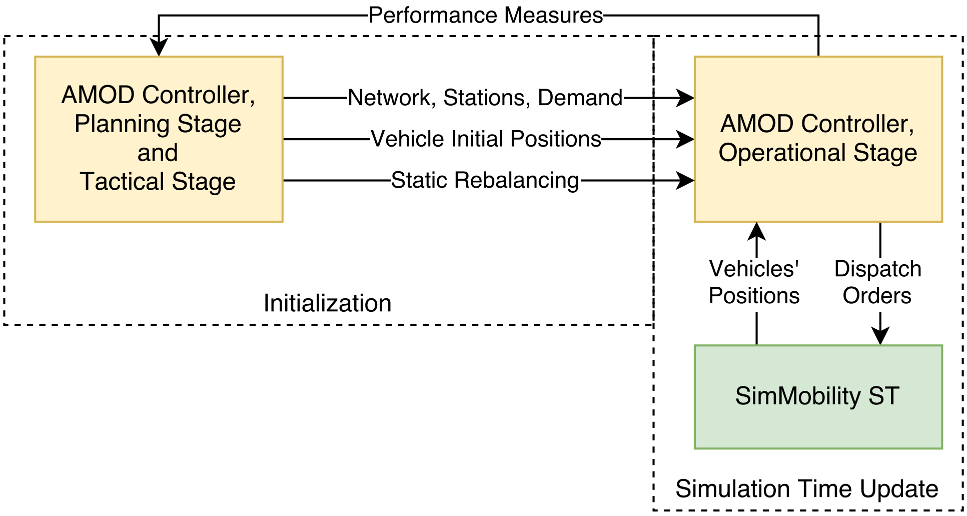

Performance Measures
AMOD Controller, Planning Stage and Tactical Stage
Network, Stations, Demand
Vehicle Initial Positions
Static Rebalancing
AMOD Controller, Operational Stage
Vehicles' Positions
Dispatch Orders
SimMobility ST
Initialization
Simulation Time Update

Figure 2.15: Integration of SimMobility and the Operational Stage of AMOD Controller. AMOD Controller Operational Stage gets initialization data from the Planning and Tactical Stage, i.e., network information, estimated demand, and location of stations. It performs assignments of vehicles to (passenger or empty) trips and dispatches trips to SimMobility. SimMobility updates the Controller with positions of the vehicles.

Controller Operational Stage gets initialization data from the Planning and Tactical Stage, i.e., network information, estimated demand, and location of stations. It performs assignments of vehicles to (passenger or empty) trips and dispatches trips to SimMobility. SimMobility updates the Controller with positions of the vehicles so that the Controller can update the dispatch orders. Every dispatch consists of the following message being sent to the simulator: `veh_id` is a unique id for the vehicle which is being dispatched; `destination_id` which is the destination node id (it may be a pick-up or drop=off or end of the rebalancing trip); `way_points` which is a list of nodes and edges (a path) which vehicle should take. Upon dispatch, AMOD Controller through the update module keeps track of all its vehicles and assigns them to new trips, e.g., to drop-off location after picking-up the customer.

2.5 Run-time Operation

The AMOD Controller handles fleet management of autonomous vehicles. Its fleet management component, which is a core component, consists of five main subcomponents responsible for: (i) facility location, (ii) passenger to vehicle assignment (including ride-sharing), (iii) routing, and (iv) empty vehicle rebalancing. During the run-time operation tasks of components (ii), (iii), and (iv) are performed.

2.5.1 Reservation Procedure

A general view of the reservation procedure The AMOD fleet is assumed to be homogeneous consisting of 2-seater autonomous cars. Each vehicle changes its role dynamically based on the requests from passengers, i.e., each vehicle can serve as a private and shared AMOD vehicle. There are two roles for vehicles:

(i) single-rider AMOD, *p-amod*, serves a single passenger providing a door-to-door service.

(ii) shared AMOD, *s-amod*, serves maximum of 2 bookings providing a door-to-door service for all customers (usually at an increased service time).

As presented in Figure 2.16 each passenger gets two offers from the AMOD pre-booking system (for private and shared ride) and he/she chooses to accept one of them or reject both. A ride request consists of the following information: (i) Pick-up and drop-off locations (ii) Time of the booking request Note that the customer can only request an immediate ride (no advanced booking is allowed). The trip offers are generated according to first-in first-out (FIFO) order (to reduce the computational burden) and are based on the current state of the system. Each offer has following attributes:

(i) Service type (*p-amod*, *s-amod*),

(ii) Estimated pick-up time,

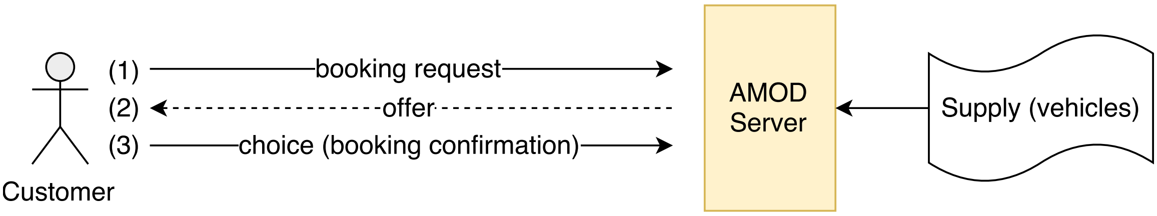


Figure 2.16: Reservation procedure. The AMOD customer request a ride by specifying origin and destination of the trips. The server sends back two offers: for single-rider and shared trip. The server provides estimated pick-up and drop-off time and estimated price for both options. To confirm (or cancel) the booking the customer selects one of the options or rejects both. Pseudo-code for the implementation of the reservation procedure is presented in Algorithm 2.1

(iii) Estimated drop-off time,

(iv) Fare.

Pick-up and drop-off time for single-riders are obtained based on the estimated arrival time of the nearest available vehicle. For the shared rides the upper bound of waiting and travel times are provided. For the shared ride we allow a 30% increase in waiting and travel time (compared to the private ride). We also provide a 30% reduction in the fare due to shared ride.

The customer selects the offer (*p-amod*, *s-amod* or reject) based on her/his personal attributes:

(i) Sensitivity to increase in commuting time, and

(ii) Sensitivity to price increase.

Currently the above attributes are synthetic, i.e., generated at random for each customer. To give an example, if a customer is sensitive to price, he/she will not accept any increase in price.
Pseudo-code for the implementation of the reservation procedure is presented in Algorithm 2.1. The AMOD customer request a ride by specifying origin and destination of the trips. AMOD server sends back two offers: for single-rider and shared trip. When the offer is send to the customer, the system suggests to share the ride. The customer evaluates the suggestion, and possibly, agrees to share. At this time, she/he can also decide to cancel the booking.

Algorithm 2.1: Reservation procedure

```
Data: availableVehicles, bookingQueue
Result: accepted single-riderTrips, accepted SharedTrips
if bookingsQueue is empty then
    break
end
for bookingQueue do
    if wait time exceeded then
        discardBooking
    end
end
if no vehicles available then
    break
end
for bookingQueue do
    generateTripOffers
    selectOffer
end
```

The reservation function is not triggered if there are no ride requests or no available vehicles. Customers stay in the queue only for a limited time (e.g., 5 minutes) and leave the system if they do not receive any offer within this time. Also note that the waiting time here is measured from the customer's arrival time until he/she gets an offer, i.e., it is not equivalent to waiting time to being picked-up by a vehicle. Based on the customer's decisions the AMOD Controller optimizes the assignment of vehicles, which is described in details in Chapter 4. AMOD server keeps track of vehicle positions and statuses, which is taken into account when optimizing the system.

2.5.2 Run-time Event Structure

In addition, this server is based on an event system so that all sub-modules just wait until the desired type of event has occurred. The event structures of the AMOD operation are presented in Figure 2.17 and Figure 2.18.

Booking management The simulations are discrete in time, with time step of 0.1-1.0 second. Bigger time step speeds up the simulation runtime, however at a cost of some accuracy in driving behavior. Besides the discrete time frame, the progression of simulation is also event-based. The event structure for managing the trip requests is presented in Figure 2.17. After the booking is received, the customer is prompted for the willingness of getting a shared ride (the pre-booking system, which is responsible for providing the options to customers, is presented in Section XX). If the customer accepts a shared ride, *SharedBookingServiced* and *SharedVehicleDispatched* events are triggered. For each pick-up and each drop-off an event is risen (*FirstPickup*, *SecondPickup*, *FirstDropoff*, *SecondDropoff*, respectively). After the second drop-off, the vehicle becomes available. A similar event structure is designed for a single-booking ride. All events are logged with the time stamp, IDs of the entities and coordinates (where applicable). If the process fails, e.g., due to the simulator's error, the error event is logged and the booking is destroyed.

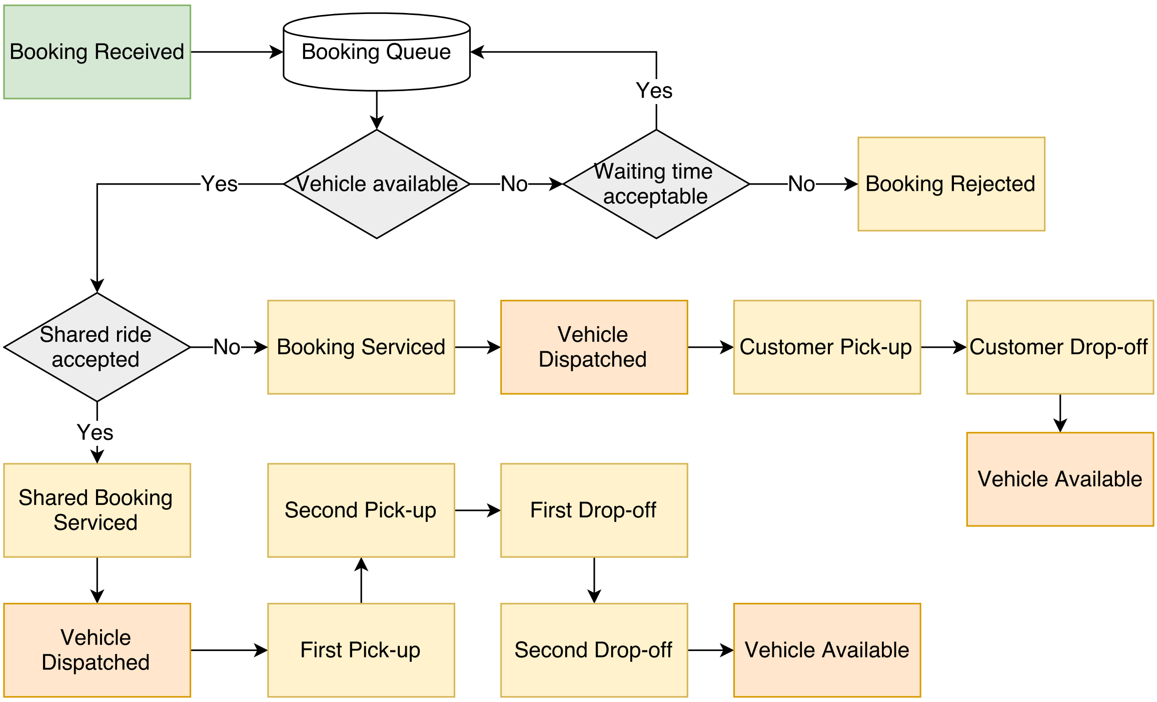


Figure 2.17: Event structure for managing the trip requests. All events are logged with the time stamp and IDs of the entities. If the process is distracted, e.g., due to the simulator's error, the error event is logged and the booking is destroyed.

Vehicle management The event structure for managing the vehicles is presented in Figure 2.18.

As a simple example that has been implemented, consider a first-in-first-out (FIFO) service that assigns to each customer the nearest available vehicle (in terms of shortest-path distance). The AMOD vehicles are routed with the least cost path between two different locations, where the cost is proportional to the traversed distance. After dropping off passengers, vehicles can either return to the originating station, the closest station or simply wait at the drop-off location for a service request. The implemented model of AMOD Controller

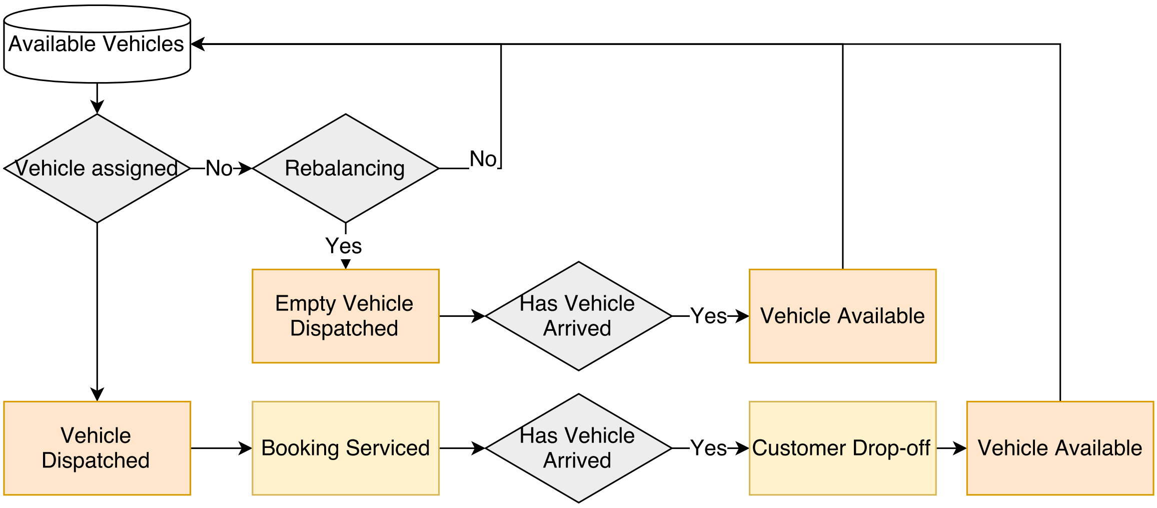


Figure 2.18: Event structure for managing the vehicles. All events are logged with the time stamp and IDs of the entities. If the process is distracted, e.g., due to the simulator's error, the error event is logged and the vehicle is *resurrected* with the time delay at the station of the destination.

is summarized in Figure 2.18. More sophisticated assignment and routing algorithms are embedded into the Controller as presented in Chapter 4.

Chapter 3

Facility Location

This chapter focuses on the most important strategic aspect of management of AMOD systems—selecting the number and location of new facilities. Intuitively, the spatial distribution of demand in a city is non-uniform and hence, strategically placed facilities can reduce customer waiting times and the fleet size. In traditional MOD systems (e.g., bike-sharing or car-sharing), accessibility to the stations (in terms of distance from your location to the station) is a critical factor, because people must walk to get a vehicle. In AMOD systems, customers do not have to walk to get a vehicle, but location of the vehicles can influence waiting time of passengers.

The facility location problem (FLP) presented in this chapter is closely related to the offline rebalancing method, which is described in Chapter 5. In the literature, these two decision levels are generally treated separately [Repoux et al., 2015]. However, facility location influences directly operational decisions and vice versa. In this dissertation we attempt to correlate and merge together these different aspects of an autonomous mobility on demand systems.

3.1 Background

Facility location problems is a strategic planning decision, which seeks to solve for a number and locations of a set of facilities to minimize the cost of satisfying a set of customers with respect to some set of constraints [Farahani and Hekmatfar, 2009, Liu, 2009, Boyaci et al., 2015]. The cost is typically associated with fixed (e.g., investment) and variable (e.g., conveying from facilities to customers) cost. The facilities are usually characterized by their number, type (e.g., size), and cost. Facility location is critical element in strategic planning for a wide range of problems and it influences numerous operational and logistical decisions [Farahani and Hekmatfar, 2009]. FLP models are used in a variety of applications. Some of them include locating warehouses within a supply chain to minimize the average time to market, locating noxious material to maximize their distances from the public, locating railroad stations to minimize the unpredictability of delivery schedules, locating automatic teller machines to serve bank customers better, etc. It is also closely related to similar problems in communication networks, logistics and distribution systems [Carlo et al., 2012].

3.1.1 Basic Classification of Facility Location Problem

The FLP problems can be classified into continuous and discrete space. The continuous FLP assumes that facilities can be located anywhere within the analyzed area and are referred to as site-generation models since the generation of appropriate sites is left to the model at hand. Solution to the continuous FLP does not consider any specific locations and therefore, is likely to select a facility location that is infeasible. However, the solution can still be useful to identify a region in which the facilities should be located. The discrete FLP assumes that the facilities are selected from pre-identified locations and are referred to as site-selection models since we have a priori knowledge of the site candidates [Farahani and Hekmatfar, 2009, Carlo et al., 2012]. Clearly, the discrete FLP has a much smaller solution space than the continuous FLP. The most widely studied model in discrete space is *uncapacitated FLP, UFLP*. In this problem we are given (i) a set of potential customers, (ii) a set of potential facilities, (iii) cost of opening each single facility, and (iv) distance between all elements satisfying triangle inequality. The goal is to select (open) undetermined number of facilities and assign each customer to an open facility such that the sum of the total opening cost and total service cost is minimized. This NP-complete problem has been studied in the literature extensively and there exist some successful approximation algorithms to solve the problem [Vogel and Mattfeld, 2004]. One of the most efficient solution techniques for this problem, presented and reviewed in [Verter, 2011], is as a dual-based algorithm [Erlenkotter, 1978]. In many cases, it is more realistic to incorporate the capacity limitations on the facilities. This problem is known as *capacitated FLP, CFLP* and it is a generalization of the uncapacitated problem. In contrast to the UFLP, each facility in CFLP can provide limited quantity of product. Although mathematical models of those problems not very differ much, but solving methods for CFLP are more difficult. Based on [Verter, 2011], the most efficient algorithms for CFLP are the cross-decomposition algorithm [Van Roy, 1986] and the Lagrangean-based approach [Beasley, 1988].

Models of FLPs can differ in their formulations. An overwhelming majority of the proposed algorithms aim at minimizing the total fixed and variable costs relevant to the location problem under consideration. In general, we distinguish six components that describe facility location problems:

(i) customers, who are assumed to be known a priori,

(ii) set of potential facilities that will be located (in case of discrete space model),

(iii) a space in which customers and facilities are located,

(iv) the objective function,

(v) the number and size of facilities to be located, and

(vi) a metric that indicates cost (distances or time) between customers and facilities.

Covering problem One of the most popular models among facility location models is covering problem [Farahani et al., 2012]. In most of the covering problems, customers are served by facilities depending on the distance between the customer and facilities, i.e., the customer can be served by the nearest available facility. The concept of covering problems can be classified in two categories: (i) Set Covering Problem (SCP) where coverage is required, and (ii) Maximal Covering Location Problem (MCLP) where coverage is optimized. Both models are analyzed in this dissertation (see Section 3.2 for more details).

3.1.2 Related Work

The research focusing on the strategic problems regarding the number and locations of facilities for mobility-on-demand systems is very broad [Shu et al., 2010, Lin and Yang, 2011, Martinez-Barbera and Herrero-Perez, 2012]. In the related problems [Boyaci et al., 2015, Carlo et al., 2012, Chen et al., 2013, Correia and Antunes, 2012, Farahani and Hekmatfar, 2009, Garcia-Palomares et al., 2012, Larsen et al., 2013, Lin and Yang, 2011, Kumar and Bierlaire, 2012], facility locations are optimized based on the expected demand for the service. [Lin and Yang, 2011] address the strategic planning of public bicycle sharing systems. The authors propose an integer nonlinear program to model an integrated decision problem, which determines (i) where to locate the bike stations, (ii) where to build the bicycle lanes, and (iii) what paths should be used for users from each origin to each destination. Due to the complexity of the model, it is only applicable to small scenarios. [Martinez-Barbera and Herrero-Perez, 2012] presents an heuristic mixed integer linear program, that simultaneously optimizes the location of stations, the fleet size and relocation activities required in a regular operation day of a bike-sharing system. The results show the spatial distribution and capacities of docking stations and the cost structure for introducing different scenarios of the bike-sharing scheme. [Garcia-Palomares et al., 2012] proposes integration of location-allocation models with one of the most commonly used commercial GIS. The authors applied the proposed methodology for the optimal location and capacity of bike-sharing stations in the city center of Madrid. Facility location for car-sharing services are usually formulated in a similar way. In general, one of the main problems faced by the car-sharing businesses is finding the locations to park the vehicles. These locations should be chosen by attractiveness of various parameters such as the socio-demographic-economic profile of the population that resides or works at the location and accessibility of other forms of public transport in that location [Boyaci et al., 2015]. However the presence of one or more stations in the vicinity and the type of vehicles placed in a station are also critical for decision-making. The overarching problem addressed in [Correia and Antunes, 2012] is how to select sites for locating depots in order to maximize the profits of a one-way car-sharing organizations. Their approach is based on mixed-integer programming model which maximizes the total profit of a car-sharing operator. The approach is illustrated by the case study of Lisbon, Portugal. [Kumar and Bierlaire, 2012] use two-step procedure to optimize the new station locations for an existing car-sharing service in the city of Nice, Switzerland. At the first step, the authors analyze the performance of an electric car-sharing service across different stations (the performance of the station is measured by the average number of rides per day), and estimate the driving factors (attractiveness measures) for people to use the service. Secondly, the authors use the attractiveness measures to find the best locations for new stations so that the overall performance of the system is maximized. Literature on electric vehicle charging stations also highlights the importance of facility (in this case charging stations) location problem. It is commonly agreed that availability of charging stations will impact EV adoption rates [Chen et al., 2013]. The charging stations should be (i) pervasive enough such that an EV anywhere can easily access a charging station within its driving range, (ii) widely spread so that EVs can cruise around the whole city upon being recharged [Lam et al., 2014]. There is an increasing number of research in the domain of charging stations for EVs. [Wang et al., 2010] present a numerical method for the layout of charging stations using a multi-objective planning model. The authors design their algorithm based on demand priority and the usage of the existing gas station. They apply their algorithm to the city of Chengdu, China. [Wang et al., 2013] formulates a quantitative method for location of charging facilities. The model is based on the conversion of oil sales to electricity sales, which correlates replacing traditional gasoline vehicles with electric vehicles. [Liu et al., 2013] identifies the optimal sites of EV charging stations by a two-step screening method with environmental factors and service radius of EV charging stations considered. Then, the authors apply a modified primal-dual interior point algorithm for the optimal sizing of EV charging stations with the minimization of total cost associated with EV charging stations. The algorithm is tested on the IEEE 123-node test feeder and the results show that the algorithm improves the overall electric grid performance. [Lam et al., 2014] prove that the problem of location of charging stations is nondeterministic polynomial-time hard. The authors propose four solution methods to tackle the problem: (i) iterative mixed integer linear program, (ii) greedy approach, (iii) effective mixed integer linear program, and (iv) chemical reaction optimization. The work is concluded that each method is suitable for different situation depending on the modeler's requirements. [Wang et al., 2016] formulate the problem of the electric vehicle charging station placement for public buses. The authors use relaxation of the integer linear program.

3.2 Model Formulation

In this dissertation, we address the problem of the design of an autonomous mobility on demand system. More specifically, in this section we present model formulation for the facility location problem. We develop exact methods for deciding on the location of distribution and charging stations in a [station-based] autonomous mobility on demand system that provides one-way trips to its users. We conduct computational experiments on the data created based on estimated AMOD trips in the extended CBD in Singapore.

3.2.1 Problem Definition

In this section we describe definitions, terminology and assumptions used for the model formulation. We formulate the facility location problem as a discrete optimization problem. For this reason we provide some details on the discretization which is performed based on the SimMobility's road network.

Representation of the road network Road network used in this dissertation is represented as a graph. An example of a simple road network represented as a graph is shown in Figure 3.1. Graph G is a set of vertices V (nodes), which are connected by edges E (arcs). A vertex in a graph is associated with at least one arc, while an arc is associated with exactly two vertices. There are a few basic types of graphs depending upon the number of vertices, number of edges, interconnectivity, and their overall structural properties. (i) A graph may be undirected, in which edges have no orientation, or directed, in which edges are directed from one vertex to another (compare examples in Figure 4.2). A directed graph

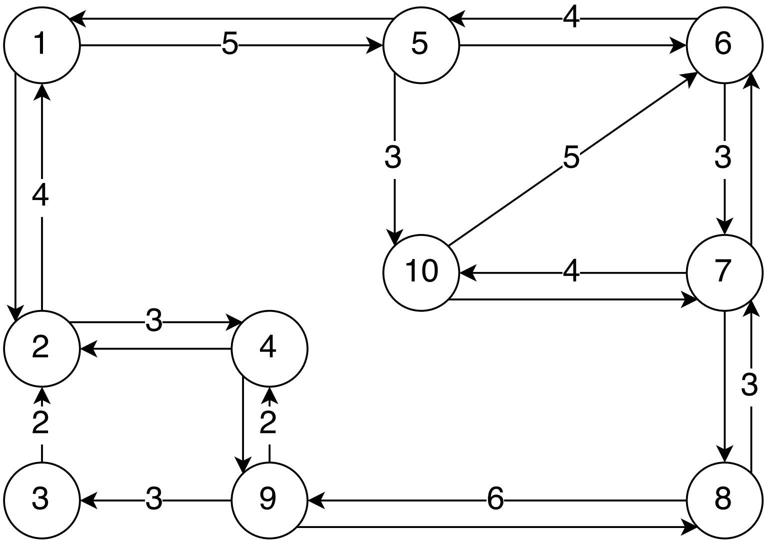


Figure 3.1: A general weighted graph representing a road network. Circles are nodes (intersections) labeled with unique node IDs, while arrows are the edges. Each arrow is characterized by weight (cost) associated with traveling from the origin to the destination node, i.e., cost of traveling from node 1 to node 5 (and from node 5 to node 1) is equal 5. The road network consists of two-way and one-way roads, which can be easily read from the graph. Note, that the graph satisfies the triangle inequality.

$G = (V, E)$ consists of a set V of vertices and set E of directed edges, where a directed edge is an ordered pair *(i,j)* of distinct nodes. In a directed graph, for any arc *(i,j)*, we say that i is the start node and j is the end node. In a directed graph, a vertex is said to be a sink node if there is no path to or from the vertex. (ii) A graph may be weighted, in which there is a weight assigned to each edge. Without the weights, the graph is typically assumed to be unweighted. (iii) A complete graph contains all possible edges. Otherwise, the graph is incomplete. (iv) A *walk* from vertex u to vertex v in undirected graphs is defined as a sequence of vertices, while a *walk* in directed graphs is defined as a sequence of vertices, together with associate sequence of edges. A walk is said to be a *path* if all its nodes are distinct [Bertsimas and Tsitsiklis, 1997]. For a weighted graph, length of the path is defined as the sum of the costs of all edges in the path. *The shortest path* is a path from a given origin node to a given destination node whose length is the smallest. For the graph shown in Figure 3.1 the sequence *5, (5,10), 10, (10,7), 7, (7,6), 6, (6,5), 5, (5,1), 1* is a walk, but not a path, because node 5 is visited twice. The sequence *10, (10,7), 7, (7,6), 6, (6,5), 5, (5,1), 1* is a path, but it is not the shortest path. (v) A graph may consists of a path starting and ending at the same vertex, which is referred as cycle (closed walks). A graph consisting of no cycles is known as acyclic graph. (vi) Edge-weighted graph satisfies the triangle inequality if the graph has no shortcuts. This holds for any graph representing points in a metric space. (vii) Trivial graph is a graph which contains only one vertex and no edges.

Road Network used in this dissertation Road network is represented as a weighted directed graph. The graph is connected and consists of no cycles (no edges *(v,v)* are allowed for all $v \in V$). The nodes represent points in a metric space. This implies that they satisfy the triangle inequality [Atallah and Blanton, 2009]. The triangle inequality is property that $weight(u, v) \leq weight(u, w) + weight(w, v)$ holds for all vertices *u, v, w.* Informally, it means that the graph has no short cuts. An example of the road network retrieved from SimMobility is shown in Figure 3.2.

The road network used by AMOD Controller is taken from SimMobility. SimMobility's network is a comprehensive representation which provides information on all lanes including pedestrian walkways and bicycle's paths. It also gives detailed information on the intersections, i.e., whether there are traffic signals and what are the turning groups.

The Extended Central Business District (ECBD) in Singapore, presented in Figure 3.2, is highlighted in pink. The area of the analyzed zone is 57 km^2, which is 8 % of the entire Singapore area. This zone carries over 30% of the entire passengers' traffic as most of the daily trips are to and from CBD. The blue circles represent nodes and are located at the drivers' decision points, e.g., intersections, entrances or exits from the expressway. Nodes are connected by links (edges), which represent road lanes. The analyzed road network is a part of the entire network. The ECBD network is cut in such a way that there is no sink and source nodes as they are substituted by u-turns. For any source or sink node which

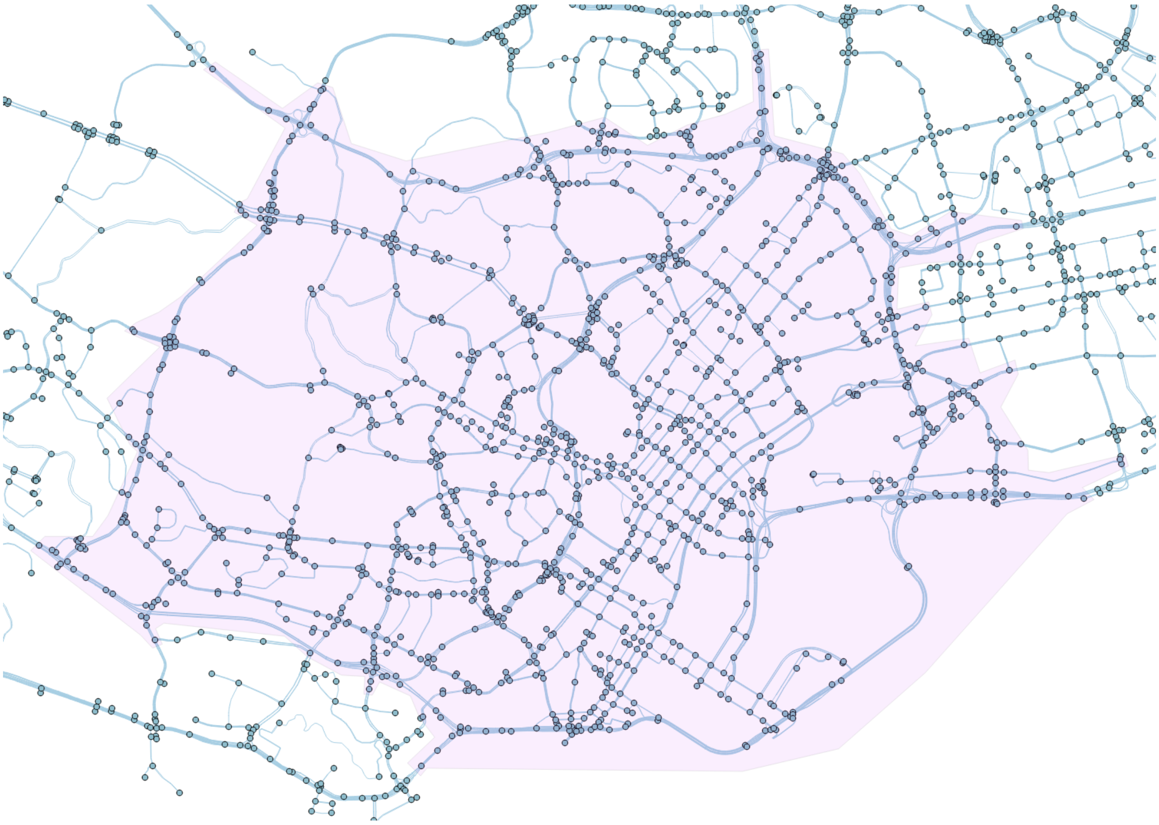


Figure 3.2: Road network of the Extended Central Business District (ECBD) in Singapore (highlighted in pink). The network has been generated based on SimMobility's network information. Note, that circles represent nodes and are located at the intersections and other decision points. Blue edges are road lanes. The ECBD network is a subset of the entire Singapore network.

arose from the cut, a link from a neighbor node has been created so that the network is fully traversable for vehicles. We believe this is a fairly realistic assumption given the fact that in the real world vehicles can almost always make a u-turn.

Assumptions in the models We formulate facility location problem as a discrete optimization problem. There are n potential facility locations and m passengers, who will be served from these locations. We represent our set of potential facilities as a set of nodes in a connected graph. Nodes are placed at the intersections and decision points of the road network as presented in Figure 3.2. All edges in the graph are directed.

The problem is to find *the best* locations to place distribution centers for autonomous mobility on demand vehicles. We were working on 3 case-study scenarios:

(i) A 14 km^2 zone in the Central Business District in Singapore, consisting on 2945 nodes.

(ii) A 57 km^2 zone in the Extended Central Business District in Singapore, consisting on

1,255 nodes.

(iii) The entire Singapore network of 714 km^2, consisting on 5,424 nodes.

In each formulation presented below, the set of potential facilities consists on all nodes within the analyzed area. The demand generation is case specific. The detailed description of the demand generation is presented is Chapter 6. A general rule is that the origin and destination of all trips is always at nodes, i.e., trips never start in the middle of the link or between the links, they are always approximated to the nearest node. For the facility location problem we use 3 basic demand cases:

(i) For the 14 km^2 zone we analyze the demand for the 2-hour period during evening peak (5:00 PM to 7:00 PM) consisting on 28,525 trips within the region.

(ii) For the 57 km^2 zone we analyze we analyze the demand for the 12-hour period from 3:00 AM to 3:00 PM consisting on 808,356 trips within the region.

(iii) For the entire Singapore we analyze we analyze we analyze the demand for the 12-hour period from 3:00 AM to 3:00 PM consisting on 2,396,543 trips within the region.

In this chapter, we show application of the the models to the 57 km^2 zone in Singapore with the demand consisting on 808,356 trips. The remaining scenarios, as well as a complete picture of scenario presented here, are presented in Chapter 6.

The following sections introduce models analyzed in this dissertation and the comparison of their performance. The performance of each model is estimated based on the average waiting time of passengers.

3.2.2 Distribution of Stations Based on Location Factor Rating

In this approach we select facilities based on the factor rating. We identify one factor which is measurable and important to us: the proximity to the highest number of customers. We do not consider other factors such as the geographic location and proximity to the offices or shopping centers as we do not have this information. The model can be easily extended to account for additional factors. We assume no capacity constraint on the facilities, i.e., each facility could hold as many cars as required.

Location factor rating is defined as a function of the number of trips originating at each node. In this formulation, we rank all nodes within the network based on the number of trip origins, i.e., the highest rank is given to the nodes with the highest number of trip origins. If there are two nodes with the same number of estimated trip origins, we give them the same rating. After assigning the ratings, we select the first n nodes to be the stations for AMOD vehicles. In the first cut-implementation, the number of facilities, n, is selected based on the trial and error method through simulation of different number of facilities and based on their performance score. This model is further extended to capture relation with the set covering model (presented in Section 3.2.4). The relation between two models is established in the following way. We assume that the number of station in location factor rating model is:

(i) Equal to the number of facilities generated by the set covering model,

(ii) Equal to 130% of the number of facilities generated by the set covering model,

(iii) Equal to 70% of the number of facilities generated by the set covering model,

Station distribution based on the location factor rating is the first-cut implementation of the facility location problem. By evaluating high-demand locations we are looking at the difference in AMOD performance between simple and optimization-based approaches of the facility location problem. We present our optimization-based approaches in the following subsections.

There are few limitations of the model. The formulation may be bias towards some specific areas within the city, e.g., model may generate too many stations in the CBD areas where many commuters start or end their trips leaving some neighborhoods with no stations in the proximity. It also does not account for the capacity of the stations and number of charging infrastructure required by the system.

3.2.3 One Station for the Entire System

It is a problem of locating one centralized station (depot) for the entire AMOD fleet. There was no capacity constraint on the depot, i.e., the facility could hold as many cars as required. We use the *Center of Gravity Technique*, or weight center, to locate the depot (based on Equation 3.1). This technique is a quantitative method for locating one facility at the center of movement in a geographic area based on weight and distance.

$$x = \frac{\sum_{i=1}^{n} x_i W_i}{\sum_{i=1}^{n} W_i}, \quad y = \frac{\sum_{i=1}^{n} y_i W_i}{\sum_{i=1}^{n} W_i} \tag{3.1}$$

At each demand point $i \in N$ we have W_i number of customers. We locate a depot at *(x,y)* location, which is at the gravity center.
The problem with this approach is that the resulted coordinates are based on straight-line distances, which may not be accurate for the real road distances. However, similarly to Section 3.2.2 we aim here to cross-check if the facility location influences significantly the design of the AMOD system.

3.2.4 Stations Based on the Set Covering Problem

The set covering problem is one of Karp's 21 NP-complete problems [Karp, 1972]. In covering problems, customers need to be with a specific distance through facilities which are servicing [Farahani and Hekmatfar, 2009]. In order to formulate the FLP as a classical maximum coverage problem, we use the integer programming technique (as presented in [Bertsimas and Tsitsiklis, 1997]). We define a binary decision variable x_j for each location j, which is equal to 1 if facility j is selected, and 0 otherwise. Next, we define a set N containing all stations $n \in N$ and a maximum acceptable service distance S. In our case we assume $S = 4km$ (if vehicle moves 40 km/h, customer waits 0.1 hr). A set of the potential facilities N_i within S, so that $N_i = \{j | d_{ij} \leq S\}$, The set covering problem is formulated in Equation 3.2. Objective function of the model is to minimize number of required facilities covering all demand points.

$$\begin{aligned} \text{minimize} \quad & \sum_{j=1}^{n} x_j \\ \text{subject to} \quad & \sum_{j \in N_i} x_j \geq 1, \ i = 1, ..., m \\ & x_j \in \{0,1\}, \quad j = 1, ..., n \end{aligned} \tag{3.2}$$

The objective function minimizes the total number of facilities. First constraint shows the service requirement for the demand node i. Second constraint is the integrality constraint.

Equivalent formulation of the problem is presented in Equation 3.3.

$$\begin{aligned} \text{minimize} \quad & \sum_{j=1}^{n} c_j x_j \\ \text{subject to} \quad & \sum_{i=1}^{n} a_{ij} x_j \geq 1, \quad \forall i, \quad i = 1, ..., m \\ & x_j \in \{0,1\}, \quad j = 1, ..., n \end{aligned} \tag{3.3}$$

In this formulation the objective function minimizes the cost of locating facilities (c_j is the cost of placing facility x_j). We assume that $c_j = 1 \forall j$. We introduce a binary parameter $a_i j$ which is equal 1 if distance from candidate facility j to the customer i is not greater than S. In order to cover all demand nodes, first constraint enforces that for each demand node, at least one facility must be located within distance S. Second constraint is the integrality constraint.

Greedy algorithm to solve the set covering problem The set covering problem is found to be NP-hard [Karp, 1972]. One of the methods to solve this problem is the greedy polynomial time approximation algorithm. First we define service radius S and discretize the demand based on the nodes in the network. Then, we evaluate the coverage within distance

S for all potential facilities and run Algorithm 3.1.

Algorithm 3.1: Greedy approximation for the covering set problem

```
input  : D, N
output : sN
repeat
    pick the set that covers the maximum number of uncovered elements
    mark elements in the chosen set as covered
until all elements are covered
```

Out input is a set D consisting of n customers (demand) and a collection N consisting of j subsets of U (representing coverage of all potential facilities). Our goal is to select as few subsets as possible from N such that their union covers U. The output of the algorithm is sN, which is a subset of N.

3.2.5 Stations Based on the Maximal Coverage Problem

In this formulation we assume that the resources are not sufficient to cover all of potential customers with desired level of coverage, i.e., we can only place a limited number of facilities. This model (introduced by [Church and Velle, 1974]) maximizes the amount of demand covered within the distance S by locating a given fixed number of new facilities. The formulation of the Maximal Covering Problem is an intuitive extension of the Set Covering Formulation with an additional constraint on the number of selected facilities. In other words, we aim to cover as many customers as possible by a fixed number of facilities. To formulate the problem, we introduce index i as the index of demand nodes and j as the index for potential facilities. In our case the number of demand nodes and potential facilities are equal. We define h_i as the number of demands at node i and S as service time (or coverage distance). The maximum number of stations to be located is P. We introduce a binary variable, x_j, indicating if a facility is positioned at j or not, and a binary decision variable z_j indicating if node i is covered (1 if yes, otherwise 0). The problem formulation is presented in Equation 3.4.

$$
\begin{aligned}
\text{maximize} \quad & \sum_i h_i z_i \\
\text{subject to} \quad & z_i \leq \sum_j a_{ij} x_j, \quad \forall i \\
& x_j \leq P \\
& z_i \in \{0, 1\}, \quad \forall i \\
& x_j \in \{0, 1\}, \quad \forall j
\end{aligned}
\tag{3.4}
$$

The objective maximizes the number of customers covered by the stations. First constraint states that demand node i is covered if there exists at least one facility at one of the potential sites which is able to cover node i. Second constraint limits the number of located facilities to P. The last two constraints are integrality constraints.

Greedy algorithm to solve the set covering problem The Maximal Coverage Problem, similarly to the Set Covering Problem, is found to be NP-hard [Karp, 1972]. One of the methods to solve this problem is the greedy polynomial time approximation algorithm introduced in Algorithm 3.1.

3.3 Results

In this section we present analysis of the facility location methods. Methods are evaluated in terms of the customers' waiting time (from booking request until the pick-up). Note, that the simulation assumptions and complete analysis of the AMOD system is presented in Chapter 6.

Data In order to estimate the potential demand of an AMOD system, we use the data for the taxi rides in Singapore. We filter the data for the extended Central Business District area including incoming and outgoing trips. The data are obtained from the Comfort Taxi provider in Singapore through the Singapore-MIT Alliance for Research and Technology. The data consist of 2 weeks traces of the taxis including their statuses, e.g., available, passenger on board. We run the analysis on the network of 5424 nodes. To evaluate performance of the facility location models we run 10 simulations for each model and compare the average waiting time for each model. As our customer's locations the trip origins are selected (destinations are not taken into account when optimizing the facility location). We limit the sampling period to the morning peak period (6am until 9am) and evening peak period (5pm until 8pm), which helps us to capture all most important points of interest (home and work locations). The model generated a total demand in the study area of 418,383. The location of the potential depots was determined using the models described in Section 3.2.

3.3.1 Model Evaluation

In this analysis we test performance of the facility location as a function of both, the coverage radius and facility location model. The coverage radius is a parameter for all facility location models presented in this thesis. Therefore, we evaluate facility location models with different values of the coverage parameter. The coverage radius we test ranges from 500m to 2500m (Table 3.1). For each coverage radius we evaluate three alternatives:

Table 3.1: Setup for the facility location analysis. CR is the coverage radius for each station; MNF is the minimum number of facilities based on the set covering problem, 70%C is the solution to the maximal coverage problem with the number of facilities open equal to 70% of the number of facilities facilities open in MNF solution; MFO is the solution to the location rating formulation based on the most frequent trip origins

CR [m]	MNF	70%C	MFO
500	89	62	89
750	50	35	50
1,000	36	25	36
1,250	29	20	29
1,500	18	13	18
1,750	13	9	13
2,000	11	8	11
2,250	10	6	10
2,500	7	5	7

(i) Solution to the set covering problem, which gives us the minimum number of facilities required for covering the entire demand within the given coverage radius,

(ii) Solution to maximal coverage problem which gives us maximal coverage of the demand by 70% of the number of facilities required to cover the entire demand (70% of the number of facilities in the set covering problem),

(iii) Solution to the distribution based on the location factor rating. In this model we choose the most popular origins of the trips. The number of facilities we select in this formulation is equal to the number of facilities selected by the set covering problem. Note that no specific radius is directly evaluated in this model. We only choose the number of selected facilities to be equal to the optimized distribution of the same number of facilities.

As presented in Table 3.1 for the smallest radius of 500m we need as many as 89 facilities when solving for the minimum number of facilities to cover all demand points. We also test the case where we are able to open only 70% of the facilities required by the set covering model. This case is solved with the maximal coverage formulation restricting the number of station to 70% of the set covering solution. The visualization of the output for the service radius of 2,000 m is presented in Figure 3.4. Waiting time for each formulation is shown in Figure 3.3. The results show a general trend that the waiting time increases as we increase the service radius. This may be explained by the fact that if we have fewer number of facilities, they are more spread over the region and therefore, the vehicles need more time to drive from the facilities to most of the locations. This trend is valid across all models. Further, we conclude that when we place facilities based on the set covering problem (blue line on the graph), the customers wait 1.5 minute shorter to be picked-up than for the case where facilities are placed at the high demand nodes. This gives us an insight that even if we place stations at high demand nodes, there is still a significant number of customers who are not within proximity of the station. Furthermore, our results suggest that placing fewer facilities, which locations are optimized, performs better than a higher number of facilities placed purely based on our location rating (green line vs read line on the graph), e.g., it is more desired to place 62 optimized facilities rather than 89 facilities which are purely based on the location rating. To support the claim, Figure 3.4 shows a graphical representation of the output from each model for the case of the coverage radius equal to 2,000m. In this figure we can see clearly that for the stations based on the location rating we tend to cluster facilities at one region leaving a significant part of the region not covered. Given the travel time information, we know that the mean travel time on the network is 13 minutes. Based on our analysis, we conclude that up to 15% of the additional commuting time can be saved by simply optimally placing the same number of station stations rather than placing them based on the location rating. The average rate of the waiting time increase is 0.5 minute, 1.1

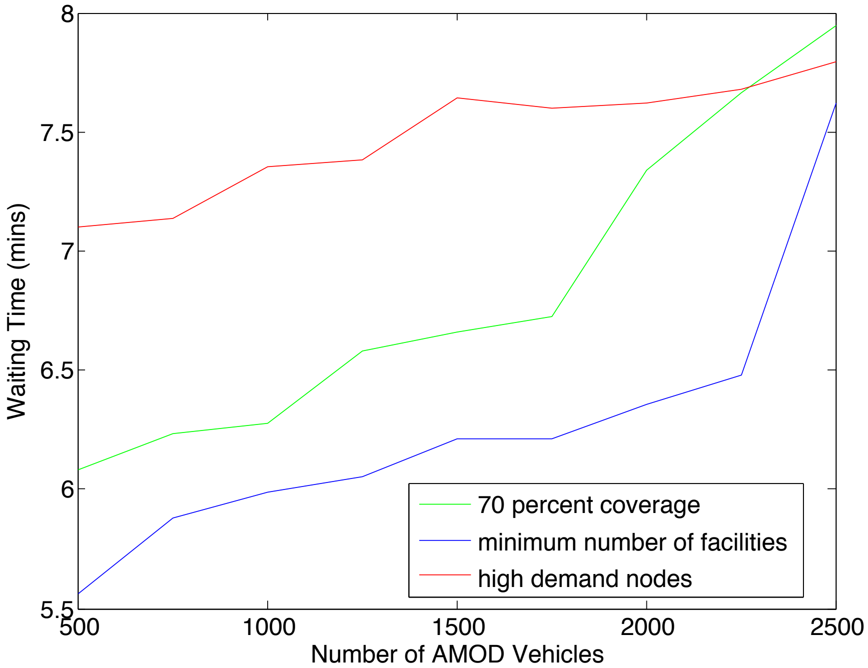


Figure 3.3: Customer waiting time as a function of the service radius of facilities. The results show a general trend across all models that the waiting time increases as we increase the service radius. Placing fewer facilities, which locations are optimized, performs better than a higher number of facilities placed purely based on our location rating (green line vs read line on the graph). Placing facilities based on the set covering problem gives us the shortest waiting times (blue line).

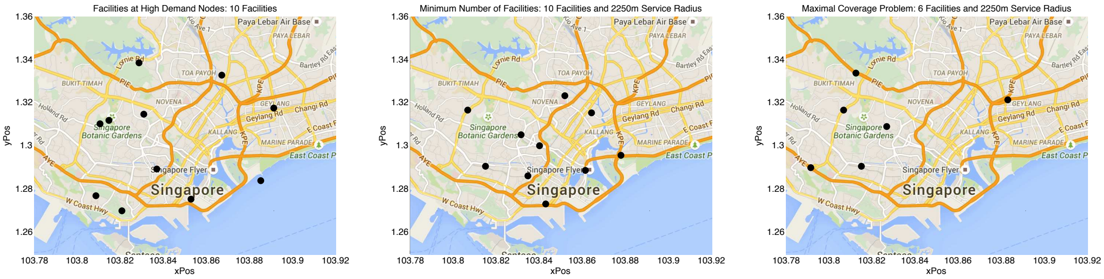

Facilities at High Demand Nodes: 10 Facilities
Minimum Number of Facilities: 10 Facilities and 2250m Service Radius
Maximal Coverage Problem: 6 Facilities and 2250m Service Radius
yPos
xPos
1.36
1.34
1.32
1.3
1.28
1.26
103.78
103.8
103.82
103.84
103.86
103.88
103.9
103.92
Singapore
Paya Lebar Air Base
Singapore Botanic Gardens
Singapore Flyer
TOA PAYOH
NOVENA
GEYLANG
KALLANG
MARINE PARADE
BUKIT TIMAH
W Coast Hwy
East Coast P

Figure 3.4: Facility location for different models: high demand nodes (left), set covering formulation (middle), and 70% coverage (right).

minute and 0.4 minute for each 1 km of service radius for MNF, 70%C and MFO models, respectively. Additionally, each additional station in the set covering problem reduces the average waiting time by 1.2 second. In the maximal coverage problem the waiting time is reduced by 3.5 seconds per one station added and in the location rating model by 1.0 second per one station added. Therefore, the most significant improvement in the waiting time is achieved with increase in the number of stations in the maximal coverage model. Finally, it has to be noted that the analysis is dependent on the demand distribution. In our case the demand was spread over the entire network (not clustered at only few locations).

Our results highlight the importance of the facility location problem for the performance of an autonomous mobility on demand system. We conclude that it is the most desired for an AMOD service to place facilities based on the set covering problem.

3.4 Summary

This chapter addresses strategic problem of facility location for an autonomous mobility on demand system with regular (single-class) electric vehicles.

3.4.1 Chapter Summary

We formulated five different models to solve the problem:

(i) Stations at high demand nodes,

(ii) Stations at random locations,

(iii) One depot for the entire system,

(iv) Stations based on the maximum set covering problem,

(v) Stations based on maximal coverage.

Our results suggest that the service radius of stations plays an important role in the performance of an AMOD system. We observe that the shorter the service radius the shorter the waiting time as vehicles need less time to reach customer locations. The analysis provides us with the insight that the maximum set covering model proved to be efficient and sensitive to different operational configuration for a medium to large case study (gives the shortest waiting times for the customers with the minimum fleet size).

3.4.2 Limitations

We are aware that our results may change significantly if we change the fleet size, assignment and rebalancing policies. Further extensions of the proposed solutions should consider:

(i) Geographic information, which is required for cross-checking of the feasibility of potential locations;

(ii) Capacity of each station; at the current implementation each station can *store* as many vehicles as needed, however we are aware that this assumption is not realistic.

(iii) Requirements for charging infrastructure at each station; in real situation electric vehicles are running on batteries with limited range and therefore they have to be recharged. In our work, we neglected the battery capacity and range of the vehicles. As opposed to combustion engine vehicles, charging batteries of EVs may take significant time during which vehicles are offline. Additionally, external factors such as weather conditions may affect range of electric vehicles, e.g., EVs have lower fuel efficiency at colder temperatures, decreasing how far the vehicle can travel without refueling.

(iv) Cost of placing new car parks may be affected by its location and number of parking lots available; in the current study we assume that the stations do not differentiate by price, i.e., the price was not a decision factor for placing facilities.

Chapter 4

Fleet Assignment

Operation of an AMOD system involves picking up and dropping off passengers, which is intrinsically linked to many operational issues. Given a distribution (location) and capacity of available vehicles, the fleet assignment problem faced by the operator is to determine which vehicle should pick up which customer. This decision has a major impact on performance of the system.

In this chapter, we develop optimization approaches specifically tailored to the dynamics of an e-hailing environment where customers continuously enter and leave the system and vehicles become available and busy based on the trips they serve. We build a simulation environment based on the estimated travel demand for the city of Singapore, and use it to test our assignment concepts.

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

(i) We develop optimization approaches for the dynamics of a practical AMOD system where riders continuously enter and leave the system.

(ii) We build a simulation environment based on the estimated demand for the city of Singapore, and use it to test our AMOD concepts.

(iii) We demonstrate the value of optimization approaches over greedy matching methods in a dynamic AMOD system.

This chapter is organized as follows. Definitions, fundamental concepts and related work are presented in Section 4.1. Proposed methods to solve the assignment problem in an AMOD system are described in Section 4.2. Short notes on the demand management by introducing price incentives are presented in Section 4.6. Finally, Section 4.7 discusses the results for different assignment models and Section 4.8 provides the summary the chapter.

4.1 Introduction to the Assignment Problem

The assignment problem is one of the fundamental combinatorial optimization problems [Bertsimas and Tsitsiklis, 1997]. It attempts to find a maximum weight matching (or minimum weight perfect matching) in a weighted bipartite graph. Mathematically, an assignment is defined as a bijective mapping of a finite set into itself, i.e., a permutation [Burkard and Cela, 1999]. In general, the problem has a number of suppliers and a number of consumers. Any supplier can be assigned to any consumer, incurring some cost that may vary depending on the supplier-consumer assignment. It is required to satisfy all consumers by assigning exactly one supplier to each consumer and exactly one consumer to each supplier in such a way that the total cost of the assignment is minimized [Bertsimas and Tsitsiklis, 1997]. If the numbers of suppliers and consumers are equal, then the problem is called the *linear assignment problem*. Linear assignment problem corresponds to the perfect (maximum) matching in graph theory, which is described in Section 4.1.2.

4.1.1 Basic Definitions and Terminology

In this section we provide definitions we use in the assignment formulation.

Online and Offline Algorithms Online, or dynamic, algorithms attempt to model a real life situations, where the entire input is not available beforehand [Khuller et al., 1994]. The input is obtained incrementally, and the algorithm has to make irrevocable decisions to respond to the input. Offline, or static, algorithms have the a priori knowledge about the entire input. Typically, in real life situations, we have to find solution to the problem with imperfect information. In this dissertation, we study the online algorithms for the assignment problems. To evaluate the performance of an online algorithm, we measure the ratio of its performance to the performance of an optimal off-line algorithm. The performance is measured in the number of served customers and the average waiting time incurred by each algorithm. For most online problems studied in the past [Khuller et al., 1994] randomization helps in improving the performance. The simple deterministic *first come, first served* appears to be the first on-line problem for which provably one cannot do better with randomization.

Representation of the Road Network Road network is represented as a graph. Formal definition of the road network is provided in Section 3.2. For the completeness we provide definition of bipartite graph, which is a special case of the general graph. A bipartite graph is a graph $G = (V, E)$, in which all vertices V are divided into two disjoint sets, $V = V_1 \cup V_2$, and every edge in E connects a vertex in V_1 to a vertex in V_2. A bipartite graph $G = (V, E)$ with partition $V = V_1 \cup V_2$ is said to be a complete bipartite graph if every vertex in V_1 is connected to every vertex of V_2. In general, a complete bipartite graph is not a complete graph.

In this thesis, we focus on two graph representations:

(i) *Road network graph*: road network is represented as a weighted directed graph. The graph is connected and consists of no cycles (no edges *(v,v)* are allowed for all $v \in V$).

The nodes represent points in a metric space. This implies that they satisfy the triangle inequality [Atallah and Blanton, 2009]. The triangle inequality is property that $weight(u, v) \leq weight(u, w) + weight(w, v)$ holds for all vertices u, v, w. Informally, it means that the graph has no short cuts. Many problems involving edge-weighted graphs have better approximation algorithms if the problem is restricted to weights satisfying the triangle inequality [Atallah and Blanton, 2009]. An example of the road network is presented in Figure 3.1.

(ii) *Graph for the assignment*: vehicles and customers are represented as a weighted bipartite graph. Each vehicle v belongs to a set of vehicles V and each customer (passenger) p belongs to a set of passengers P. Note, that both sets are disjoint and all nodes $N_{all} = V \cup P$. Edge $e \in E$, which connects node v and node p represents the cost of assigning vehicle v to passenger p. If edge $e = \{v, p\} \notin E$, this indicates that node v and p cannot be matched. Informally, the edge does not exist if there is no path from location of vehicle v to location of passenger p. An example of the bipartite graph of vehicles and customers is presented in Figure 4.1

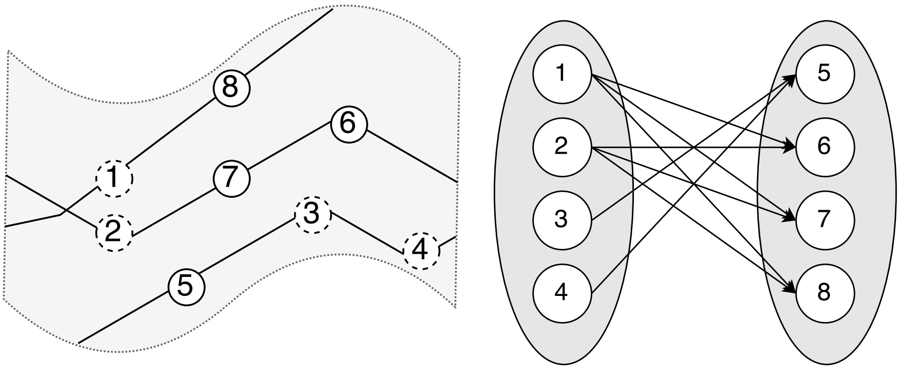
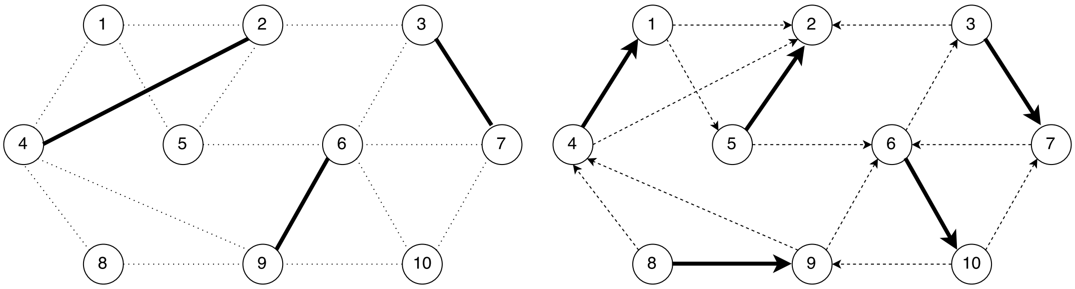


Figure 4.1: Two disjoint sets: vehicles (node 1-4) and customers (node 5-8). Figure on the left shows both sets on a road network, while figure on the right shows its bipartite graph representation. Note, that the two sets are disjoint and there are no edges connecting elements within one set. Not all nodes are connected, i.e., there is no path between vehicle 4 and passenger 8. Weights of the edges are neglected in both representations.

4.1.2 Matching Problem in a Graph

A matching is pairing of items (or persons), so that each item is matched with exactly one other item [Bertsimas and Tsitsiklis, 1997], e.g., it gives an assignment of suppliers to consumers. To solve the matching problem, one have to calculate the cost of all possible assignments. The cost is a function of the distance (or travel time) on the network and the customers waiting time in the queue.

Formally, matching in a graph $G = (V, E)$ is a collection of vertex-disjoint edges, that is, no two edges share a common vertex. The goal is to find a matching which minimizes the total cost of assignment [Bertsimas and Tsitsiklis, 1997]. An example of matching is shown in Figure 4.2. A matching is maximum (or maximum weight) if there is no other matching of larger cardinality (or larger weight). A matching is maximal (or maximal weight) if there is no other matching containing it which is of larger cardinality (or larger weight). A matching is called the perfect matching if the matching contains *n/2* edges (the largest possible), meaning perfect matchings are only possible on graphs with an even number of vertices. A perfect matching is sometimes called a complete matching or 1-factor. Every perfect matching is a maximum matching and corresponds to the assignment problem introduced in Section 4.1.



(a) $\{2,4\},\{3,7\},\{6,9\}$ **(b)** $\{4,1\},\{5,2\},\{3,7\},\{6,10\},\{8,9\}$

Figure 4.2: Examples of matching in a general graph: matching in a general unweighted graph (a), and maximal matching in a directed unweighted graph (b).

Known algorithms for matching in general graphs If the graph is general and unweighted, the Edmonds' algorithm [Edmonds, 1965] and improvement by Micali-Vaziarani [Micali and Vazirani, 1980] can solve the problem in $O(|E|\,|V|^{1/2})$. If the graph is general and weighted, the Edmonds' algorithm [Edmonds, 1965] with improvement by Galil [Galil, 1986] solves the matching problem in $O(|E|\,|V|\,log\,(|V|)$.

Matching Problem in Bipartite Graphs Matching in bipartite graphs is a special case of the general matching problem. A matching in a bipartite graph is a set of the edges chosen in such a way that no two edges share an endpoint. A maximum matching in bipartite graph is a matching of maximum size (with the largest possible number of edges). It is globally optimal. In a maximum matching, if any edge is added to it, it is no longer a matching. There can be more than one maximum matchings for a given bipartite graph. The bipartite matching is defined as a problem of finding a maximum cardinality matching between two sets. A perfect matching is a matching in which each node has exactly one edge incident on it. Matching in a bipartite graph is the perfect matching if the size of the matching equals the number of nodes in each partition. Solution to the matching problem solves only for one passenger in a car.

Known algorithms for matching in bipartite graphs If the graph is bipartite and unweighted, the Hungarian algorithm [Kuhn, 1955] and Hopcroft-Karp algorithm [Hopcroft and Karp, 1973] can solve the problem in $O(|E|\,|V|)$ and $O(|E|\,|V|^{1/2})$, respectively. If the graph is bipartite and weighted, the Kuhn-Munkers (Hungarian) algorithm [Kuhn, 1955] solves the matching problem in $O(|E|\,|V|^{2})$.

4.1.3 Related Work

Safety, traffic congestion, and environmental concerns have recently increased the interest in services that allow people to use personal automobiles more wisely [Agatz et al., 2012]. As we discussed earlier, mobile Internet technology has created opportunity to enable these dynamic and on-demand transportation services, i.e., e-hailing applications. These services—especially through the ride-sharing which aims to bring together travelers with similar itineraries—have the potential to provide huge societal and environmental benefits by reducing the number of cars used for personal travel and improving the utilization of available seat capacity. The heart of the real-time e-hailing concept is the development of algorithms for optimally matching vehicles (or drivers in traditional systems) and customers [Wang, 2013]. We have seen in the literature a growing interest to address the optimization issues in the dynamic assignment for autonomous mobility on demand systems but, as of today, the number of specific contributions is still small. [Agatz et al., 2011] considers the problem of matching drivers and riders for shared rides in dynamic settings. The authors formulate a bipartite matching optimization problem and integer program to solve the assignment. Bipartite matching is used for one-leg trips while the integer program is used when the riders want to schedule a round-trip. They deal with uncertainty by using a rolling horizon approach. Based on the simulation results for the metropolitan Atlanta region, USA, the authors show that their methods lead to ride-sharing systems that generate larger overall system travel cost savings. They also discuss the importance of sufficient numbers of participants to enable dynamic ride-share matches to be established on short notice in practice, because 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. [Xu et al., 2015] attempts to find the complex relations between ride-sharing and traffic congestion to evaluate the ride-sharing enterprises. The authors propose a traffic assignment model that explicitly represents ride-sharing as a mode of transportation. Their results indicate that the ride-sharing price influences the congestion level, and within a certain price range, an increase in price may reduce the traffic congestion. They also show that the utilization of ride-sharing increases as the congestion increases. Therefore, for the AMOD service to be efficient and better than the current systems, we have to carefully consider the adoption policies. [Furuhata et al., 2013] present a classification to understand the key aspects of existing ride-sharing systems. Some of the identified difficulties stem from the requirement of instantaneous coordination of itineraries, schedules, and cost-sharing among participants and building of trust among unknown travelers in the systems. [Wang, 2013] in her dissertation presents matching algorithms for dynamic ride-sharing systems. The author formally defines real-time ride-sharing, identifies optimization problems for finding best sets of ride-share matches, develops approaches for solving ride-share optimization problems, and tests the concepts via a simulation study of work trips in the Atlanta metropolitan area. [Kleiner et al., 2011] propose an auction-based mechanism to determine the driver's compensation. This approach takes into account the different ride valuations of individual riders. The auction mechanism provides rankings of the bids received from passengers. The bids are ranked based on the utility function of each driver. In this environment, higher driver compensations correspond with more ride-share matches because drivers accept longer detours. [Geisberger et al., 2009] suggest to divide the cost of the shared part of the trip evenly between the customers who share the ride. [Amey, 2011] studies the ride-sharing potential at the MIT campus in Cambridge, Massachusetts. The methodology seeks to improve upon previous research by differentiating between modeled ride-sharing potential based on known trip characteristics, and observed ride-share behavior, within the same commuting population. 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 author concludes that disparity between the potential and observed behavior suggests that the barrier to ride-sharing is in human attitudes rather than incompatible trip characteristics.

There are also several papers consider using agent-based simulations with intelligent agents to design a ride-sharing, or assignment, solutions. [Xing et al., 2009] consider an agent-based ride-sharing system with the objective of maximizing the number of served riders. The drivers and riders are matched en-route. The experimental results show that with sufficient number of drivers, dynamic ride-sharing may be an attractive alternative to public transportation. [Atasoy et al., 2015] introduces a demand-responsive personalized services to passengers. The passengers have a flexibility to choose service type (taxi, shared-taxi and mini-bus) from a menu that is optimized in an assortment optimization framework. For operators, there is flexibility in terms of vehicle allocation to different service types. The system is built based on a choice model and consumer surplus is taken into account in order to improve the passenger satisfaction.

The ride-sharing problem can be seen as a special case of the vehicle routing problem with pickup and delivery (VRPPD), which has been studied extensively [Agatz et al., 2012]. These problems involve scheduling of vehicles' routes to satisfy transportation requests at different locations. Typically, to operate the routes, a fleet of vehicles with a given capacity is available. The pickup and delivery problem for ride-sharing focuses on transportation of passengers which implies additional constraints on the passengers' convenience, e.g., total travel time, waiting time, the number of stops while on board. [Cordeau et al., 2007] formulates the ride-sharing problem as VRPPD. However, the VRPPD is NP-hard since it generalizes the Traveling Salesman Problem (TSP) commonly known to be NP-hard. The problem is usually solved with construction of heuristic algorithms [Cordeau et al., 2007]. Another approach known in literature to solve ride-sharing problem is The Dial-a-Ride Problem (DARP). DARP is a particular case of the VRPPD arising in contexts where passengers are transported between specified origins and destinations. In a dial-a-ride system all vehicles typically operate out of one or more depot locations, however the riders do not necessarily arise dynamically over time. [Psaraftis, 1980] study the immediate-request case for the dial-a-ride problem with single vehicle in which a list of requests should be served as soon as possible. The model presented by the author assumes no time windows specified by the customers. The objective function minimizes the sum of route completion time and customer dissatisfaction. The complexity of this algorithm is $O(n^2 3^n)$. [Psaraftis, 1983] extends work presented in [Psaraftis, 1980] to handle time windows on departure and arrival times. The algorithm uses forward recursion (as opposed to the backward recursion presented in [Psaraftis, 1980]), however the complexity remains $O(n^2 3^n)$, which is applicable to solve only small instances. [Dumas et al., 1991] have proposed a set-partitioning formulation and an exact column generation algorithm for the multiple-vehicle VRPPD. This formulation is solved by a dynamic programming algorithm and the results show that the algorithm is successful in solving two real-life instances with 19 and 30 requests. A similar approach was developed in [Savelsbergh and Sol, 1998]. [Xu et al., 2003] proposed another column generation method to address a complex pickup and delivery problem encountered in long-haul transportation planning. The results presented by the authors reported that the algorithm works on larger instances involving 500 requests. [Jaw et al., 1986] propose one of the first heuristics for the multiple-vehicle DARP. Their algorithm imposes windows on the pick-up times of inbound requests and on the drop-off times of outbound requests. The authors have developed an insertion heuristic that selects users in order of earliest feasible pick-up time and gradually inserts them into vehicle routes. The authors show results on artificial instances involving 250 users and on a real data set with 2617 users and 28 vehicles. More recently, [Cordeau and Laporte, 2003] have developed a tabu search heuristic for the problem in which users specify a desired arrival time for their outbound trip and a desired departure time for their inbound trip as well as their maximum acceptable travel time. The search algorithm iteratively removes requests from its current route and reinserts it into another route. The algorithm was tested on randomly generated instances with up to 144 users. Finally, [Agatz et al., 2012] presents a review of the challenges for the dynamic ride-sharing system. Based on the review the authors concluded that there is a growing interest from the research community to address the dynamic ride-sharing but there is still a big room for contributions in the field. In particular, they see the future research in finding fast optimization approaches for real-life instance sizes, incentive schemes to build critical mass and optimization approaches that allow choice. This are the milestones we attempt to address in this dissertation.

4.2 Overview of the Assignment in AMOD Controller

As mentioned in Chapter 2, AMOD Controller optimizes the vehicle to customer assignment. The optimization is performed based on the customer's decisions for single-rider or multiple-riders trips. Examples of single-rider and multiple-riders trips are presented in Figure 4.3. We consider a specific settings for the dynamic assignment for the AMOD service, which is described in Figure 2.16 and Algorithm 2.1. By dynamic assignment, we refer to a system where the AMOD provider matches vehicles and riders on a very short notice, or even en-route. In this setting, the provider of AMOD system receives trip announcements (bookings) from customers. Note, that customers have to be registered in the system before they can make bookings. Every time a customer intends to book a ride, the AMOD provider asks whether the customer is willing to share the ride. The AMOD server provides an estimate of the fare, waiting time and travel time for each option (single-rider or shared trip) to the customer. Based on the estimate the customer may choose to participate in a ride-sharing to reduce travel costs. Each confirmed booking specifies whether the customer intends to be a single-rider, or is flexible to share the trip. Booking also contains an origin and a destination location. A customer announces his or her trip at the time he or she wants to departure (immediate booking). The AMOD systems allows only immediate bookings, and the customers can not book vehicles in advance. With this information, the provider automatically establishes assignments, matching vehicles and riders. A trip announcement (or booking) is said to expire if a successful match cannot be found within a predefined time. The maximum waiting time is adopted differently for different case scenarios, e.g., it is assumed to be equal 10 minutes for the extended CBD region in Singapore. To evaluate the performance of our ride-matching optimizations, we propose a benchmark method which provides an upper bound on the solution quality. The benchmark assignment is the offline (static) model, which is introduced in this section. In the static variant, it it is assumed that

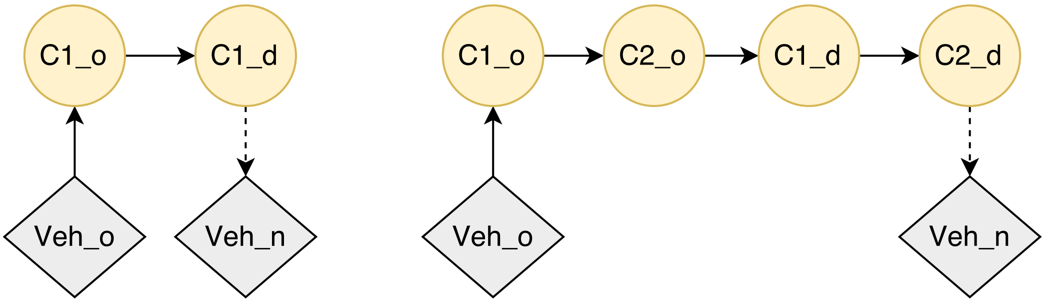


Figure 4.3: Single-rider (left) and multiple-riders (right) trip. For the single-rider trips the vehicle is dispatched from its location to pick-up location (origin of the C1's trip), and then is dispatched to trips destination. Upon arrival at the destination the vehicle may idle at that location or may be send to a new locations. For the multiple-riders the vehicle picks-up first customer, later the second one, followed by dropping off and eventually going to a new location.

all driver and rider requests are known in advance, prior to the execution of a matching process. Given this setting, we explore formulations of the assignment (with and without ride-sharing) problem, in which the AMOD system seeks to minimize total system-wide vehicle-miles, while satisfying all (or almost all due to those leaving the system before being assigned) customers. Note, that 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 customers, it also minimizes total travel costs. Finally, if the AMOD system is compensated for the travel cost savings, the objective is also consistent with maximizing the revenues of the provider.

High level algorithm for the assignment Objective of the assignment is to minimize total transportation cost while satisfying all the supply and demand restrictions. Given two list of customers (willing to share the ride and those who choose single-rider trip), the assignment aims to match available vehicles to waiting customers. This task is performed as described in Algorithm 4.1.

Algorithm 4.1: High level algorithm for the fleet assignment.

```
input  : accepted SingleRiderTrips, accepted SharedTrips, availableVehicles
output : Vehicle2Passenger assignment
if no vehicles available then
    break
end
if no booking queue then
    break
end
for SharedTrips do
    pair customers for shared trip
    match available vehicles to paired customers
end
remove assigned vehicles
for SingleRiderTrips do
    match available vehicles to single-riders
end
dispatch assignedVehicles
erase dispachedVehicles from the list of availableVehicles
```

Algorithm 4.1 describes the high level model for matching vehicles with customers. Vehicle assignment is run as two separate optimizations: for single-rider and shared trips. Input for the algorithm (list of customers willing to share the ride, list of customers not willing to share, list of available vehicles) comes from the output of Algorithm 2.1, which is described in Section 2.5.1. The output consists of the assignment of vehicles to trips. If there is no vehicle available (line 1) or no booking queue (line 4), the algorithm breaks. Otherwise lines 7-16 are executed. For shared trips (line 7-10) we first pair the customers, which is detailed in Section 4.4, and after finding a pair, we search for a vehicle to service the trip. For the trips with one customer (line 12-14), we perform assignment as described in Section 4.3. Next, we dispatch vehicles to the assigned trips and update the list of available vehicles.

The next two sections, Section 4.3 and Section 4.4, describe approach and algorithms for matching without ride-sharing and with ride-sharing, respectively.

4.3 Assignment without Ride-sharing

In this Section we develop optimization-based approaches for finding optimal matches in a standard problem setting, with the goal of minimizing total assignment cost, which is equivalent to minimizing travel time to pick-up all customers.

Assignment, or matching, without ride-sharing aims to match vehicles to the customers, who are single-riders. Single-riders are the bookings with one origin (pick-up point) and one destination (drop-off point). Note, that in a single-rider booking, there may be more than one person on board. We propose two methods to match vehicles to single-riders: (i) greedy assignment, which is introduced in Section 4.3.1, and (ii) minimum weight bipartite matching, which is described in Section Section 4.3.2.

4.3.1 Greedy Matching

To gain some understanding of the value of optimization-based approaches in the vehicle to customer assignment problem, we compare our optimization-based methods with a greedy algorithm. The greedy matching algorithm that we propose is a straightforward approach to match riders and drivers without solving any optimization problem. The greedy algorithm works as follows. Given a set of active bookings, we go over all bookings in the order of their booking time. For each booking we determine which vehicle can pick-up the customer for the smallest cost. If the vehicle in found, we assign it to the trip. In other words, we prioritize the order of booking requests and therefore, whoever requests the ride first, is assigned to a vehicle first (for each request we assign the nearest—in terms of the travel cost—available vehicle).

Algorithm for solving greedy matching Greedy matching is solved as presented in Algorithm 4.2. This is a simple first-in-first-out, FIFO, algorithm.

Algorithm 4.2: Greedy matching for vehicles to single-riders assignment.

```
input  : accepted SharedTrips, availableVehicles
output : Vehicle2Passenger assignment
if no vehicles available then
    break
end
if no booking queue then
    break
end
for SharedTrips do
    findNearestVehicle
    dispatch assignedVehicles
    update the list of availableVehicles
end
```

4.3.2 Bipartite Matching

Suppose that an optimization procedure seeks to find the best vehicle to customer matches from within the current set of open bookings. Therefore, we have two disjoint sets: open bookings and available vehicles. If the total cost of all matches can be expressed as the sum of the individual cost, we can represent this assignment problem using a maximum-weight bipartite matching model. The bipartite matching problem can be solved using any linear programming or network optimization tool. In our model we solve the bipartite matching in batches, i.e., we collect the bookings for a period of time Δt and at the end of each interval we execute the bipartite algorithm. If we do not assign all open bookings at current iteration (e.g., due to a limited number of the available vehicles), we leave them in a queue and consider them during the next iteration, or until they exceed the maximum acceptable waiting time. Note, that the waiting time may be taken into consideration when each individual assignment cost is being calculated.

Formulation Let P and V define the number of passengers in the queue and the number of available vehicles, respectively. For every time interval we solve optimization problem which minimizes the cost of picking-up passengers. The cost of picking-up is a function of the expected passenger's waiting time to be picked-up, which is described by Equation 4.1.

$$c_{ij} = t_d + f_w t_w \tag{4.1}$$

Cost of assigning vehicle i to customer j, c_{ij} $[sec]$, is the sum of waiting time from the successful match (assignment) to the pick-up, t_d $[sec]$ plus waiting time from opening the booking request until the assignment, t_w $[sec]$. Figure 4.4 shows partitions of the waiting time. The waiting time, $t_d = \frac{d_{ij}}{s_{ave}}$ is proportional to the distance between locations i and j, d_{ij} $[meter]$, and divided by the average speed, $s^{ave} = 8.2\ \frac{m}{s}$. The average speed was assumed based on the Singapore statistics [Sun et al., 2014] for the arterial roads in Singapore. The waiting time, t_w, is the time the customer had to wait from the booking confirmation until the vehicle assignment. Factor f_w takes values between 0.0 and the positive infinity to express the value of time before the assignment. This factor was established to highlight importance of time before the assignment as the customers who are in the queue without getting a vehicle are more likely to leave the system.

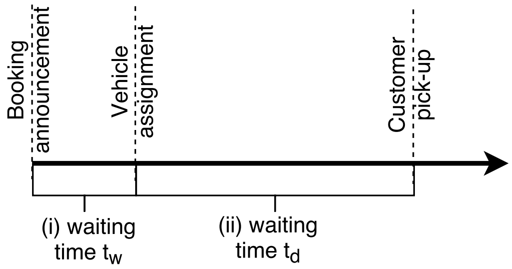


Figure 4.4: Waiting time taken for the matching problem. Note that the total waiting time consists of two parts: (i) the waiting time from booking announcement until vehicle assignment, and (ii) from vehicle assignment until being picked-up.

In formal definition of the problem, we introduce binary variable x_{ij}, which is true if vehicle i is assigned to passenger j, and 0 (false) otherwise. The minimization problem is formulated as follows:

$$\begin{aligned} \text{minimize} \quad & \sum_{ij} (c_{ij} x_{ij} - R) \\ \text{subject to} \quad & \sum_{i} x_{ji} \leq 1\ j \in P \\ & \sum_{j} x_{ji} \leq 1\ i \in V \\ & x_{ij} \geq 0, integer \end{aligned} \tag{4.2}$$

where $R = max(c_{ij}) + 1$ is the maximum cost of assignment increased by one (1). In the objective we minimize a negative value (due to the subtraction of R) of the assignment cost, which implies that we always search for the smallest value of c_{ij}. First constraint ensures that each passenger is assigned to at most one vehicle. Second constraint ensures that each vehicle is assigned to at most one passenger. Last constraint is the integrality constraint.

Finally, we state that the problem always terminates with the best solution, i.e., we always want to add another client in the objective function—which makes our objective function to decrease—while each passenger and each vehicle can only be assigned ones. The objective function in Equation 4.2 is equivalent to maximizing the sum of an inverted cost defined in Equation 4.3.

$$max \sum_{ij} \frac{1}{c_{ij}+1} x_{ij} \tag{4.3}$$

Replacing the objective function in 4.2 for 4.3 gives us the maximization problem with the same constraints.

Algorithms Our goal is to find the maximum matching in a graph. Note that, a maximal matching can be found very easily by just adding edges to the matching until no more can be added. Moreover, it can be shown that for any maximal matching M, we have $|M| \geq \frac{1}{2}|M^*|$, where $|M^*|$ is the maximum matching. Therefore we can easily construct a *2-approximation* to the maximum matching.

Hungarian algorithm for solving bipartite matching The bipartite matching problem is a special case of the binary integer linear programming, which is NP-hard. However, due to the specifics of the problem, there are algorithms to solve this problem. One of the best known algorithms for bipartite matching in a graph is the Hungarian method, or Kuhn-Munkres algorithm [Kuhn, 1955, Mills-Tettey et al., 2007]. We find two different implementations of the Hungarian algorithm [Mills-Tettey et al., 2007]. Both are graph theoretic, one runs at $O(n^4)$ complexity, and the second one, which is harder to implement, runs at $O(n^3)$ complexity. Our implementation of the Hungarian algorithm solves the assignment problem in $O(n^3)$ time, where n is the size of the biggest partition of the bipartite graph.

We formally state prerequisites for implementation of the Hungarian algorithm. Let $G(X, E)$, where $X = V \cup P$ be a bipartite graph, where vertices $V \cap P = \varnothing$ and edges $E \subseteq V \times P$ and *w(v,p)* is a weight of edge *v,p*. For each vertex we assign a label such that $l : V \rightarrow R$. The labeling is feasible if $l(v) + l(p) \geq w(v, p)$. A spanning subgraph of G, $G_l = (V, P, E_l)$, is called an equality subgraph iff it contains edges (*v,p*) such that $(v, p) \in E_l \Leftrightarrow (v, p) \in E \wedge l(v) + l(p) = w(v, p)$. Finally, if M^* is a perfect matching in the equality subgraph G_l, then M^* is a maximum-weighted matching in graph G. Consider we have matching M and $M \subseteq E$. Vertex $x \in X$ is matched if $\exists v \in V : (v, p) \in M \vee \exists p \in P : (v, p) \in M$. Otherwise the vertex is exposed, or unmatched. Path P is called alternating if its edges alternate between M and $E \setminus M$. If both, the first and last vertex, in alternating path are exposed, then the path is called augmenting path. If an augmenting path is found, we can increment the size of the matching by flipping matched and unmatched edges along this path. A tree which has a root in an exposed vertex, and a property that every path starting in the root is alternating, is called an alternating tree.

Pseudo-code for our implementation of the Hungarian algorithm is presented in Algorithm 4.3. A general idea of the algorithm is to maintain both, matching M and equality graph G_l. We start with $M = \varnothing$ and a valid labeling l. We continue until M becomes perfect matching on G_l. At each step we either augment M or improve the labeling $l \rightarrow l'$.

Algorithm 4.3: Hungarian algorithm for vehicles to single-riders assignment.

```
input  : A bipartite graph ($V$,$P$,$E$), where $|V| = |P| = n$ and $n \times n$ edge cost
         matrix
output : A complete Vehicle2Passenger matching
Initialize:
$M = \varnothing$
$\forall v \in V : l(v) = max_{(v,p) \in E}(w(v,p))), \forall p \in P : l(p) = 0$
repeat
    find augmenting path
    if no augmenting path found then
        improve labeling $l \rightarrow l'$
    end
until M is perfect matching on $G_l$
```

Algorithm 4.4 describes the procedure to find augmented path. Note that, to find augmented path some exposed $v \in P, v \notin M$ must exist.

Algorithm 4.4: Algorithm finding augmented path.

```
input  : $V$, $P$
output : augmented path
repeat
    find exposed vertex $v \in P, v \notin M$
    if no vertex found then
        continue;
    end
    create augmenting path $\alpha$ from $v$ to $p$
    Flip the matching by replacing the edges in $M$ with the edges in the augmenting
      path that are in $E_l \setminus M$
until augmented path found
```

In Algorithm 4.4, since we start and end unmatched, this increases the size of the matching $|M'| > |M|$.

To improve labeling, we build alternating tree (Algorithm 4.5).

Algorithm 4.5: Algorithm finding alternating tree.

```
input  : $V$, $P$, $l$
output : $l$'
repeat
    Create array $S \subseteq V$ and array $T \subseteq P$ such that $S$ and $T$ represent the current
      augmenting alternating path between the matching M and outside other edges
      in $E_l \setminus M$
    if $J_l(S) = T$ then
        Compute $\delta_l = min_{v \in S, p \notin T} \{l(v) + l(p) - w(v,p))\}$
        Improve $l \to l'$ based on Eq. 4.4
    end
until l' $is\ a\ valid\ labeling\ and\ E_l \subset E_{l'}$
```

$$l'(r) = \begin{cases} l(r) - \delta_l & r \in S \\ l(r) + \delta_l & r \in T \\ l(r) & otherwise \end{cases} \tag{4.4}$$

Analysis of the complexity of Algorithm 4.3 is the following. We increment matching n times. At each iteration we search for augmenting path, which takes $O(|X|)$ to find an unmatched vertex and $O(|X|)$ to flip the matching. During labeling we update δ_l, which takes at most $O(|X|)$ operations. Finally, updating labels takes another $O(|X|)$ operations if no augmenting path is found. So the total of $O(|X|)$ rounds work $O(|X|)^2$ times in a single round. Therefore, the total complexity of this implementation is $O(|X|)^3 = O(n^3)$.

Assignment with impatient customers In this assignment method we perform exactly the same matching as described in 4.3.2 and 4.3.1, but we assume that if customer is not matched within a short period of time, e.g., 30 seconds, he or she leaves the queue immediately (and we call him or her an impatient customer). Performance of this assignment is evaluated based on the number (percentage) of satisfactorily served customers.

4.4 Assignment with Ride-sharing

In the formulation presented in Algorithm 2.1 and Algorithm 4.1 we operate a static ride-sharing model by maintaining a list of customers who are willing to share a ride. If two trips from the list are matched, we dispatch a single AMOD vehicle. To match customers to shared rides, the network is discretized into zones. If two customers' origin and destination are in the same zones, respectively, then we assign a single vehicle to service the trip (Figure 4.5). Pairing customers within the zone is solved by solving *closest pair of point problem.* The algorithm is executed for trip origins. Vehicles to paired customers' matching is solved as described in Algorithm 4.6.

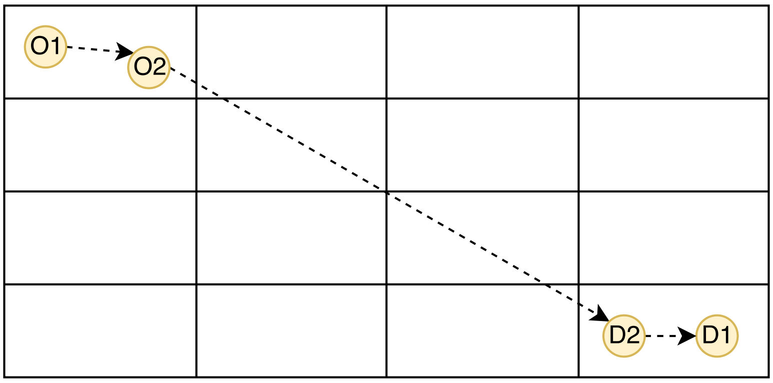


Figure 4.5: Discretization for pairing customers.

Algorithm 4.6: Vehicle to paired customers assignment

```
    Data: pairedTrips; availableVehicles
    Result: Assignment and routing for paired trips
1   if no vehicles available then
2       break
3   end
4   for each pair of customers do
5       vehicle1 = findNearestVehicle(customer 1)
6       vehicle2 = findNearestVehicle(customer 2)
7       if vehicle1 == vehicle2 then
8           findShortestRoute for the vehicle
9       else
10          findBestVehicle
11          findShortestRoute for the best vehicle
12      end
13  end
```

For each customer from the pair, the algorithm finds a vehicle which can service him/her in a shortest time (line 4–6). If we find a vehicle, then the trip order is scheduled (line 7–11), e.g., pick-up customer1, pick-up customer2, drop-off customer1, drop-off customer2.

4.5 Assignment with Priorities

It is a common practice in the assignment problem to prioritize some recipients over others based on priority structure. More formally, priority assignment is to decide who, when and over whom should be assigned to available vehicle. This question is particularly important when the demand exceeds supply (and side payments are not allowed). In the priority structure, each new customer comes with a ranking which describes who is to be prioritized over whom. Ranking usually ranks *classes* of customers, where everyone within the same class is deemed in a tie [Ehlers and Erdil, 2010]. Note that, stable matches with strict rankings fail when ties are introduced [Ehlers and Erdil, 2010]. In our implementation, there is no possibility for ties as we rank everyone individually, not in groups.

In the AMOD system, if the demand exceeds supply, we may decide to change the rules for assignment and give priority to customers who are waiting long to be matched with vehicles. To do that, we modify the assignment cost c_{ij} introduced in Equation 4.1 by giving a higher weight to waiting time t_w. Alternatively, we force FIFO priority, i.e., in bipartite matching, instead of adding slack variables to set of vehicles V, we truncate list of customers, P, to make it of equal size as V. Therefore, we perform assignment only for customers who are waiting in the queue for the longest time.

Other than setting priority for some customers, the operator may prefer to set priorities for vehicles, i.e., in order to maximize profit. This problem is left for the future work. Additionally, it is within our interest to further generalize priority structure to prioritize certain group of customers, e.g., frequent customers, or vehicles, e.g., loyalty drivers.

4.6 Demand Management with Price Incentives

Demand management with price incentives, or dynamic pricing, aims at adjusting cost of service based on certain criteria. The goal of dynamic pricing is to allow AMOD service for real-time modifications of price to better meet demand. It is a very common approach in many industries. Similarly, this approach is receiving a growing popularity in shared-mobility services, e.g., for bike-sharing systems it is common that the system operator gives credits for returning the bike to a less popular destination, e-hailing transportation apps charge more during the peak hours to discourage customers from taking service during high-demand period and encourage drivers to come on road, or airlines change prices depending on the day of the week and number of days before the flight.

In the AMOD service we follow a similar intuition. If we have not sufficient number of vehicles to service current demand, we start surcharging single-riders (indirectly encouraging people to take share rides).

Please note, that our dynamic pricing model is not the main subject of this dissertation. The building block is basic, but easily expendable to handle more sophisticated choices, which is left for the future work.

In the following section (Section 4.6.1) we present our model for the demand management through the price incentives.

4.6.1 Dynamic Pricing in AMOD Controller

Dynamic pricing in AMOD is formulated as a function of booking queue, which is a list of customers who are not assigned to vehicles, yet). Based on the queue we define cost of single-rider and shared trip. Price of a single-rider trip, P_P, is a function of distance between origin and destination d_{OD} times peak period surcharge S_{peak}:

$$P_P = 4 + 0.53 d_{OD} * S_{peak} \tag{4.5}$$

Based on the Singapore's taxi market the booking fee is set at 4$ and 1 km of travel costs 0.53$. Price for shared rides, P_S, is assumed to be cheaper than a single-rider trip. :

$$P_S = S_{shared} * P_P \tag{4.6}$$

Discount factor for shared ride, S_{shared}, is constant and equal to 0.7.

Surcharge factor, S_{peak} gives disincentive to customers for traveling during the peak hours and is calculated based on the relation of current queue and the number of available vehicles, as shown in Equation 4.7.

$$S_{peak} = \begin{cases} 1.0 & \text{if } 0.8V_{avail} - Q > 0 \\ 1.0 - \frac{0.8V_{avail} - Q}{Q + 1.0} & \text{if } 0.8V_{avail} - Q \leq 0 \end{cases} \tag{4.7}$$

Surcharge factor varies from 1.0 to 2.0.

4.6.2 Choice Modeling

Choice modeling represents the decision process of customers. It uses discrete choices in order to infer the decisions. We consider an individual choosing among the available transportation modes. The customer has a set of alternatives and must choose exactly one, known as discrete choice. We assume that once the customer has decided on a mode, she or he utilizes that mode and makes no changes to the decision.

More formally, Customer i has J modes (*j=1,2...,J*) to choose from. There is a vector of observable characteristics of each mode. The choice set, or the set of available modes, consists of

(i) AMOD single-rider trip,

(ii) AMOD shared trip, and

(iii) not AMOD trip.

We assume that there are K observables for each mode:

(i) expected waiting time, t_w,

(ii) expected travel time, t_t, and

(iii) estimated fare to be paid, f_e.

We assume that each customer makes her or his choice in two steps:

1. Calculating the characteristics of each mode in the choice set, and
2. Selecting the mode with the highest value of the utility.

Each customer i is characterized by two parameters: (i) value of time, p_t, and (ii) value of money, p_m. Both factors take values from interval $< 0, 1 >$, where 0 means the highest sensitivity and 1 that the customer is not sensitive to a given observable. The utility function u for each mode j is designed as presented in Equation 4.8.

$$u_j = \frac{t_w + t_t}{t_{max}} p_t + \frac{f_e}{f_{max}} p_m \tag{4.8}$$

where, t_{max} and f_{max} are the maximum time allowed for the trip and maximum fare which can be paid, respectively.

The customer makes the choice, cc, which is to select the mode with the highest utility, based on Equation 4.9.

$$cc = max(u_j) \tag{4.9}$$

Decision process In this dissertation, we model decision process in two ways: fixed (a priori) decision and decisions based on customer's characteristics. In the fixed decision model, we assume that a specific percentage of customers always takes shared rides (refer to analysis presented in Chapter 6). The fixed decision model helps us to evaluate how different percentage of the shared trips influences the network performance. The model, which takes customers' decisions is set to test different pricing strategies.

4.7 Results

In this section we present performance comparison for different assignment algorithms. Comparison of the assignment methods is evaluated in terms of (i) number of customers served, and (ii) total distance traveled for the assignment. For each assignment method, we analyze two cases, where:

1. the number of vehicles is sufficient to serve the trips (off-peak period), and
2. the number of vehicles is not sufficient to serve the trips (peak hours).

Finally, we check how the performance changes if the network information is not perfect, i.e., we perform matching based on the Euclidean distances instead of the current network travel times (please refer to our paper [Meghjani and Marczuk, 2016]). Note, that a complete analysis of the the performance of an AMOD system is presented in Chapter 6.

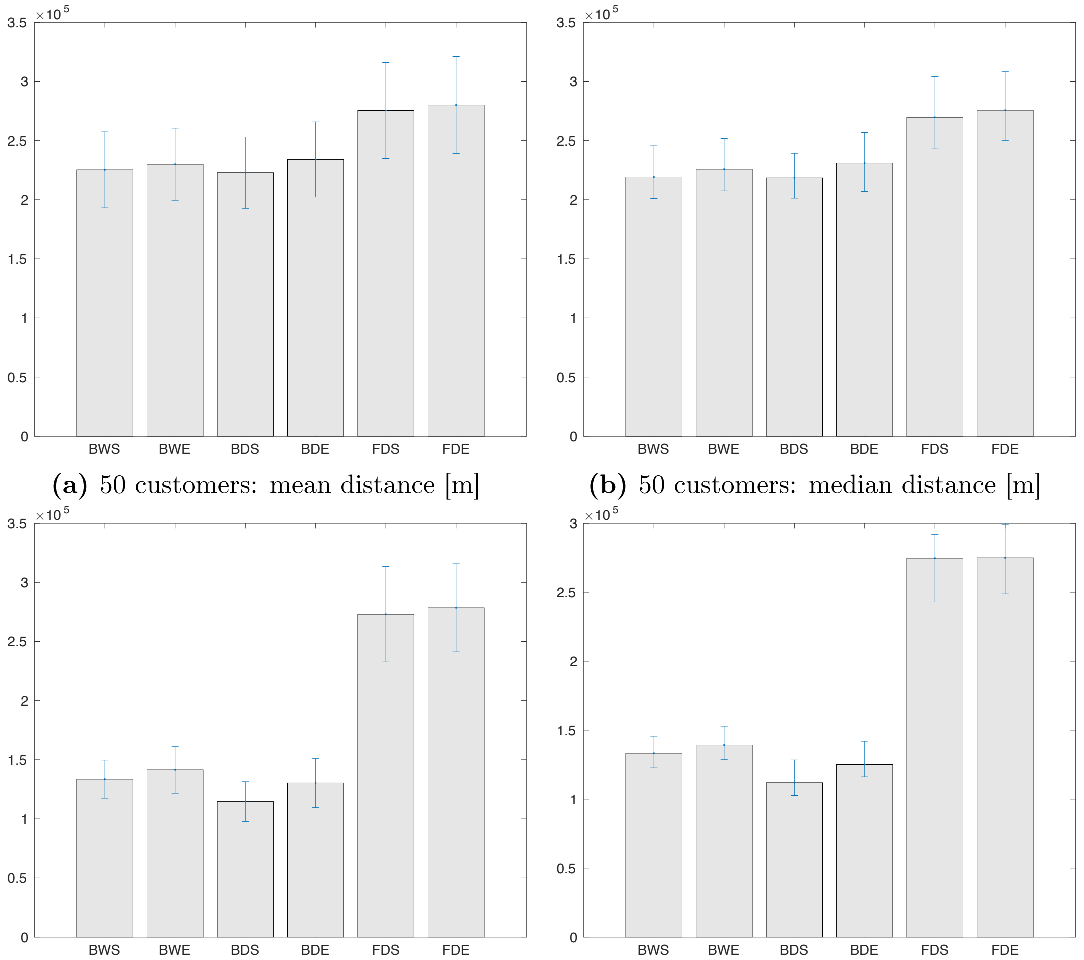
Data We use the Household Interview Travel Survey 2012 (HITS 2012) data for the entire city of Singapore. HITS is a comprehensive survey in Singapore which is conducted every four to five years, that provides relevant indicators on household travel patterns. A total of about 10,500 households participated in the 2012 survey, which accounted for about 1% of the total number of households (1,14 million). The survey represented 84% of total population.

We run the analysis on the network of 5424 nodes. To evaluate performance of the matching algorithms, 50 vehicles are initialized at 50 stations, which implies that we initialize 1 vehicle per station. Stations locations are optimized as described in Chapter 3. For the case where the number of vehicles is sufficient to serve the trips, 50 trips are sampled from the the synthetic population of SimMobility Mid-Term. For the case where the number of vehicles is not sufficient to serve the trips, 75 trips are sampled from the the synthetic population of SimMobility Mid-Term. Note, that we only sample the trips which are performed on mode *taxi*. Each trip request consist of origin and destination location and time when the booking was made. We limit sampling to the period of 30 minutes during the morning peak hours, so that the maximum waiting time is bounded within that period.

Algorithm For this analysis we use the static, or benchmark, algorithms for Greedy and Bipartite Matching. We adjust parameters in the cost function (Equation 4.1) to account for the waiting time. We assume, that there is no booking lead-time, meaning that the customers are not allowed for the advanced booking. Therefore, the time from a booking request to customer pick-up is referred as the waiting time for being serviced. We also assume that customers do not leave the system if a vehicle is assigned to their booking. However, if a booking remains in the queue for over 10 minutes without any vehicle being assigned, then this customer leaves the system. If a customer drops his or her booking, the supply side is penalized by the maximum waiting time (assumed to be equal 10 minutes) added in the objective function.

Results and discussion Mean and median distance traveled to pick-up customers for both cases, with sufficient and insufficient number of vehicles, are presented in Figure 4.6.

In the case with sufficient number of vehicles (Figure 4.6a and Figure 4.6b) we simulate 50 booking requests and 50 available vehicles. In the second case (Figure 4.6c and Figure 4.6d) we increase the number of bookings by 50% to simulate 75 customers and 50 vehicles. For each case we generate 40 sets of data and present averaged distance for each algorithm. Figure 4.6 shows the mean and median distance traveled to pick-up all assigned customers.



(a) 50 customers: mean distance [m]

(b) 50 customers: median distance [m]

(c) 75 customers: mean distance [m]

(d) 75 customers: median distance [m]

Figure 4.6: Mean and median distance traveled to pick-up 50 customers. Note that, the number of customers is equal to the number of available vehicles. BWS—bipartite matching, waiting time included, distances based on the SimMobility's network; BWE—bipartite matching, cost based on the Euclidean distance and waiting time; BDS—bipartite matching, cost based on the distances from SimMobility; BDE—bipartite matching, cost based on the Euclidean distance; FDS—greedy matching, cost based on the distances from SimMobility; FDE—greedy matching, cost based on the Euclidean distance; The total distance of each matching is in meters and considers all distances vehicles have to traverse to pick-up customers on SimMobility's network, regardless of the assignment cost.

We perform assignment based on Greedy Algorithm (FDS—greedy matching in SimMobility 's network, FDE—greedy matching in Euclidean space) and Bipartite Algorithm (BWS—bipartite matching with cost based on distances from the SimMobility's network and waiting time, BWE—cost based on the Euclidean distance and waiting time, BDS—cost based on the distances from SimMobility, BDE—cost based on the Euclidean distance). We evaluate performance of each algorithm based on the SimMibility's network distances. Note, that total distance for each algorithm is presented in meters and considers the sum of SimMobility's distances, which vehicles have to drive to pick-up customers.

As presented in Figure 4.6, for each case, the mean distance is longer than the median, which indicates that there are some trips which take very long time with a fewer number of very short trips. In the sufficient supply case without considering waiting time when performing assignment, on average, each vehicle has to travel 4.4 km to pick up a customer. In the insufficient supply case, an average distance reduces to 3 km. This may be due to a bigger choice of customers to select from. However, when we match based on Greedy Algorithm, the average driving distance to pick up all customers increases almost twice. This finding highlights importance of the optimization techniques in modeling the assignment problem. Furthermore, We can see in Figure 4.6 a general trend across different cases that deterministic greedy algorithm (last two columns in each figure) yields on the average 17% increase in the distance traveled to pick-up customers as compared to any more sophisticated optimization. Taking into account waiting time (first two columns) always makes the total distance traveled worse. Mean distance to pick up all customers increases by 2% on average, if we give priority to customers who are longer in the queue. Therefore, from the operator's perspective, it is ultimate to assign vehicles based on the bipartite matching with a perfect knowledge of the network. From the customer's perspective, the waiting time is a crucial factor describing reliability of the service. The average waiting time for the unbalanced case shows the opposite trend to the one for the driving distance. Note, that in the case of not sufficient supply, both metrics driving distance and waiting time are not correlated. If we consider waiting time in the objective function, we serve long-waiting customers first preventing them from dropping the booking. The average waiting time is smaller in the case where the cost function accounts for queuing time. This decrease is explained by a smaller number of penalties due to dropped bookings. Finally, different cost function force us to select different set of customers, e.g., if we consider waiting time, the customers are selected with the consideration of the time they have already spent in the queue. If we assume to perform *BDS* assignment (cost function based on the network distance only), the mean waiting time of customers increases by 12% compared to the assignment which accounts for the waiting time. This translates to increase in the average waiting time of nearly 1.8 minutes, and savings in driving distance of almost 50%. Concluding, we believe that it is not desired to include waiting time in the objective function because it increases the driving distance significantly, which in a long run may reduce vehicles' availability.

4.8 Summary

In this chapter we present the assignment methods to match vehicles and customers. We define assignments methods for single-rides as well as for multiple-riders trips. We also present algorithms for demand management through price incentives, which aim at realigning demand and supply. Our simulation results indicate that the use of optimization methods instead of simple greedy matching techniques substantially improves the performance of AMOD systems. Furthermore, including waiting time in the objective function decreases the average waiting time by 12% and increases driving distance by almost 50%. Based on our results, it is not desired to include waiting time in the objective function because it increases driving distance significantly, which in a long run may reduce vehicles' availability.

In our future work we are interested in extending the concept of providing door-to-door transportation to integrate it with other modes of transportation, such as public transit. We believe that the idea of an AMOD system may provide a very effective alternative to increase the usage of public transportation systems, i.e., by providing solution to first and last mile problem. In such a setting, an AMOD vehicle would bring rider(s) from their origins to a public transport hub, i.e., MRT station in Singapore, then the riders would use public transit to get close to their destinations, and finally they would walk or use another AMOD vehicle to travel from the transit stop to their destinations.

We are also interested in improving our priority assignment. Other than setting priority for some customers, we may prefer to set priorities for vehicles, i.e., in order to maximize profit. We can further extend our priorities to prioritize certain group of customers, e.g., frequent customers or certain vehicles, e.g., with higher fuel economy. It is also within our interest to extend the assignment model to account for the vehicle's battery level before dispatching it to a customer.

Chapter 5

Rebalancing

Efficiency of an AMOD system is related to many operational issues. As the distribution of the demand varies over the day, vehicles become unbalanced, accumulating at popular destinations and becoming depleted at less popular locations. To mitigate this issue we propose rebalancing polices which aim in moving vehicles based on the fluctuating demand. Empty vehicle rebalancing aims in finding optimal control policies for realigning demand and supply by minimizing the total cost of rebalancing. Rebalancing is known from variety of transportation problems, e.g., empty container repositioning in shipping industry, rail car redistribution in rail transportation, rebalancing in bike-sharing and car-sharing systems. Self-driving vehicles hold a great promise in mobility on demand, because they can rebalance themselves without hiring drivers.

Contribution of this chapter is the following:

(i) We develop optimization approaches of the static and dynamic rebalancing policies for an AMOD system. Static rebalancing solves for the total number of vehicles required to operate the system and the number of rebalancing trips between different locations for the course of the entire day. Dynamic rebalancing solves rebalancing problem at *real-time*.

(ii) We integrate our optimization techniques within a simulation environment and test the algorithms for the case studies placed in Singapore.

(iii) We demonstrate the value of optimization-based rebalancing policies in the operation of the AMOD system.

This chapter is organized as follows. Section 5.1 provides background information relaed to rebalancing of mobility on demand services. Section 5.2 describes our approach to solve rebalancing for an AMOD system. Details of the static and dynamic models are presented in Section 5.3 and Section 5.4, respectively. Section 5.5 presents results for different rebalancing policies and Section 5.6 summarizes the chapter.

5.1 Introduction to the Rebalancing Problem

A crucial factor in the success of any mobility-on-demand system is its ability to meet the fluctuating demand for service [Raviv et al., 2013]. In order to reduce shortages, operators of shared systems are responsible to regularly transfer vehicles from some locations to others. We refer to this activity as rebalancing, or repositioning, of vehicles. The goal of this operation is to minimize the shortage of vehicles incurred by moving people in the system. Example of an imbalanced system is presented in Figure 5.1. In the figure, we can observe that the top right (TR) station is heavily underserved, while there are vehicles available at the bottom right (BR) station. The rebalancing is to keep the system in balance and to maintain accessibility from different locations. In our example we should move vehicles from BR to TR station.

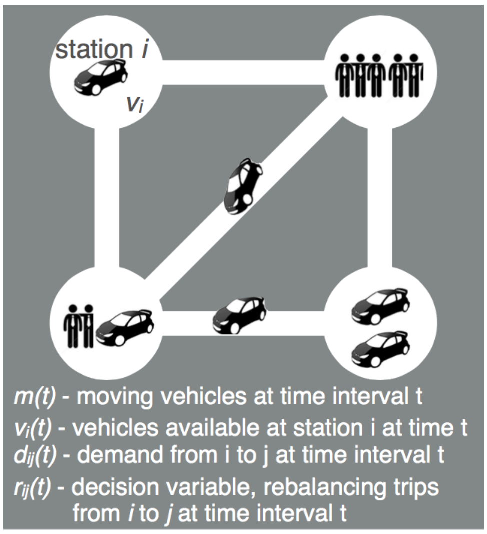

station i
v_i
$m(t)$ - moving vehicles at time interval t
$v_i(t)$ - vehicles available at station i at time t
$d_{ij}(t)$ - demand from i to j at time interval t
$r_{ij}(t)$ - decision variable, rebalancing trips
from i to j at time interval t

Figure 5.1: A simple example of an unbalanced AMOD system, i.e., we can observe that the top right station is heavily underserved, while there are vehicles available at the bottom right station. The rebalancing aims to align the demand with the supply.

5.1.1 Terminology, Definitions and Assumptions

There is a promise, that rebalancing can be automated in the coming decades as the research on autonomous vehicles is very active [Spieser et al., 2014, Pavone, 2015, Le Vine et al., 2015]. On the academic side, algorithms for automated vehicle redistribution are beginning to be explored [Barrios and Godier, 2014].

There are various approached to solve the rebalancing problem. Here, we distinguish between static and dynamic rebalancing mode. In static mode we solve the rebalancing for the entire day and given the knowledge of a general daily trip pattern. In the static version, a snapshot of the level of occupation at the stations is taken and then used to plan the redistribution.

Dynamic mode is solved during operation of the system and serves as adjustment to the static solution. In the dynamic version, the real-time usage of the system is taken into account, and the rebalancing counts are updated as soon as the information required to make decisions is available. Static mode of operation benefits from a practical advantage of pre-computed decisions and when combined with dynamic repositioning, it reduces the amount of work required in the latter mode. Dynamic repositioning takes place during the day in order to cope with looming shortages. Usually, static rebalancing is associated with a redistribution process that is performed during the night, when the system is kept closed or the demand is very low, whereas dynamic rebalancing is associated to redistributions operated during the day, when demand may be high. In practice, many operators work in both modes.

For each model we have different rebalancing policies which are based on the amount of rebalancing:

1. Zero rebalancing, where an AMOD system does not have a systematic redistribution strategy;
2. Periodic redistribution, where we perform rebalancing one time (or a few times) a day, e.g., overnight or before or after peak hours (periodic repositioning is usually performed during the night, when the system is nearly idle); and
3. Continuous redistribution, which involves relocating cars while the system is in operation.

In this dissertation we test different rebalancing models and strategies for an AMOD system. The AMOD system is understood as one-way, free-floating mobility on demand service with autonomous vehicles. We assume that autonomous cars can rebalance themselves without the need to hire drivers. We test our models in simulation environment of SimMobility. The detailed simulation results are provided in Chapter 6. The following section describes related work on rebalancing for mobility on demand services.

5.1.2 Related Work

In order to increase reliability of shared-use mobility systems the operators are responsible to regularly rebalance their fleet of vehicles, e.g., in bike-sharing schemes the rebalancing is referred to removing bicycles from the most popular destination stations and transferring them to the most popular origin stations, using a dedicated fleet of trucks. In car-sharing schemes, the process is more challenging because vehicles (cars) can not be easily loaded on the trucks and the operator needs a dedicated staff to perform repositioning. For all shared-use mobility services, the goal of the rebalancing is to minimize the number of shortages incurred in the system and the fleet operational costs. Consequently, empty vehicle repositioning involves (i) routing decisions concerning which vehicles should be sent and to where, and (ii) inventory decisions concerning the number of vehicles to be removed or placed in each station (zone).

Research on empty vehicle rebalancing for mobility on demand services is growing in popularity, especially in the last few years. It is shown that one of the main complaints from the bike-sharing users relates to unavailability of bicycles at origins and unavailability of lockers at their destinations [Raviv et al., 2013]. This pushed many researchers and operators to look for solutions to the problem. [Raviv and Kolka, 2013] discusses that meeting the demand for bicycles and vacant lockers is challenging due to inherent imbalances in the renting and return rates at the stations. The authors claim that BSS may be used as a partial substitute for other transportation modes, which in turn may cause vehicle imbalances. The rebalancing issue, especially for BSS, has been research quite widely. [Vogel and Mattfeld, 2010] model the repositioning activities with the help of a system dynamics approach which is solved with a simulation tool. This contribution assesses the prospects of operational repositioning services by means of an aggregate feedback loop model. The authors adopt a clearing function to model the probability of successful rentals under a certain number of requesting users in the system. They model two, short- and mid-term, rebalancing approaches: direct repositioning activities and indirect customer based distribution through pricing and incentives. They conclude that more effort spent on repositioning leads to a better performance in terms of satisfied customers. Their model, however, is useful for strategic planning but is not detailed enough to support repositioning operations. [Farahani et al., 2014] explore efficient operation of BS system. The authors propose algorithms for relocation of empty bikes and a scheme that uses price incentives to encourage users to change the destination of their trips in a way to reduce the rebalancing effort. [Sayarshad et al., 2012] demonstrates discrete formulation of rebalancing problem for a bike-sharing system. To minimize the fleet size and maximize the usage of bicycles, the proposed model distributes bicycles between stations to better meet the demand. A case study of a four stations in the city of Tehran shows that the model minimizes the unmet demand as well as number of rebalancing vehicles. However, the authors did not evaluate the model for a large-scale network. [Bruglieri et al., 2013] formulates a generalization of the many to many pickup and delivery problem to solve the rebalancing problem for a bike-sharing system. The authors propose a branch-and-cut algorithm and tabu search for solving a relaxation of this problem. This work restrains to the rebalancing problem with one vehicle and in the static case. The authors in their formulation perform rebalancing once a day during the night when the number of moving bikes is negligible. They divide the city into districts. Each district is covered by a single truck that has to redistribute the bikes in order to respond to the morning peak at best. Te authors optimize the route for each vehicle which is performing redistribution of the bikes. They show that solution of the relaxation problem provides a good lower bound of the optimal solution of the original problem. [Erdogan et al., 2012] extend the above studies by allowing the final inventory at each station to be within a pre-specified interval instead of at a given target value. [Raviv and Kolka, 2013] introduces a closed-loop inventory model suited for the management of bike rental stations. The authors introduce a user dissatisfaction function (UDF) to measure the performance of each station and present a dynamic inventory model of a bike-sharing rental station. UDF is the initial inventory function and measures the expected penalty caused by a shortage of bicycles and lockers at a station. They model each single station neglecting interactions between different stations as the goal is to find an optimal inventory for each station separately. The main goal of the static repositioning in Tel-O-Fun is to reduce the amount of repositioning work to be done during the day. Therefore, the authors evaluate quality of the solution by counting surplus or deficit bicycles at each station. Although their model considers only a single station system, they demonstrate through simulation that their results are robust to the inherent interactions between stations in a large real system. They run the repositioning by incorporating the UDF into the mixed integer programming formulation presented in [Raviv et al., 2013]. [Raviv et al., 2013] extends work presented in [Raviv and Kolka, 2013] focuses on the static mode of rebalancing for bike-sharing system. The input of the problem is a set of stations, initial inventory, capacity, penalty function for each station, travel-time between stations, and a set of nonidentical capacitated repositioning vehicles. A solution is defined by a route for each vehicle and the quantity of bicycles to load or unload at each station along this route. The authors demonstrate that their MILP formulations are capable of solving problems of a moderate size of up to 60 stations with acceptable optimality gaps. [Caggiani and Ottomanelli, 2012] presents a fuzzy decision support system for redistribution process in BSS. The aim of the proposed method is to minimize the redistribution costs for bike-sharing companies, determining the optimal bikes repositioning flows, distribution patterns and time intervals between relocation operations, with the objective of a high level users satisfaction. The core of the forecasting demand method in the proposed approach is based on Artificial Neural Networks (ANN) and Fuzzy Logic. Method presented by the authors determines the relocation time windows, the optimal carrier vehicles route and the number of bikes to be repositioned. The approach has only been tested on a small networks. [Caggiani and Ottomanelli, 2013] extends work presented in [Caggiani and Ottomanelli, 2012] and presents a simulation model for dynamic bikes redistribution process. The proposed model considers dynamic variation of the demand, for both bikes and free docking slot. Their simulation-based solution determines the optimal repositioning flows, distribution patterns and time intervals between consecutive rebalancing by explicitly considering the route choice for trucks among the stations. The results are compared to those based on fuzzy decision support system [Caggiani and Ottomanelli, 2012]. Authors show that for the case of low demand their model with constant relocation time intervals performs better (in terms of number of satisfied users) than the method with variable rebalancing time window. As the congestion increases, the model based on the fuzzy decision support system, shows better performance. [Ciari et al., 2014a] addresses the static bike-sharing rebalancing problem as a special case of one-commodity pickup-and-delivery capacitated vehicle routing problem. In this problem a fleet of capacitated vehicles is employed in order to redistribute the bikes. The authors present four mixed integer linear programming formulations of this problem and solve them by branch-and-cut algorithms. They test their mathematical formulations using data obtained from twenty two (22) different bike-sharing systems and show that their formulation efficiently solves instances with up to 50 stations. [Contardo et al., 2012] formulates dynamic version of the rebalancing problem for bike-sharing system, allowing the use of more than one vehicle to perform repositioning. An arc flow formulation working on a discretized time horizon is proposed and solved with Dantzig Wolfe and Benders decomposition. This methodology uses decomposition techniques to move the difficult variables and constraints into the subproblems, which can be solved efficiently. Their formulation, however, cannot handle medium or large scale instances.

Another approach to rebalancing seen in the literature is offering incentives to encourage customers to use certain stations. Some operators may also give incentives for user-based redistribution, i.e., in which the rider performs redistribution, by using demand-based pricing in which users receive a price reduction or credit for docking bicycles at empty docking locations [Shaheen et al., 2010, Vogel and Mattfeld, 2010, Fricker and Gast, 2014]. [Fricker and Gast, 2014] considers a stochastic model with incentives to redistribute bikes in bike-sharing system. To compensate for real-time congestion problems, an alternative is to use real-time pricing mechanisms. This type of congestion control mechanism is widely applied in the transportation or car-rental industry [Yang et al., 2010, Guerriero et al., 2012]. The authors show that simple incentives, such as suggesting users to return the vehicle to a less loaded station close to their destination, improve the situation by exponential factor. The authors compute rate at which bikes have to be redistributed by trucks to ensure a given quality of service. They also study a variant of the model where users know which stations are empty or saturated. The customers always arrive to non-empty stations and return their bikes only to non-saturated stations. They investigate the influence of the station capacities on the performance of homogeneous bike-sharing systems. Using a stochastic model and a fluid approximation, they provide analytical expressions for the performance. They prove that, without repositioning via incentives or trucks, performance is very poor. [Forma et al., 2015] proposes a 3-step algorithm for the static rebalancing problem. The authors first cluster the stations according to geographic and inventory considerations. Then, they route vehicles through the clusters . Finally, the original static repositioning problem is solved for all stations. [Ji et al., 2014] describes the operational concepts and system requirements of a fully automated electric bike (e-bike) sharing system demonstrated through a pilot project at the University of Tennessee, Knoxville (UTK) campus. The authors extend the bike-sharing literature by incorporating e-bikes into the system, focusing on system configuration, and managing supply to meet demand through a simulation approach. The e-bike system is more attractive to casual riders, who may not otherwise consider traditional bicycles as a viable transport mode. Their study focuses on three demand variables: trip generation, trip length, and trip duration. The results show that by applying quick-charging (reducing the recharge time to half of the baseline), the number of batteries out of service can be lowered dramatically. However, the service rate for the system does not increase substantially because of the limited number of available e- bikes. Therefore, a pre-defined duration for users to return the e-bikes is crucial to the system, balancing system performance, capital cost, and user convenience. Based on the simulation results, we found that the minimum number of e-bikes and batteries required by the system are significantly sensitive to trip rate, trip length, and activity duration.

There are major differences between bike- and car-sharing systems in terms of repositioning. A car is moved from one location to another by assigning a driver to it, while bicycles are moved in batches by repositioning vehicles. Today, most CS operators use traditional station-based system offering round-trip rental [Privat, 2015]. Traditional business model, which requires cars to be collected and returned at the same location is easier to manage by the operator than a free-floating, or point-to-point model. The drawback is that it becomes uneconomical for trips that require lengthy stays at the destination, because the user must also pay for the time the car is parked. For this reason point-to-point services can quickly attract three to four times the number of members of a traditional round-trip service [Brown, 2015]. However, for the operator, the cost of point-to-point systems is much higher since the systems requires larger fleet size to start with [Brown, 2015] and is more challenging from the operational perspective due to rebalancing issues. The findings summarized in the literature review presented in [Jorge and Correia, 2013] show that the major issue in development a one-way car-sharing is how to balance the demand and supply [Jorge and Correia, 2013]. To estimate the minimum required fleet size, many researchers have focused on rebalancing strategies for both station-based and free-floating carsharing system [Alshamsi et al., 2009, Barrios and Godier, 2014, Barth and Todd, 1999, Brownell and Kornhauser, 2014, Ciari et al., 2014b, Correia and Antunes, 2012, Fan et al., 2008, Horn, 2002, Jorge et al., 2014, Kek et al., 2009, Maciejewski, 2014, Nourinejad and Roorda, 2014, Nourinejad and Roorda, 2015, Pavone et al., 2011, Smith et al., 2013]. [Barth and Todd, 1999] develop a simulation model of car-sharing operations and conclude that a sufficient fleet size for satisfying customers is 3-6 vehicles for every 100 trips but that 18-24 vehicles per 100 trips are required to minimize relocation costs. This conclusion shows the importance of robustness of FMS. [Fan et al., 2008] propose a multi-stage stochastic linear integer model which attempts to capture system uncertainties such as car-sharing demand variation. The objective function of their model maximizes the revenue obtained from servicing customers while minimizing the cost of vehicle relocation. [Kek et al., 2009] design a mixed integer linear program to determine a set of near-optimal manpower and operating parameters for the vehicle relocation problem. Simulation tests, based on a set of commercially operational data for the city of Singapore, indicate that optimization of manpower can reduce staff expenses by up to 50% and zero vehicle time (duration of vehicle shortage at parking stations) by up to 13%. The model however cannot be applied for a real-time operation. [Pavone et al., 2011] and [Smith et al., 2013] show a theoretical solution to fleet sizing by introducing rebalancing assignments that minimize the number of empty vehicles traveling in the network and the number of rebalancing drivers needed, while ensuring stability. They introduce a rebalancing policy based on a fluidic model. Using both theoretical and simulation results, the authors determined the minimum number of vehicles required to maintain system stability. [Correia and Antunes, 2012], on the other hand, focus depot location in one-way car-sharing systems where vehicle stock imbalance issues are addressed. Considering all decision variables (depot locations, satisfying demand and relocation), the authors present a mixed integer optimization approach to maximize the revenue of the car-sharing operator while minimizing operating cost such as vehicle maintenance, parking provision, vehicle depreciation, and vehicle relocation. Based on the case study of Lisbon, their results show that the depot location and trip selection schemes have an impact on the profitability of such systems.

5.2 Rebalancing in AMOD Controller

As described in details in Section 2.3, the core component of AMOD Controller is responsible for the fleet management of AVs, i.e., assignment, routing, and rebalancing. The basic idea is that AMOD Controller sends dispatches to a microscopic simulator, which handles vehicle movements and provides tracking information for all vehicles in the system. In this framework we do not treat each decision level separately as the strategic and tactical decisions influence directly operational decisions. When considering strategic and tactical decisions, we take into account operational matters. In this settings we implemented two models of rebalancing: static and dynamic model (Figure 5.2).

Static Rebalancing is solved at the tactical level of the operation based on a snapshot of the demand for the entire day at all stations which is used to plan the redistribution.

Dynamic Rebalancing is solved at the operational level based on the real-time usage of the system, which is used to update the rebalancing counts as soon as the information required to make decisions is available.

Figure 5.2: Rebalancing models. Our rebalancing problem consists of two submodules: static and dynamic rebalancing. In our approach, the dynamic model is solved during operation of the AMOD system and serves as an adjustment to the static solution.

Approaches to solve rebalancing In this dissertation we analyze static and dynamic rebalancing model. In the static version, a snapshot of demand for the entire day at the stations is taken and then used to plan redistribution. In the dynamic version, the real-time usage of the system is taken into account, and the rebalancing counts are updated as soon as the information required to make decisions is available. In our approach, the dynamic model is solved during operation of the system and serves as adjustment to the static solution. From the perspective of the system manager, the static model solves problem of strategic planning and long-term operation, specifically what is the number of vehicles needed and how to distribute them between stations. The dynamic model solves for the short-term operating decisions, specifically how to move vehicles based on the current situation on the network. Both models are described in details in the following sections (static model is described in Section 5.3 and the dynamic model in Section 5.4)

Rebalancing strategies For each, static and dynamic, model we have different rebalancing policies which are designed based on rebalancing frequency. In our methodology we distinguish three rebalancing policies:

(i) No rebalancing, in which vehicles are only moving when assigned to customers and are parked at the destination of the trip. This is our baseline scenario for the analysis.

(ii) Periodic rebalancing, where we perform rebalancing one time a day, e.g., overnight or before peak hours (periodic repositioning is usually performed during the night, when the system is nearly idle).

(iii) Continuous rebalancing, which involves relocating cars while the system is in operation.

The next two sections will describe optimization models for static and dynamic formulation of rebalancing for an AMOD system, respectively.

5.3 Static Rebalancing and Fleet Size

The static rebalancing model presented in this section evaluates a dispatching strategy that makes decisions by finding an exact solution of the offline problem at each decision epoch. This model extends the linear program introduced in [Pavone et al., 2011]. The objective of our model is to find the minimum number of vehicles at each station at the beginning of the day (Fig. 5.1). As proven in [Pavone et al., 2011] this is equivalent to minimizing the rebalancing effort. This policy requires a priori knowledge of the demand $d_{ij}(t)$. We assume that this knowledge is available through the estimated demand. Note that, the demand cannot be validated because the service is yet not in operation. This strategy is meant to serve as a benchmark for real-time, or dynamic, strategies. In this model, we plan the number of rebalancing trips between stations based on the estimated demand for the entire day. The static model is solved at the tactical level and solution to the model gives us an estimate of the fleet size required to run the service. We use n number of discrete time intervals, t, and evaluate the trips within each interval. In the static model we assume that all trips occur between stations, i.e., all customers board and alight at stations (or centroids of the zones) and all rebalancing trips are from and to the same station. Additionally, we assume deterministic travel time between all stations.

The mathematical model is formulated as follows. Let $v_i(t)$ and $d_i^w(t)$ be the number of vehicles owned by station i at time t and the anticipated number of vehicle arrivals with customers at station i at time t such that $d_i^w(t) = \sum_j [d_{ji}(t - \tau_{ji}) - d_{ij}(t)]$. Note that $d_i^w(t)$ is the sum of vehicle arrivals minus departures for each station. It can be understood as a trip flow at each station. Two decision variables are the number of empty (rebalancing) vehicles to send from station i to station j at each rebalancing period, $r_{ij}(t)$, and the total number of vehicles owned by each station i at time t, $v_i(t)$. The total number of vehicles at each rebalancing period, $N(t) = \sum_i v_i(t) + m(t)$ where *m(t)* is defined as the number of vehicles in transfer between stations during interval t. This formulation guarantees that N is constant over time. Our objective is to minimize the total number of vehicles, N, at each period of time i. The number of vehicles owned by station i, at any time t, $v_i(t) = v_i(t - \Delta t) + \sum_j \Big[d_{ji}(t - \tau_{ji}) - d_{ij}(t)\Big] + \sum_j \Big[r_{ji}(t - \tau_{ji}) - r_{ij}(t)\Big]$ for all *i,t*, where τ_{ji} is defined as the travel time between station i and station j. In the definition of $v_i(t)$, we sum all vehicles arriving at station i, which is equivalent to summing over all vehicles which have departed for station i, τ_{ji} ago, and we subtract all vehicles which are departing from station i during current rebalancing interval. We define $v_i^{reb}(t) = \sum_j \Big[r_{ji}(t - \tau_{ji}) - r_{ij}(t)\Big]$ and assume that the travel time from station i to station j is known and given as τ_{ij}. τ_{ij} is constant over time for any i and j, $i \neq j$. Length of the rebalancing interval, Δt, is assumed to be equal or shorter to an average travel time on the network (if the rebalancing interval is too short or too long, then we may be performing too many or too little rebalancing trips, respectively).

The problem is formulated in Equation 5.1.

$$\begin{aligned}
\min \quad & N(t=0) & & \\
\text{s.t.} \quad & N(t) = \sum_i v_i(t) + m(t) & & \\
& N(t) = N(t + \Delta t) & & \forall t \\
& v_i(t) = v_i(t - \Delta t) + d_i^w(t) + v_i^{reb}(t) & & \forall i, t \\
& m(t) = m(t - \Delta t) - \sum_j \Big[d_i^w(t) - v_i^{reb}(t)\Big] & & \forall i, t \\
& v_i(t=0) = v_i(t = T_p) & & \forall i \\
& r_{ij}(t=0) = r_{ij}(t = T_p) & & \forall i, j \\
& m(t=0) = m(t = T_p) & & \\
& r_{ij}(t), v_i(t), m(t), N(t) \geq 0 & & \forall i, j, t
\end{aligned} \tag{5.1}$$

The objective of this problem (Eq. 5.1) is to minimize the total number of vehicles in the system while satisfying all trip requests. This objective is solved to answer the tactical decision of estimating the fleet size required to run the AMOD service. The first constraint tells us that total number of vehicles at each rebalancing interval is equal to vehicles owned by all stations plus the vehicles in transfer. The second constraint is set to ensure that the number of vehicles in the system is constant over time. The third constraint is the flow conservation at each station and the fourth constraint is the flow conservation for vehicles in transfer. The fifth and sixth constraints describe the periodicity, i.e, the number of vehicles in transfer at the end of the day is equal to the number of vehicles at the beginning of the next day. The last set of constraints are the non-negativity constraints.

The model solves for the number of vehicles required to service given booking requests and the rebalancing counts between different intervals. The number of vehicles at the beginning of the simulation and the rebalancing counts resulted from the model were fed to the simulator and later compared against the solutions of the online model.

5.4 Dynamic Rebalancing Problem

The dynamic, or real-time, model proposed in this dissertation is based on the fluid model first introduced in [Pavone et al., 2011, Spieser et al., 2014]. The model repeatedly solves the optimization formulated in [Azevedo et al., 2016] and finds the rebalancing counts to match the anticipated demand at all stations. We use the *rolling horizon* approach which provides us high quality solution to the dynamic rebalancing problem. Rolling horizon is a common practice for decisions in a dynamic stochastic settings [Sethi and Sorger, 1991]. In essence, this practice involves immediate decisions based on a (deterministic or stochastic) forecast for a certain number of periods in the future. The term *horizon* refers to the number of periods for which we perform forecasting. The number of periods for the forecast (the horizon) is of a great interest to the decision maker in order to make an optimal decision. In our case we study the time horizon of one period ahead as we perform rebalancing to satisfy customers in the next time horizon. This procedure repeats every period justifying the term *rolling* horizon.

In the dynamic model, the number of excess demand at station i, $\hat{d}_i$, is the number of customers that cannot be served using only the vehicles available at station i, i.e., $\hat{d}_i = v_i - d_i$. A negative $\hat{d}_i$ indicates that there are vehicles available to send. We assume that the cost of sending one vehicle from station i to station j is equivalent to the travel time between i and j and given as c_{ij}, which is constantly updated by SimMobility (online and time-varying travel time). Our decision variable r_{ij} is the number of empty (rebalancing) vehicles to send from station i to station j. Note that $\hat{d}_i(t)$ is the predicted number of excess demand for the next time period. The objective function minimizes the rebalancing effort.

$$
\begin{aligned}
\min \quad & \sum_{ij} c_{ij}(t) r_{ij}(t) & \\
\text{s.t.} \quad & \sum_{j} (r_{ji}(t) - r_{ij}(t)) \geq \hat{d}_i(t) & \forall i, j \\
& \sum_{j} r_{ij}(t) \leq v_i(t) & \forall i \\
& r_{ij}(t) \geq 0 & \forall i, j
\end{aligned}
\tag{5.2}
$$

The first constraint ensures that the number of rebalancing counts is greater or equal to the excess demand. The second constraint prevents us from sending more vehicles than we have available in at station i at the current time t. The third constraint is the non-negativity constraint.

The dynamic model is solved during operation of an AMOD system. In AMOD Controller the model is executed at every time Δt.

5.5 Results

In this section we present comparison of the performance analysis for different rebalancing strategies in an automated mobility on demand system. Comparison of the rebalancing is evaluated in terms of (i) number of customers served, and (ii) total distance traveled for the rebalancing.

5.5.1 Fleet Size Estimation

Fleet size estimation is a tactical decision problem. The optimal number of vehicles in a fleet varies with time depending on aspects such as market demand or the service level one wants to achieve. Therefore, the operator has to establish when to calculate the optimal number of vehicles that are required for a certain period of time. We estimate the optimal fleet size to operate an AMOD system based on the estimated demand for the service for a typical day. Our results for the fleet size are shown in Figure 5.3. The fleet size is evaluated as a function of customers' waiting time.

5.5.2 Comparison of Rebalancing Policies

We evaluate the following four policies:

1. No rebalancing policy, which serves as a lower bound on the performance.

2. Predictive dynamic rebalancing, where we optimize the rebalancing counts based on the demand predictions.

3. Static rebalancing which provides an upper bound on the performance where we perform rebalancing with the perfect information of the future state of the system.

Network and demand data To evaluate performance of the three rebalancing policies, we use a $56km^2$ network of the Central Business District in Singapore. The network consists of 1229 nodes, 14948 lanes and 313 traffic signals. The demand generation process for AMOD is based on integration of SimMobility MT with SimMobility ST. The simulated population in SimMobility MT was estimated based on the Household Interview Travel Survey for 2012 (for more details please refer to [Azevedo et al., 2016]). Given the output from SimMobility MT, we replaced all transport modes except of the Mass Rapid Transit (subway) and public buses with the AMOD service. If only a part of the trip was inside the CBD, then we cut the trip and simulated it from or to the border of the network. We run the simulation for the period of 3am-12pm. The total number of AMOD trips for this period was 363,859. Customers do not drop the booking requests and leave the system only when they finish their trips.

Facility location Locations of the stations are optimized based on the set covering model (detailed description in [Azevedo et al., 2016]). For this analysis, we assumed that the coverage radius of a station is 1000 meters, which is equivalent to 2-3 minute ride at the average speed of 30 km/h (based on the Land Transport Authority's data, the average speed for arterial roads during peak hours in Singapore is 28.9 km/h [Land Transport Statistics, 2015]). Solution of the model is 34 stations within the analyzed zone as presented in Figure 6.7.

Simulation setup In the dynamic model, we initialized stations with equal number of vehicles across the stations. We simulated fleet sizes of 10,000 to 40,000 vehicles. In the static model, the stations were initialized with the number of vehicles based on the optimization output for the fleet size. Note that, the static model solves for the fleet size and initial vehicle distribution, while the dynamic model is solved during the simulation time and only for the rebalancing counts. Both models are sensitive to the rebalancing interval. Therefore, the interval for the static model is 15 minutes, while for the online model is 1 hour. The reason of this difference is the following: too long interval in the offline model results in overestimation of the fleet size (as the travel time is discretized based on the interval size). On the contrary, too short interval in the dynamic model results in rebalancing during the period when there is a booking queue. All simulations are run in SimMobility ST, which simulates the individual decisions and the transportation network at the sub-second level (microscopic level).

Results Results are presented in Figure 5.3. Performance of different rebalancing policies

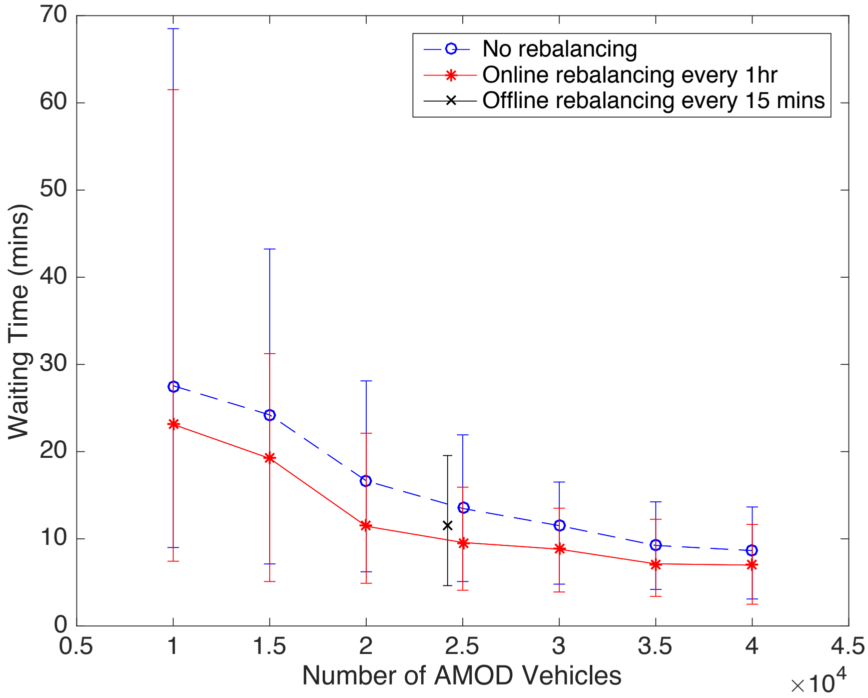


Figure 5.3: Mean waiting time for different rebalancing policies. Performance of different rebalancing policies expressed in terms of the customers' waiting time across different fleet sizes. The no-rebalancing policy (the blue line) performs quite poorly compared to the static and dynamic rebalancing methods. Both rebalancing methods help in reducing the fleet size, i.e., we need 35,000 vehicles without rebalancing for the average waiting time to be below 10 minutes and only 25,000 if we perform rebalancing. This finding translates to significant savings in the number of required parking lots.

is expressed in terms of the customers' waiting time across different fleet sizes. The results show a trend *the more vehicles, the shorter waiting time* valid across different models. Our objective is to run the service as efficient as possible, and therefore we aim to satisfy all customers at certain level of service with the smallest possible fleet size. For that reason we are looking at the waiting time of customers from the time they make the booking until they are picked-up by an AMOD vehicle. In our analysis, the no-rebalancing policy (the blue line in Figure 5.3) performs quite poorly compared to the static and dynamic rebalancing methods (also called offline and online, respectively). In general, rebalancing help in reducing the fleet size, i.e., we need 35,000 vehicles without rebalancing for the average waiting time to be below 10 minutes and only 25,000 if we perform rebalancing. This finding translates to significant savings in the number of required parking lots.

5.6 Summary

A crucial factor in the success of any mobility-on-demand system is its ability to meet the fluctuating demand for service. In order to reduce shortages, operators of shared systems are responsible for regular rebalancing, or transferring, vehicles from some locations to others. In this chapter we provided liner optimization models for solving static and dynamic rebalancing problems. Our simulation results indicate that the use of optimization methods for the rebalancing substantially improve the performance of AMOD systems.

Our future plan is to design congestion-aware rebalancing methods. It is also within our interest to extend the rebalancing model to account for the battery level before dispatching vehicle for the empty trip.

Chapter 6

Large Scale Implementation

In this chapter we provide simulation results for large-scale implementations of an AMOD system. We analyze two scenarios for the city of Singapore:

1. Introduction of an AMOD system as an alternative to private vehicles. In this case-study we introduce a new policy restricting private vehicle usage within the high-traffic Central Business District (CBD) area in the city of Singapore (Section 6.1).
2. Introduction of an AMOD system in the extended CBD in Singapore. In this case-study we assume that all trips within the analyzed zone are replaced by the AMOD trips (Section 6.2).

For each case-study we analyze four performance metrics:

1. Number of customers served by our service as a percentage of the total number of requests.
2. Customers' waiting time, which is measured from the booking time until the customer is picked-up,
3. Vehicle utilization rate, which is a ratio of the time when vehicles are servicing customers to the total time of the simulation. The time vehicles are busy includes time which is required to arrive and pick up the customer.
4. Amount of rebalancing, which is defined as the ratio of the distance vehicles drive for rebalancing and the total distance driven on the network.

6.1 Case Study of the Central Business District in Singapore

In this section, we describe case-study simulations designed to evaluate the effect of a new policy restricting private vehicle usage within the high-traffic Central Business District (CBD) area in Singapore (Fig. 6.1). The $14km^2$ network consists on 1229 nodes, 14948 lanes and 313 traffic signals. In the analyzed policy private vehicles are not allowed to enter the restricted area and the AMOD service is introduced as an alternative. In other words, only taxis, public transportation and AMOD vehicles were permitted to enter the analyzed area.

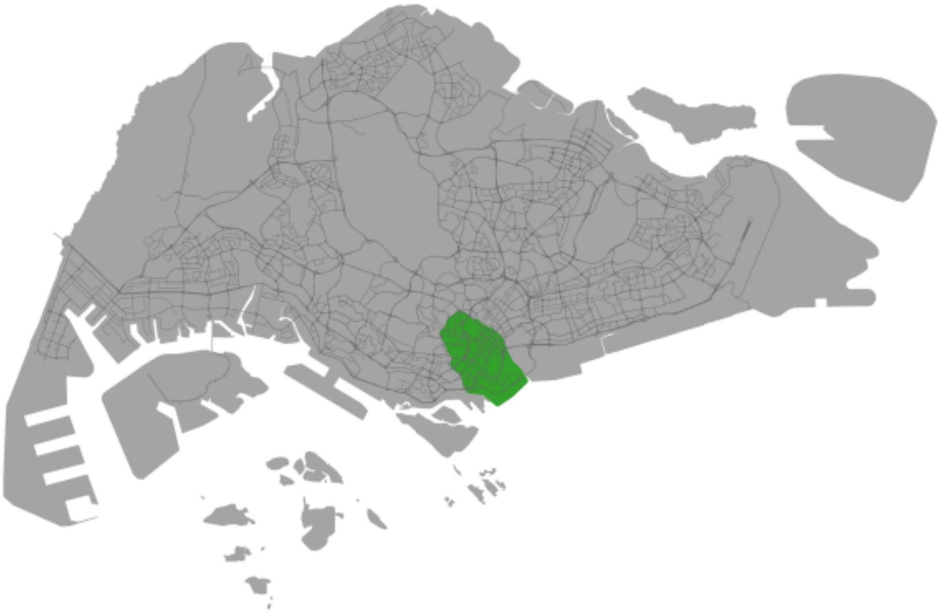


Figure 6.1: Case study of the Central Business District in Singapore. The analyzed zone, size of $14km^2$, is highlighted in green. In the analyzed policy private vehicles are not allowed to enter the restricted area and the AMOD service is introduced as an alternative.

6.1.1 Simulation Setup

We use SimMobility Mid-Term and Short-Term extended by AMOD Controller to perform analysis of an AMOD system. Details on modeling methodology within SimMobility Mid-Term and Short-Term are provided in Appendix D.

In the scenario analyzed in this section, private vehicles are not allowed to access a 14 km^2 restricted zone in the CBD and AMOD is introduced as an alternative mode of transport, which operates only within the zone. Yet, the access to this area was granted to the existing bus lines, Mass Rapid Transit (MRT) trains and taxis. Analysis of this policy is of the great interest to the local governments in Singapore as the land is limited and over 12% of the total area of Singapore is occupied by the road infrastructure. Therefore, the policy-makers are looking for alternatives to owning vehicles, which is a part the *car-lite* vision of Singapore. In our case study we provide AMOD system with single-class autonomous electric compact vehicles. We allow single-rider assignment, i.e., one vehicle is assigned to one booking. The simulations are run for a 2-hour period during evening peak (5:00pm to 7:00pm).

Demand generation The demand generation process of AMOD is based on integration of SimMobility MT simulator with SimMobility ST simulator. SimMobility MT simulates agents mobility decisions that includes their activity and travel patterns along with mode, time-of-day and route choices. Description on midterm simulator can be found in [Lu et al., 2015]. For this study the SimMobility MT model assumes all private vehicle trips as a combined modal trip (i.e., Private vehicle + AMOD) if part of the trip is inside CBD. The mode choice model in SimMobility MT is modified by making it sensitive to AMOD waiting time and additional cost terms, which actually fed back by SimMobility ST in an iterative framework to bring consistency. Further parking prices for private vehicle are reduced as now they have been parked outside the CBD region. The cost of the AMOD service was assumed as 40% less than the regular taxi service in Singapore, which is due to lower employment cost. This assumption results in an average cost of about SGD 3 per trip within the analyzed area, which is three times more expensive than the trains and four times more expensive than buses. The other modes were assumed to remain unchanged, i.e., buses and trains kept their frequencies, fares and capacities and the taxi fleet and cost remained the same. using the year 2012 configuration as reference. We performed two different simulation time setups:

1. The simulations are run for the period of two hours, 5pm–7pm, during the evening rush hours with the total number of AMOD trips equal 28,525 (referred as Case 1, detailed description in [Marczuk et al., 2015]).

2. The simulations are run for the period of twelve hours, 3am–3pm, catching the morning rush hours with the total number of AMOD trips equal 90,144 (referred as Case 2, detailed describtion in [Azevedo et al., 2016]).

Evening peak hours within the Central Business District (case 1) are very representative due to the fact that most of the taxi bookings (calls) after office hours are from CBD, so we are simulating the highest demand. In the second case, we simulate a longer period of time (12 hours) which overlaps with morning peak hours. We start the simulation earlier to mitigate influence of initialization, e.g., initial location of vehicles, on the system performance. We select morning peak hours because—alongside many performance metrics—we evaluate mode choice. In the mode choice analysis, activities related to work and school generate the most rigid constraints for the arrival time at the destination. Therefore, we were interested to look at the performance of the system, where agents are constraint to the transportation modes they can use while performing their usual work-related trips. The transportation system has changed (restricted zone with AMOD service), but the activity schedules remain unchanged.

For more details about validation of modal shifts within SimMobility, the reader is directed to Appendix E.1.

Facility location models In the station-based model, 4 different sets of facility locations were analyzed. The first set consisted of 10 nodes, which were selected based on the highest frequency of originating trips (*high-demand* nodes). The remaining three sets consisted of the top 20, 30 and 40 high-demand nodes, respectively. There was no capacity constraint on the facilities, i.e., the facility could hold as many cars as required. In the free-floating model, *initial stations* were assumed in the same manner as for the station-based model; however, in the free-floating model, cars were not required to return to these stations. Twelve different fleet sizes were simulated, i.e., from 2000 to 7500 AMOD vehicles in the system. At the beginning of the simulation, vehicles were uniformly distributed over the facilities.

Assignment, rebalancing and post-service routing In this case-study we assign single booking to single vehicle (no ride-sharing is allowed). The assignment is based on the bipartite matching algorithm which is introduced in 4.3.2. As presented in Chapter 5 the best performance in terms of the shortest average waiting time for the Central Business District case-study is to perform dynamic rebalancing every 1-hour. Therefore, in this analysis we show the results for the dynamic rebalancing every 1-hour. Note, that in this analysis we do not implement the static rebalancing model. Without static rebalancing we have no information on the optimal vehicles allocation, so all vehicles were initialized evenly across the stations. In this study, we evaluated two post-service routing alternatives, that is, how the autonomous vehicles behaved after dropping-off passengers:

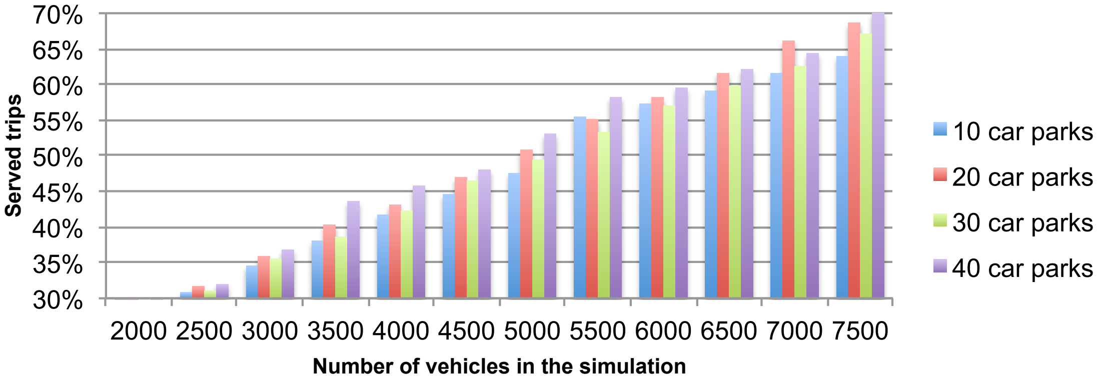
1. In *station-based model*, after servicing a trip, AMOD vehicles always drove back to the nearest station and waited for new requests (and re-charge if necessary).
2. In *free-floating model*, AMOD vehicles self-parked at drop-off locations, where they waited for new requests. It is assumed that all drop-off locations contained parking facilities where the vehicles could wait and optionally recharge.

Finally, both models assume that customers make immediate reservations (no advance booking is allowed) and that AMOD vehicles pick-up and drop-off passengers at any node in the road network.

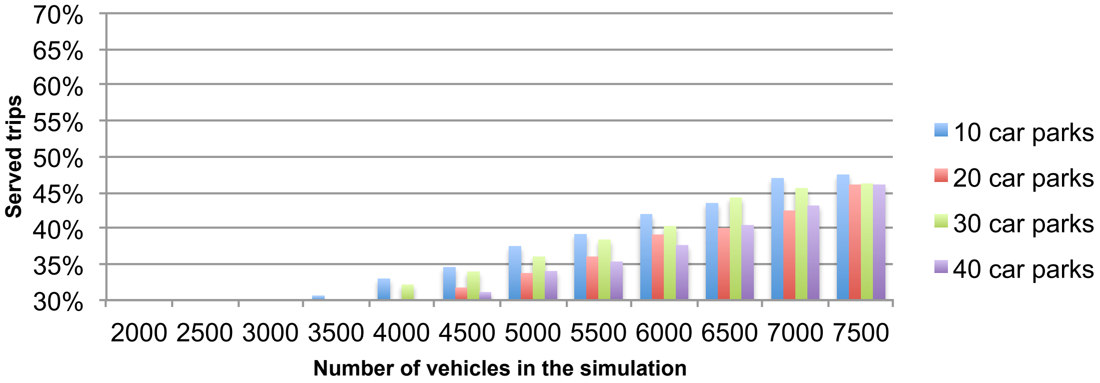
6.1.2 Results

For *non-autonomous* MOD systems, the free-floating scheme is arguably more preferable for the consumer since it alleviates him/her from the costs associated with returning the vehicle. For autonomous systems, vehicles can self-return to station, but this return leg constitutes an empty trip (which may increase road congestion and fuel-use). Furthermore, if the station is further away from the next requested service, the vehicle would be making an unnecessary trip. On the other hand, in the free-floating model, vehicles can become severely unbalanced leading to longer waiting times for consumers. The station-based model requires use of car-parks, which contributes to increased land-use. Our study seeks to evaluate the effects of both models in the densely population island nation of Singapore during a peak travel period.

Number of customers served Figure 6.2 shows the percentage of customers served versus the AMOD fleet size in the system under (a) free-floating and (b) station-based models. Trip served is referred to a trip of a customer who (i) successfully booked a ride, (ii) was picked up by an AMOD vehicle, and (iii) was successfully dropped off at the destination. Note that not all the generated trips were served because a proportion of the passengers had not yet arrived at the destination by the end of the simulation. In both models (free-floating and station-based), increasing the vehicle fleet size resulted in a linear increase in the number of passengers served, with gradient coefficients of 0.037 for the free-floating model and 0.022 for the station-based model. In other words, every additional 100 cars provisioned increased the average demand served by 3.7 percent (1055 people-trips) in the free floating scheme. For the station-based model, this increase was smaller at 2.2 percent (627.55 people-trips).



(a) Free floating model.

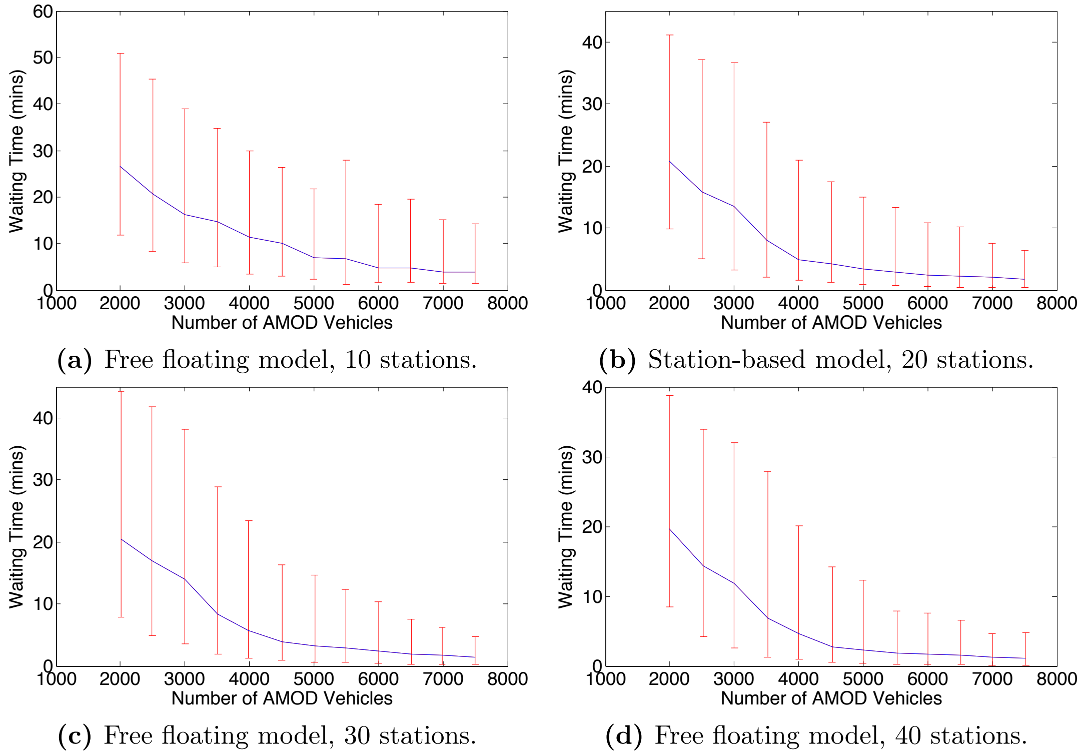


(b) Station-based model.

Figure 6.2: Percentage of customers served versus the AMOD fleet size in the system for the free-floating model (left) and the station-based model (right). Trip served is referred to a trip of a customer who (i) successfully booked a ride, (ii) was picked up by an AMOD vehicle, and (iii) was successfully dropped off at the destination. Using the free-floating model we could serve as much as 70% of the total demand (with 7000 vehicles and more), while using station-based model we could only serve up to 48% of the demand. Note that not all trips have arrived at the destination before the simulation end time.

The free-floating model was able to serve as much as 70% of the total demand, which is significantly more than the station-based model (48% of the requested trips). The low service rate in station-based model was likely caused by heavier traffic due to empty vehicle rides. This is consistent with the average travel time (which can be seen as a proxy metric for road congestion) of both models. The average travel time in the station-based model was on average longer than in the free-floating model, e.g., with 40 stations and 7500 vehicles the average travel time for the station-based model was 14.17 minutes, which accounts for about 30% increase when compared against the station-based model (10.59 minutes).

Customer waiting time The waiting time analyzed here is defined as the time difference between the trip request time and the pick-up time. Figure 6.3 shows the median customer waiting times (with upper and lower quartiles) versus the number of AMOD vehicles under the free-floating model. As expected, increasing the AMOD fleet size resulted in a fall in waiting times, since more vehicles were available to service the requested trips. For example, with 20 initial stations, the median waiting time decreased from 20.74 to 1.80 minutes as the fleet size grew from 2000 to 7500 (similarly, the variance in the waiting times decreased from 31.38 to 6.09). Unlike the effect on total demand served, this waiting time change is non-linear and shows that the rate of improvement decreases with increasing fleet size and appears minimal beyond 6000 vehicles. The initial distribution of vehicles also influenced the performance of the system. Based on our results, increasing the number of initial stations



(a) Free floating model, 10 stations.

(b) Station-based model, 20 stations.

(c) Free floating model, 30 stations.

(d) Free floating model, 40 stations.

Figure 6.3: Customer waiting time in the free-floating model.

decreased passenger waiting times. The biggest difference is between 10 and 20 stations, where we observed an average improvement of approximately 4 minutes across fleet sizes. However, further increases in the number of stations resulted in the decreases in waiting times below 1.5 minute.

Vehicle utilization rate Results for the vehicle utilization rate are presented in Figure 6.4. As expected, we observe a trend that the larger the fleet size, the smaller the utilization rate.

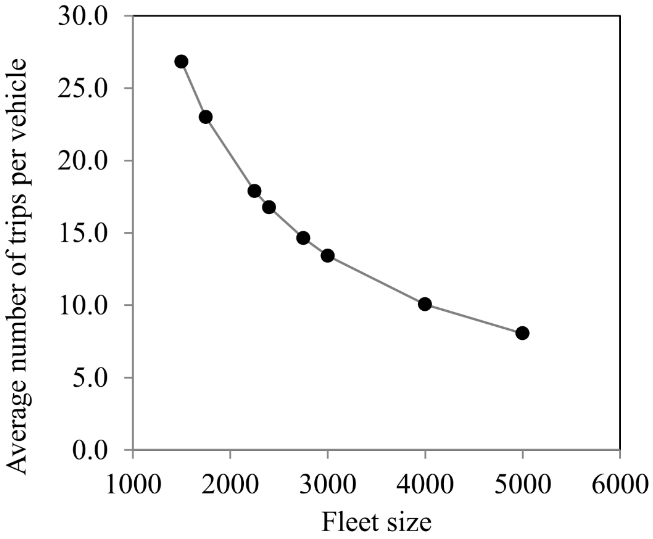


Figure 6.4: Vehicle utilization rate for the CBD case-study (12-hour simulation for the period from 3am–3pm. We observe a trend that the larger the fleet size, the smaller the utilization rate.

For the fleet size which is not sufficient to serve the entire demand (below 2'500 vehicles), the vehicle utilization rate is equal to over 20 trips per vehicle and it constantly drops as we increase number of vehicles, reaching less than 10 trips per vehicle with the fleet size of 4'000 vehicles and more. For the fleet size of 2500 each vehicle serves 16.7 trips on average. The explanation of this trend may be the following: 1'500 and 2'000 vehicles serves smaller number of customers than a fleet of 2'500 vehicles and above. However, the overall vehicle utilization rate is higher than for larger fleets, which are providing service to about 18% more customer.

6.2 Case Study of the extended CBD in Singapore

In this section, we describe case-study simulations designed to evaluate the effect of introducing an AMOD system in the high-traffic Extended Central Business District (ECBD) in Singapore (Fig. 6.5). The analyzed $56km^2$ zone consists of 1178 nodes. The zone is highlighted in pink. It is important to notice that the analyzed area is a high-traffic area,

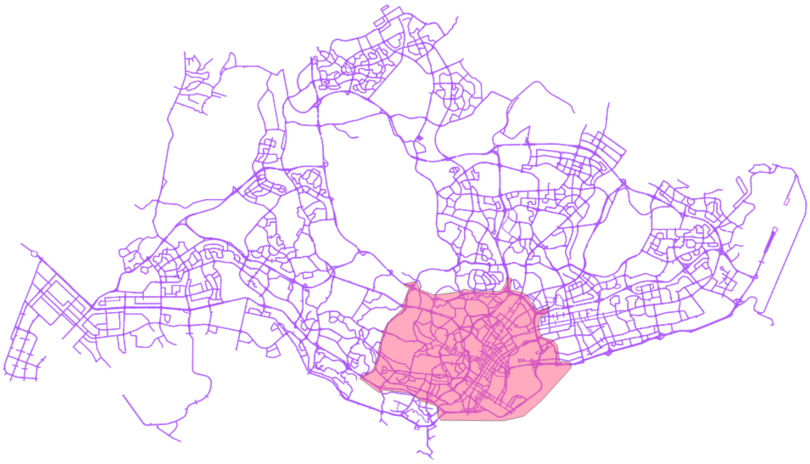


Figure 6.5: Case study of the Extended Central Business District in Singapore. The analyzed zone, size of $56km^2$, is highlighted in pink. The network consists of 1178 nodes.

which is a significant trip attractor generating over 40% of all trips in Singapore [Marczuk et al., 2016]. For the assessment of the AMOD system and its impact on traffic in Singapore, we run the ECBD network in SimMobility ST. To help the reader to visualize our approach, a screen-shot of the simulation in SimMobility is presented in Figure 6.6. The illustration shows a detailed network representation, which includes lanes and traffic signals. As show in the figure, the AMOD service is simulated alongside regular buses, taxis and private vehicles.

6.2.1 Simulation Setup

To analyze introduction of an AMOD service in the extended CBD region in Singapore, we use SimMobility as our simulator. Settings of the AMOD Controller are described in the following paragraphs.

Facility location The facility location for the Extended Central Business District in Singapore is optimized based on the maximum set covering formulation (Section 3.2.4). The solution consists of 34 stations which are distributed as presented in Figure 6.7. In this analysis, we assumed that the coverage radius of a station is 1000 meters, which is equivalent to 2-3 minute ride at an average speed of 30 km/h (based on the Land Transport Authority's data, the average speed for arterial roads during peak hours in Singapore is 28.9 km/h [Marczuk et al., 2016]).

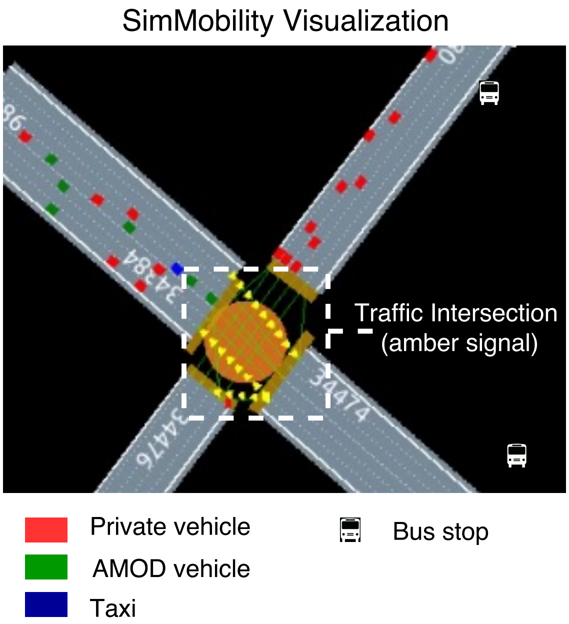


Figure 6.6: Screen-shot of the SimmMobility ST simulator, which shows detailed network representation including lanes and traffic signals. As show in the figure, the AMOD service is simulated alongside regular buses, taxis and private vehicles. AMOD vehicles are represented as green boxes.

Assignment, rebalancing and post-service routing In this case-study we analyze two assignment techniques:

1. Single vehicle to single booking assignment (no ride-sharing is allowed). The assignment is based on the bipartite matching algorithm which is introduced in 4.3.2.

2. Single vehicle to multiple rides assignment (ride-sharing is allowed). This assignment method is detailed in Section 4.4.

In this case-study we perform both, static and dynamic rebalancing. Static rebalancing provides us with the results for an optimal vehicle distribution at the beginning of the day (beginning of the operation) and rebalancing counts based on the estimated demand. Our benchmark is the simulation with no rebalancing. Both rebalancing models are sensitive to the rebalancing interval. Based on our simulation results, we set the interval for the static model at 15 minutes, and for the dynamic model at 1 hour. The reason of this difference is the following: too long interval in the static model may result in overestimation of the fleet size (as the travel time is discretized based on the interval size), while too short interval in the dynamic model may result in rebalancing during the peak period (especially if there is a booking queue).

We evaluate only one post-service routing alternative, that is, free-floating model. We did not

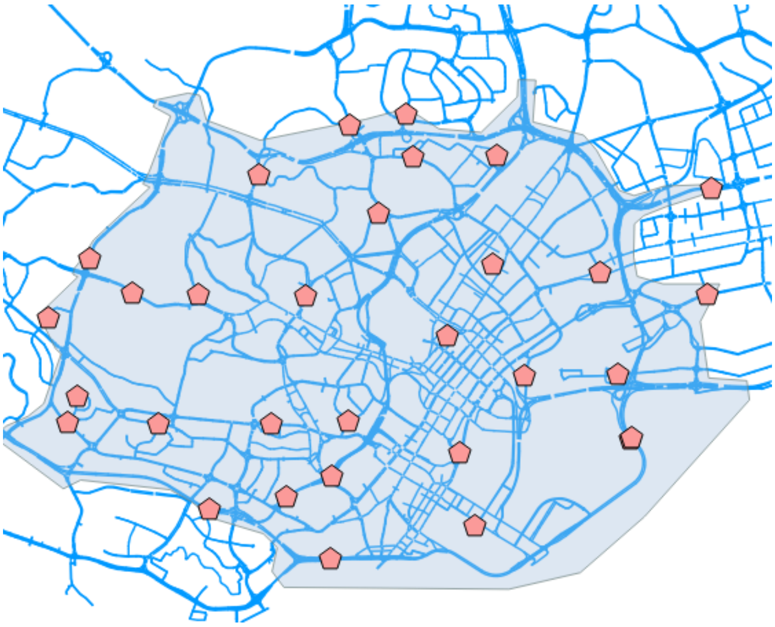


Figure 6.7: Facility location map for the Extended Central Business District in Singapore

consider routing vehicles back to the stations after finishing trips as based on our previous analysis (Section 6.1) routing back to the stations increases waiting time.

Demand generation The demand generation process for AMOD is based on integration of SimMobility MT with SimMobility ST. The simulated population in SimMobility MT was estimated based on the Household Interview Travel Survey for 2012 (for more details please refer to [Marczuk et al., 2016]). Given the output from SimMobility MT, we assign all trips made on private vehicles (cars, motorbikes), car-sharing and taxis to the AMOD service. If only a part of the trip was inside the ECBD region, then we cut the trip and simulate it from or to the border of the network. We run the simulation for the period of 3am-12pm focusing on the morning peak hours. We start simulation at 3am to minimize the effect of initial conditions for Controller's performance during peak hours (6:30-10am). Simulating morning peak hours is within our great interest because usually people have more time constraints during their trips to offices (or schools). On the evening travel time and waiting time is less crucial during their decision making process. We therefore analyze how the system performance during morning peak hours changes as we introduce AMOD mode.

The total number of AMOD trips for the simulation period is 363,859, which is equivalent to about 40,000 trips per hour on average.

AMOD demand and supply assumptions Vehicles change their roles dynamically depending on the demand, i.e., the same vehicle can serve single and multiple-riders. Additionally, vehicles do not return to stations after finishing their trips. However, they may be engaged in rebalancing, which is always sends vehicles back to the stations. Customers are sensitive to time and they drop their booking requests after queuing for more than 10 minutes. If the customer leaves the system, the operator is penalized for the dropped call.

6.2.2 Results

In this section we present results for the case study of implementing AMOD at the Extended Central Business District in Singapore.

Stations location Results for the stations location are presented in Figure 6.7. We select 34 facilities optimized based on the set covering formulation. Detailed evaluation of the facility location problem for the ECBD network is presented in Section 3.3.

Assignment There are two methods for assigning single vehicle to single booking: greedy and bipartite approach. Figure 6.8 presents comparison, in terms of waiting time, of greedy and bipartite assignment. Our results for the single-rider assignment suggest that bipartite assignment technique outperforms greedy assignment method. Regardless of the fleet size, greedy assignment results in 1-1.5 minute longer average waiting times for customers. This, however, comes at the computational cost as it takes $O(n^3)$, where n is the size of bipartite set, to solve bipartite problem and $O(n)$ to solve the greedy implementation. Our goal is to keep the average waiting time below 5 minutes, which can be achieved with 6'000 vehicles

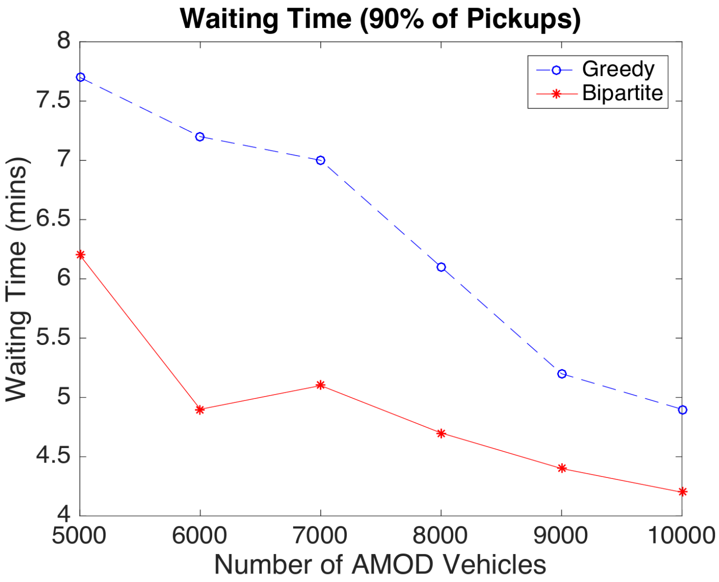


Figure 6.8: Waiting time for greedy and bipartite assignment. Regardless of the fleet size, the greedy assignment results in 1-1.5 minute longer average waiting times for customers.

and bipartite model. If we assign vehicles based on greedy model, we have to increase the fleet size to 10'000 vehicles, or 67%. Without the increased fleet size, waiting time increases 2.1 minutes, or 41%.

In the multiple-riders assignment, more than one customer is assigned to one vehicle. In our analysis, we allow maximum of two bookings to share the ride. We compare how waiting time changes as different percentage of customers decides to share their ride. Results for two fleet sizes, 5000 and 10'000, are presented in Figure 6.9. Note, that in this analysis we allow customers to wait in the queue for an unlimited time. We evaluate three cases of ride-sharing:

1. All customers are willing to share the ride,
2. 50% of all trips are shared (every second customer is willing to share the ride),
3. Non of the customers is willing to share the ride (all trips are simulated as single-rider trips).

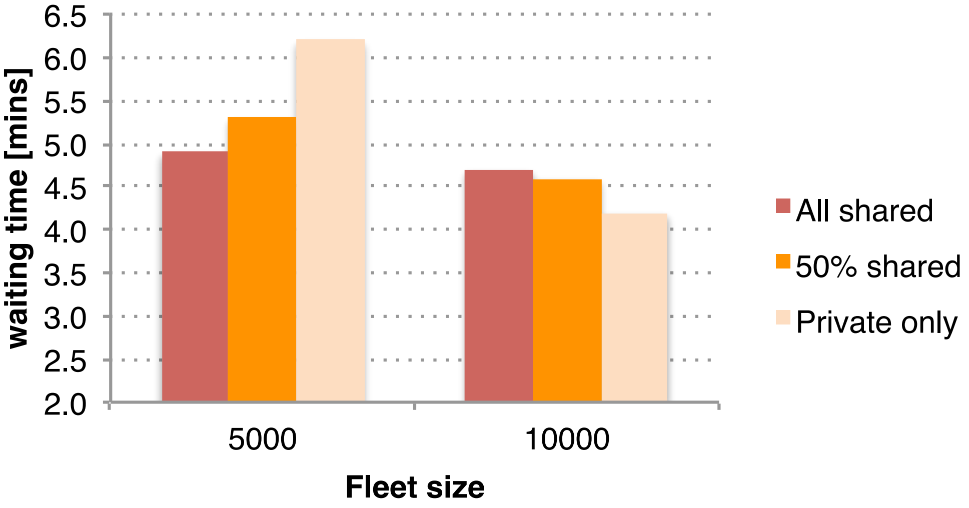


Figure 6.9: Results for assignment with ride-sharing. Note that *private* is referred to *single-rider trips*—AMOD trips with only one customer on board. Assignment to private rides are computed as described in Chapter 4. *50% shared* rides means that every second customer shares the ride, while in *all shared* everyone is willing to share the ride. Our results suggest that ride-sharing reduces waiting time of customers and reduces the required fleet size to run the service.

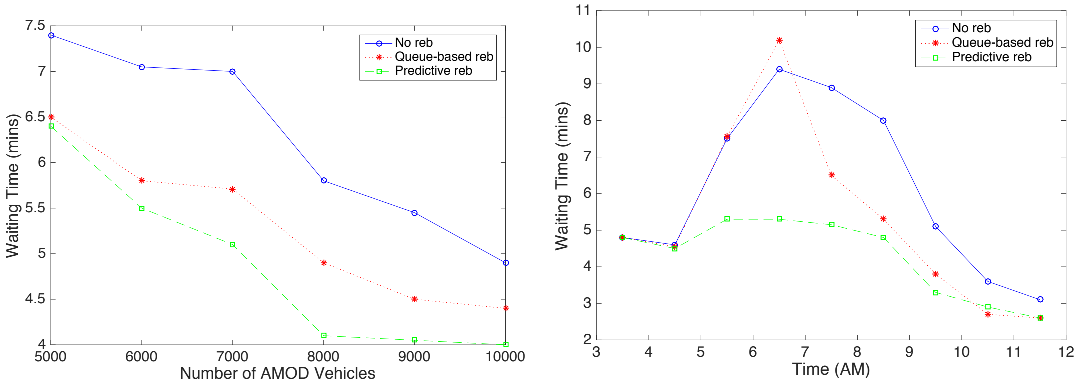
Our results show waiting time as a function of the percentage of shared trips and the fleet size. 10'000 vehicles is the fleet size which is sufficient to serve the entire demand of single-rider requests in the shortest waiting time. 5'000 vehicles is the fleet size which is not sufficient to serve everyone in the average waiting time below 5 minutes. If everyone is willing to share the ride, the fleet size of 5'000 vehicles meets the demand in a similar average waiting time as a doubled-size fleet for single-rider assignment. This leads us to conclusion that ride-sharing reduces the number of vehicles required to run an on-demand service significantly. If only half of our customers share the ride, the average waiting time is 5.3 minutes (fleet of 5'000 vehicles), which is 15% better than for single-rider trips. Additionally, with we increase in the fleet size to 10'000 the average waiting time drops further to 4.6 minutes. This result is 8% worse than if non of the customers share rides and our fleet size is 10'000 vehicles. This can be explained by the fact that 10'000 vehicles is sufficient to serve everyone and when we introduce ride-sharing we increase waiting time. We conclude that in the situation where the supply is limited and not sufficient to serve everyone, ride-sharing is a crucial factor to ease waiting queue. However, if the supply exceeds demand, customers are served faster without ride-sharing.

Rebalancing Here, we analyze how different rebalancing policies influence performance of the system. Our evaluation of the rebalancing policies consists on two parts:

1. Comparison of static, dynamic and no-rebalancing polices.
2. Comparison of different approaches to model dynamic rebalancing.

First part of the analysis is presented in Section 5.5. Results for the second part (different dynamic rebalancing approaches) are described in this section and presented in Figure 6.10. Based on our previous analysis [Marczuk, 2015], rebalancing every 1 hour results in the shortest waiting times. We tested rebalancing every 0.25, 0.5, 1, 2, and 3 hours in the dynamic settings and our conclusions are the following. (i) If the rebalancing period is shorter than 1 hour, the improvement in terms of the average waiting time is not significant (less than 10 seconds). This may be due to the fact that, in the dynamic settings, too short rebalancing interval leads to underestimation of peak demand. (ii) Rebalancing period which is too long, may lead to overshooting the number of required rebalancing trips. Therefore, in Figure 6.10 we present results for rebalacning performed every 1 hour.

The two graphs in Figure 6.10 summarize performance of different rebalancing methods introduced in Section 5. Figure 6.10 (a) shows the average customer waiting time for different rebalancing policies as a function of the fleet size. Figure 6.10 (b) shows the average customer waiting time for different rebalancing policies as a function of the time of the day. For both graphs three dynamic rebalancing policies are considered: (i) No-rebalancing policy (blue line), which is a baseline scenario. (ii) Predictive rebalancing, where we perform rebalancing based on the demand prediction for the next rebalancing interval (red line). (iii) Rebalancing based on the current queue at each station (green line). For each policy, waiting time decreases with an increase in the fleet size. We conclude that rebalancing reduces required fleet size to serve the demand, e.g., for the average waiting time to fall below 5 minutes, we need a fleet of 10,000 vehicles operated without rebalancing and as little as 7,000 with predictive rebalancing. This finding translates to 30% savings in the



(a) Waiting time as a function of the fleet size.

(b) Waiting time as a function of the time of a day for the fleet of 7,000 vehicles.

Figure 6.10: Waiting time for different rebalancing policies in the ECBD scenario. Waiting time as a function of the fleet size (left) and waiting time as a function of time time of a day (right). Rebalancing reduces required fleet size, however requires careful tunings to better meet peak demand.

required fleet size, while maintaining the same level of service. We observe that for the fleet of 7,000 vehicles, predictive rebalancing reduces average waiting time by 30 seconds as compared to the rebalancing based on the current queue. Furthermore, in Figure 6.10(b) the waiting time from no-rebalancing and queue-based rebalancing during peak hours is twice longer than for the predictive rebalancing. This may be justified by the fact that predictive rebalancing moves vehicles in advance, before the cusomters' queue accumulates, while rebalancing based on the current queue, moves vehicles only when the demand arrives in the queue. For the reference, the rebalancing counts for the fleet size of 7,000 vehicles are presented in Figure 6.11. Note that, presented rebalancing counts are for 1-hour rebalancing interval. We observe that predictive rebalancing dispatches more empty trips before the peak period, while the queue-based model sends rebalancing vehicles when the demand is already high (7am–9am). Therefore, the predictive model is able to better represent the demand, which leads to a more accurate response to the system dynamics, i.e., vehicles are prepositioned before the demand appears and as a result they can take more trips within a shorter period of time (compare Figure 6.10).

We also observe how the travel time (a measure of congestion) changes as we increase the fleet size. Results for the travel time are presented in Figure **??**. We conclude that the mean travel time is not affected significantly by the number of vehicles in the simulation. However, standard deviation increases (Figure 6.12 (b)) as we increase the fleet size. This finding suggests that due to popularity of some origins)or destinations) some of the links are congested.

Our ultimate goal was to compare waiting and travel times for customers commuting on

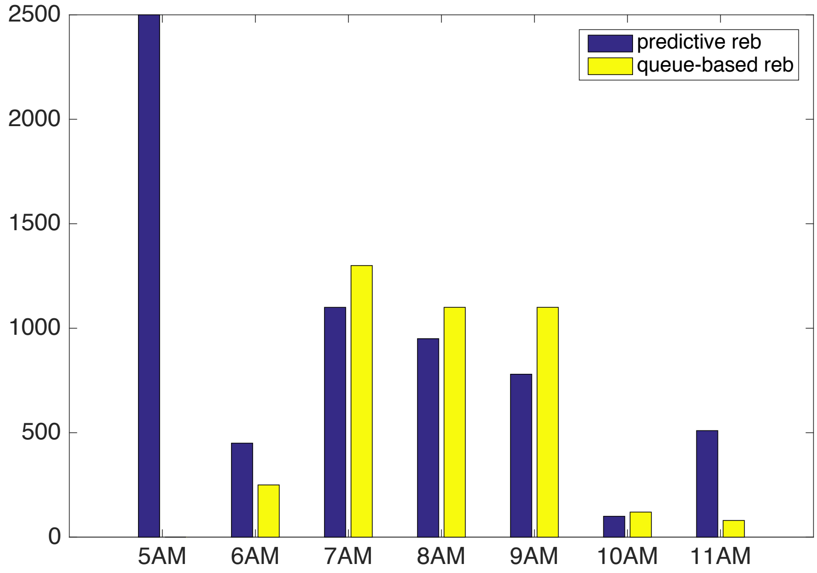
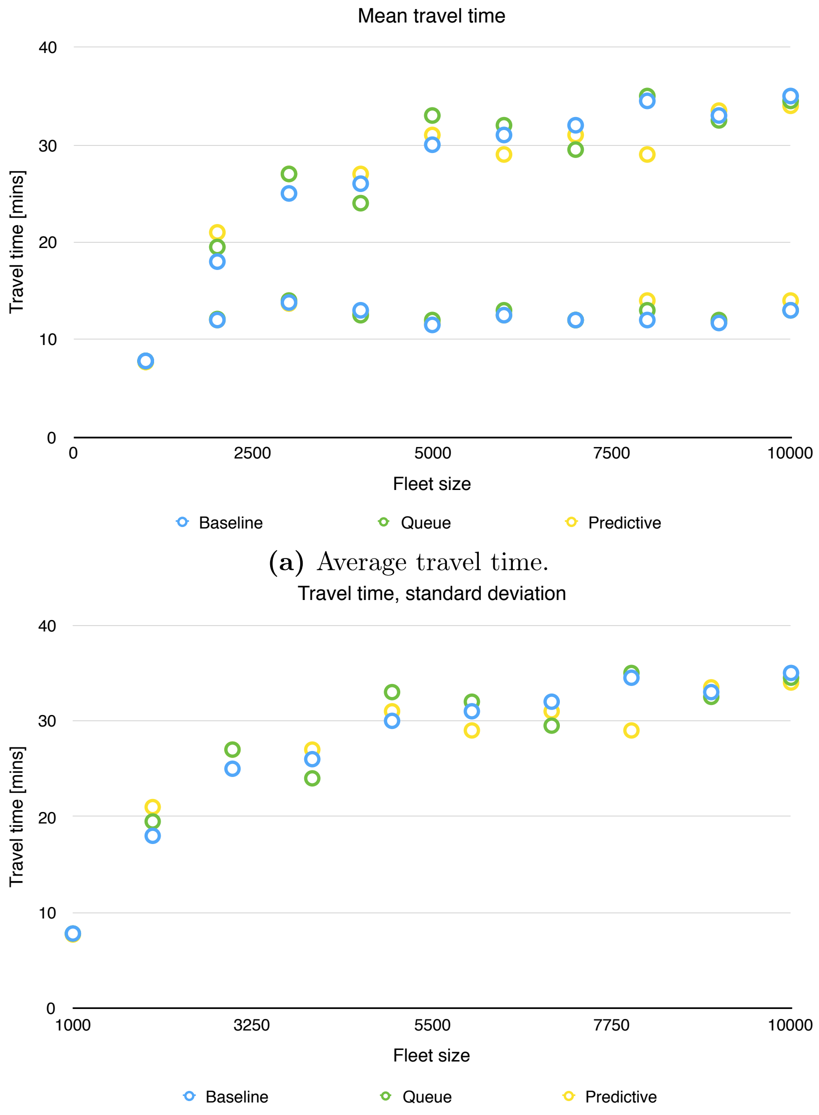


Figure 6.11: Rebalancing counts for different rebalancing policies. Predictive rebalancing moves vehicles before the peak period, so that the system is ready to take more trips within a shorter period of time (compare Figure 6.10). Note that there is no rebalancing during period 3–4AM, which is due to the optimized initial vehicle distribution.

AMOD system against the current transportation modes. Unfortunately, due to SimMobility still being under development, we could not complete this task and it is left as a future work.

Number of vehicles on the roads In the CBD area we simulate over 14,000 requests per hour, while in the ECBD region over 30,000 requests per hour. For the given demand, sufficient fleet size consists of 4'000 and 7,000 vehicles, respectively. This translates to 3.5 trips per hour per vehicle in the 14 km^2 zone and 4.3 trips per vehicle per hour in the 56 km^2 zone. Note, that these numbers are evaluated for the peak period. In both scenarios we replaced private vehicle trips with the AMOD trips. For the private vehicle trips, every trip is equivalent to one vehicle on the road. Therefore, by introducing AMOD service, we reduce the number of vehicles on roads by 3.5 and 4.3 times for CBD and ECBD, respectively. Furthermore, we increase the distance traveled by the vehicles due to the rebalancing trips. Our analysis show that by introducing an AMOD service we do not increase average travel time on the network, e.g., average travel time remains bounded within 13 minutes for the fleet size up to 10,000 vehicles in the ECBD scenario. We conclude that a carefully designed AMOD system does not increase congestion level on the network as it requires smaller number of vehicles compared to the system with privately owned vehicles. However, we observe that as we increase the fleet size, the congestion on some links increases, e.g., for the fleet sizes of 5,000 and 10,000 vehicles the standard deviation for travel time increases from 10 minutes to over 25 minutes, but the average travel time does not change significantly.



(a) Average travel time.

(b) Standard deviation of travel time.

Figure 6.12: Travel time for different rebalancing policies, the ECBD scenario. Mean travel time as a function of the fleet size (left) and standard deviation of travel time (right). We conclude that the mean travel time is not affected significantly by the fleet size being simulated. However, standard deviation increases as we increase the fleet size which suggest that some of the links are congested.

Run-time Performance Run-time performance of AMOD Controller attached to Sim-Mobility Short-Term is presented in Figure 6.13. We conclude that run-time performance of AMOD Controller is a function of the number of vehicles in the simulation and number of booking requests. As fleet size increases, the run-time is longer until it reaches a *stable state*. For the fleet size equal of larger than required to satisfy all booking requests, e.g., in Figure 6.13 above 6,000, the run-time is not affected by the fleet size. We claim that the remaining vehicles, e.g., vehicles parked at stations, do not contribute to performance of the

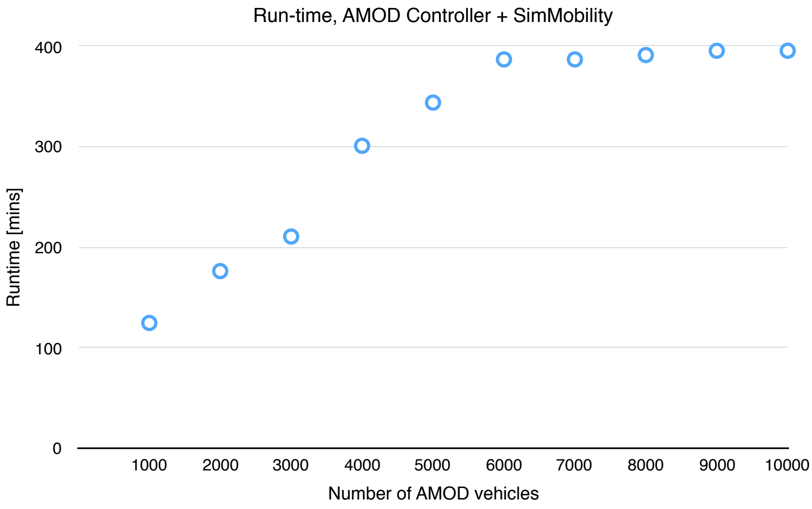


Figure 6.13: Run-Time of AMOD Controller attached to SimMobility Short-Term.

run-time.

6.3 Summary

In this chapter we provided simulation results for large-scale implementations of an AMOD system for the city of Singapore. We analyzed two case study scenarios: (i) Introduction of AMOD system as an alternative to private vehicles in the high-traffic Central Business District in the city of Singapore. (ii) Introduction of an AMOD system as a new mobility service in the extended CBD region in Singapore. We provided results for the facility location, assignment with and without ride-sharing and different rebalancing policies. We noticed that ride-sharing reduces the fleet size as well as [for insufficient fleets of vehicles] the average waiting times. We also proved that rebalancing reduces required fleet size, however requires careful tunings to better meet peak period demand. We show that the predictive model for dynamic rebalancing is able to significantly better represent the demand than the queue-based model. The predictive model responses more accurately to the system dynamics, i.e., vehicles are prepositioned before the demand appears and as a result they can perform more trips within a shorter period of time.

Our future plans are to evaluate more scenarios for the implementation of AMOD systems. We are planning to extend the analysis for the rebalancing and assignment through demand management with dynamic pricing. We are currently working on a real-vehicle implementation of the controller with a fleet of three multi-class autonomous vehicles. It is within our interest to extend the controller to handle booking systems of a large-scale real deployments.

Chapter 7

Conclusion and Future Directions

7.1 Conclusion

As planners, we need to have a holistic picture of how the overall system works—from the moment a person gets out of the house to the destination. For the convenience of the commuters, we propose an automated mobility on demand service that matches up vehicles and riders for a ride. The system is fast in response and requires minimal effort from the customers.

In this dissertation we look at the feasibility of assembling a fleet of autonomous vehicles for a mobility on demand service in order to ease congestion, increase safety, and reduce environmental impacts. We consider a new mode of urban transportation, which we refer as an Autonomous Mobility on Demand (AMOD) system. Specifically, we focus on the issues related to the fleet management of an autonomous mobility on demand system.

The main contribution of this dissertation is development of framework and related algorithms to demonstrate the role of autonomy in Mobility on Demand systems and its impact in terms of feasibility and efficiency through modeling, simulation, algorithm development and experimental demonstration. The proposed methodologies are applicable to a large-scale systems in different simulations' setup. We present our methodology which consists on three primary levels differentiated by the timeframe: planning, tactical and operational level. In each pillar we consider different decisions related to design and operation of an AMOD system:

(i) At the planning level we decide on strategic decisions, such as location and size of operating area and number and location of stations (zones).

(ii) At the tactical level we decide on long-term operation decisions such as number of vehicles which are required to run the system (fleet size), *home* location of the vehicles at the beginning of the day and static rebalancing.

(iii) At the operational level we decide on short-term operational decisions such as vehicle to customer assignment, rebalancing policies, and dynamic pricing.

The key part of this undertaking relies on the tests via simulation studies for the city of Singapore. Simulations are performed with the use of SimMobility—an agent-based simulation platform.

Contribution of each chapter is the following. In Chapter 1 we survey the related research models in academic literature. In Chapter 2, we introduce AMOD Controller, which is designed to manage a fleet of autonomous vehicles. We describe our methodology and systematically outline the system architecture and functional organization of AMOD Controller. In Chaper 3, we focus on the most important strategic aspect of management of AMOD systems—selecting the number and location of new facilities. In this chapter we formulate five optimization-based approaches for finding the number and location of the AMOD stations. To assess the merits of our methods we present a simulation study based on private vehicle travel demand for the Central Business District in Singapore. We conclude that the set covering formulation gives us the best, in terms of the customers' waiting and travel times, results. In Chapter 4, we consider a real-time vehicle to customer assignment methods. We formulate two types of assignment: (i) for customers who are not willing to share the ride (called single-riders), and (ii) for customers who are willing to share the ride (multiple-riders). Our simulation results indicate that the use of optimization methods instead of simple greedy approaches substantially improves performance of AMOD systems. In Chaper 5, we describe a core-component of this dissertation, which is empty vehicle rebalancing. Rebalancing aims in finding optimal control policies for realigning demand and supply by minimizing the total cost of rebalancing. We propose static and dynamic formulation of the problem. Static rebalancing solves for the total number of vehicles required to operate the system and the rebalancing pattern between different locations for the course of the entire day. Dynamic rebalancing solves rebalancing problem at *real-time*. We integrate our optimization techniques within a simulation environment and test the algorithms for the case studies placed in Singapore. We demonstrate the value of optimization-based rebalancing policies in the operation of the AMOD system. Finally, Chapter 6 provides the analysis of a large-scale implementation of the AMOD solution for the city of Singapore.

Our results suggest that introducing AMOD service has a great potential to serve for future urban mobility. We believe that owning a conventional private car will take a backseat when the choices for being in a shared self-driving vehicle become available.

7.2 Limitation and Future Directions

To the best of our knowledge, this dissertation is the first work to perform a complete analysis of autonomous mobility on demand systems.

7.2.1 Limitations

Although this research was carefully prepared, we are still aware of its limitations and shortcomings. First of all, there are strong assumptions in our models.

1. We run the system with single-class vehicles, e.g., there is only one type of service offered (no limo taxis, no vans, etc.).
2. Our facility location models do not consider
    (i) Geographic information, which is required for cross-checking of the feasibility of potential locations;

    (ii) Capacity of each station; at the current implementation each station can *store* as many vehicles as needed, however we are aware that this assumption should be released.

    (iii) Requirements for charging infrastructure at each station; in real situation electric vehicles are running on batteries with limited range and therefore they have to be recharged. In our work, we neglected the battery capacity and range of the vehicles.
3. Our assignment models allow maximum of two customers to share the ride.
4. Offline rebalancing models do not account for time-varying travel times and uncertainties such as weather conditions and traffic accidents.

Second, we were limited by scalability of the simulator, which is still under development. Due to very detailed representation of the network and agents' choices and movement, run-time was a crucial factor limiting us from performing more simulations.

Third, we could not validate the simulations because autonomous mobility on demand services are not far beyond being at a conceptual stage with some small scale *Research and Development* implementations.

Fourth, we do not provide analysis of robustness of the proposed algorithms, nor comment on the stability and convergence of the developed control policies.

In addition, impact of AMOD parking locations within the CDB on land-use were ignored and residents of the CBD area were not given any privilege of driving their own vehicle within the restricted area. ERP (Electronic Road Pricing, or toll system in Singapore) was not included in the analysis.

We fully acknowledge all the shortcomings, which could not be released due to time constraints of the program.

7.2.2 Future Work

We are planning to extend the AMOD concept—which is currently designed to provide door-to-door transportation—and integrate it with other modes of transportation, such as public transit. We believe that the concept of AMOD system may provide a very effective means to increase the use of a scheduled public transportation system if it is used as a feeder service. In such scenario, a vehicle would bring rider(s) from their origins to a public transport hub, i.e., MRT station in Singapore, then the riders would use public transit to get close to their destinations, and finally they would walk or use another AMOD vehicle to travel from the transit stop to their destinations.

Further extensions of the proposed solutions for the facility location problem should consider geographic information, cost of placing new car park sites and the parking capacity of each station. We are also working towards including the charging requirements when designing the stations' location.

For the assignment models, we are interested in setting priority for some customers or vehicles, e.g., loyalty customers or vehicles with a higher fuel economy. We are also interested in incorporating the demand management policies in our AMOD Controller, e.g., dynamic pricing for a better respond to the fluctuating demand. It is also within our interest to extend the assignment model to account for the battery level before dispatching vehicle to the customer and to work toward more sophisticated methods for multiple-riders assignment.

Furthermore, we are planning to design congestion-aware rebalancing methods, which is especially important when rebalancing a large number of vehicles. It is also within our interest to extend the rebalancing model to incorporate the battery level in our rebalancing algorithms.

Finally, we would like to test AMOD Controller in different cities and for a real vehicle implementation.

Appendix A

Input and Output of AMOD Controller

A.1 Input to AMOD Controller

AMOD Controller is designed at three levels: planning, tactical and operational (Figure 2.4). AMOD Controller Planning Level utilizes demand and network information, which are provided by (i) SimMobility Mid-Term, and (or) (ii) external sources. The overall process is described in Figure 2.13. AMOD Controller Tactical Stage uses information on station location and estimated demand (as presented in Figure 2.14). Both inputs are provided by AMOD Controller Planning Stage. At the operation level, AMOD Controller is initialized with estimated demand data, network information, stations location, initial positions of vehicles and static rebalancing counts (Figure 2.15).

Details on input from SimMobility Mid-Term and external sources are presented in Section A.1.1 and Section A.1.2, respectively.

A.1.1 Input from SimMobility Mid-Term

Input from SimMobility Mid-Term is directly utilized by AMOD Controller Planning Stage and indirectly (as input from Planning Stage) by Tactical and Operational Stage. The input consists of (i) demand data, and (ii) network information. The data are used as described in Chapter 6.

SimMobility Mid-Term provides demand in the form of daily activity schedule (DAS). Each row of DAS consists of the following structure: `person id` is a unique id for the person; `tour_no` describes sequence number of the trip for the person; `tour_type` describes purpose of the trip, e.g., work-related trip; `stop_no`, `stop_type`, `stop_location`, which is understood as trip destination; `stop_zone`, `stop_mode`, which is understood as trip mode; `primary_stop`, `arrival_time`, `departure_time`, `prev_stop_location`, which is understood as trip origin; `prev_stop_zone`, `prev_stop_departure_time`, which is understood as trip tart time.

Network structure within SimMobility Mid-Term comprises of list of nodes, edges, connectors, traffic signals, etc. Each element is described by a string, e.g., node description format: `("node", 0, 12340, {"location":"[(372322.09,142722.57),]", "type":"2",})`, link description format: `("link", 0, 3055, {"name":"ORCHARD ROAD","from":"12896", "to":"22130", "type":"2","category":"2","fwd-path":"[7037,]",})`. Each node is characterized by a unique node ID and *(x, y)* coordinates compatible with WGS N48 system. Each link is characterized by a unique link ID, name of the road, which link is representing, start and end node ID of the link and additional information such us type and category of the link.

Given the input, we convert the daily activity schedules and network information into the format which is accepted by AMOD Controller and we pass it as an input to our simulations. As described in Section AMOD Controller Operational Level accepts the following format for the network: list of nodes `node_id`, `position_x`, `position_y`. Each node is described by unique ID and its coordinates. Coordinates are provided in planar system (WGS coordinate system; for Singapore, zone N48). The demand accepted by AMOD Controller Operational Stage consists of list of customers and list of bookings. As customers we understand all agents who exist in the system. They are initialized at the beginning of the simulation

A.1.2 External Data Sources

External (from outside of SimMobility Mid-Term and Short-Term) data sources which are input to Controller consist of: (i) Land-use data (ii) HITS 2012 (iii) 4.5 months detailed GPS data for 15,000 taxis (iv) 3 months EZ-Link data (v) Google transit data (vi) SCATS traffic light data (vii) Road network data All the sets are fed into SimMobility Mid-Term to generate daily activity schedules for the trips and trip chains which are passed to AMOD Controller and SimMobility Short-Term. Detailed description of models within SimMobility Mid-Term is outside the scope of this thesis, and for more information the reader may refer to [Lu et al., 2015, Adnan et al., 2016, Azevedo et al., 2016]. Setup and validation of SimMobility demand model is presented in Appendix E.

Passing input to the Controller takes place at the initialization phase, which is presented in Algorithm A.1. Note that Algorithm A.1 presents initialization of required parameters only, i.e., optional parameters are specified as a group of parameters.

The input parameters are read from the configuration file and passed to the controller. Additional parameters which can be specified within the configuration file of AMOD Controller include:

(i) Assignment parameters such as assignment algorithm (FIFO, bipartite matching, shared) distance cost factor, and waiting time cost factor.

(ii) Rebalancing parameters such as rebalancing method, e.g., queue-based, predictive, or based on offline estimation.

(iii) Logging parameters such as name of the output file, logging time interval.

(iv) Simulation parameters such as simulator name, e.g., SimMobility, Beta-simulator, and simulation time, i.e., what period of time do we simulate.

Algorithm A.1: Initialization phase within AMOD Controller Operational Level.

```
if any of the input files is invalid then
    throw error
end
load network nodes and edges
generate the network
load stations
if vehicle initialization optimized then
    initialize vehicles at stations based on optimized allocation
else
    uniformly distribute vehicles
end
load customers
if static rebalancing then
    load static rebalancing counts
end
if dynamic rebalancing then
    load rebalancing interval
    load demand predictions
end
if external simulator then
    establish connection with simulator
    if connection unsuccessful then
        throw error
    end
end
load additional parameters
pass data to update phase
```

All parameters initialized during the initialization phase are passed to the update phase of the controller (and simulator) and are stored *in-memory* for the entire simulation time.

A.2 Output of AMOD Controller

Output from AMOD Controller consists of all logs reported by AMOD Controller (Section B). Events which are logged are the following:

- `vehicleMoved`, which provides information on vehicle location with a time stamp.
- `BookingReceived`, which contains information about booking id, customer id, booking time and service requested, e.g., shared ride.
- `BookingServiced`, which provides details on which vehicle is assigned to which customer, and at what time.
- `BookingDiscarded`, which is logged with the reason, i.e., no path found or customer waiting time exceeded.
- `VehicleDispatch`, which can be logged with different type, i.e., dispatch to customer or dispatch to rebalancing.
- `VehicleArrival`, which provides coordinates of the location vehicle has reached, and type of arrival, i.e., arrival for pick-up, drop-off, rebalancing.
- `CustomerPickup`, describes who was picked up by which vehicle and at what time.
- `CustomerDropoff` describes who was dropped off by which vehicle and at what time.

Note that each event is logged with the time stamp of the time when event was triggered.

All events can be later analyzed and visualized offline. An example of visualization of extended CBD region can be found here. In case of any problems with opening the link, please contact the author.

Appendix B

Update Phase of AMOD Controller

Update phase of AMOD Controller is presented in Algorithm B.1. AMOD Controller performs update after successful initialization phase at every time tick. Time tick is a parameter specified at the initialization phase and ranges from 0.1 sec to 10 seconds. Resolution lower than 0.1 seconds slows down the performance of the controller and simulator. Furthermore, for any real system implementation, such low latency is almost impossible to be achieved. Therefore, our simulation were run with time tick of 1 second. Resolution coarser than 3 seconds causes simulator to *jump* from state to state resulting in problems with driver behavior model.

At every time tick we check if the current time is less than the maximum simulation time we are interested to run. If the condition is satisfied, we perform the update.

At each update, we first update the simulator state. We call a function within AMOD Controller which calls location service within Simulator, e.g., if we use SimMobility, the AMOD Controller is directly attached to SimMobility. This allows the controller to call some of the SimMobility classes directly, so to find the locations of all vehicles which are of type *AMOD*. Note, that vehicle and customer statuses are maintained within AMOD Controller only. After updating vehicles position, Controller updates its own state by updating statues of all vehicles and customers, e.g., if vehicle has arrived at pick up location, the status is changed from `MOVING_TO_PICK_UP`, to `PICKING-UP`, followed by `MOVING TO DESTINATION`. A complete list of vehicle statuses is the following:

- `FREE`,
- `BUSY`,
- `HIRED`,
- `MOVING_TO_PICKUP`,
- `MOVING_TO_DROPOFF`,
- `PICKING_UP`,
- `DROPPING_OFF`,

- PARKED,
- MOVING_TO_REBALANCE,
- MOVING_TO_FIRST_PICKUP,
- MOVING_TO_SECOND_PICKUP,
- PICKING_UP_FIRST,
- PICKING_UP_SECOND,
- MOVING_TO_FIRST_DROPOFF,
- MOVING_TO_SECOND_DROPOFF,
- DROPPING_OFF_FIRST,
- DROPPING_OFF_SECOND,
- UNKNOWN.

In the next step, the controller logs all events to the output database as presented in Section A.2. All events can be later analyzed and visualized offline.

At the next step, AMOD Controller performs assignment which is done based on the algorithms described in Section 4.2. Note, that the assignment can only be done if there are customers in the queue and available vehicles.

Finally, after the assignment is successful, the controller performs rebalancing. Rebalancing is only done at specific time intervals, so if the time condition is met, the optimization class for rebalancing is triggered. As a results, empty vehicles are dispatched to new locations.

Next, the cuntroller updates the simulator state, which is again, beginning of the loop.

Algorithm B.1: Update phase within AMOD Controller Operational Level.

```
if initialization not successful then
    throw error
end
do
    if simulator is valid then
        update simulator state
        update controller state
        log positions of all vehicles
    end
    read demand file
    log incoming bookings
    add bookings to booking queue
    if booking queue not empty then
        if there are available vehicles then
            assign vehicles to trips
            dispatch vehicles to simulator
            log dispatch orders
        end
    end
    if there are available vehicles then
        if static rebalancing then
            read predicted rebalancing counts and make online adjustments
            dispatch rebalancing vehicles to simulator
            log dispatches
        else
            optimize online rebalancing counts
            dispatch rebalancing vehicles to simulator
            log dispatches
        end
    end
until time exceeds simulation end time
```

Appendix C

Class Structure for Physical Elements of AMOD System

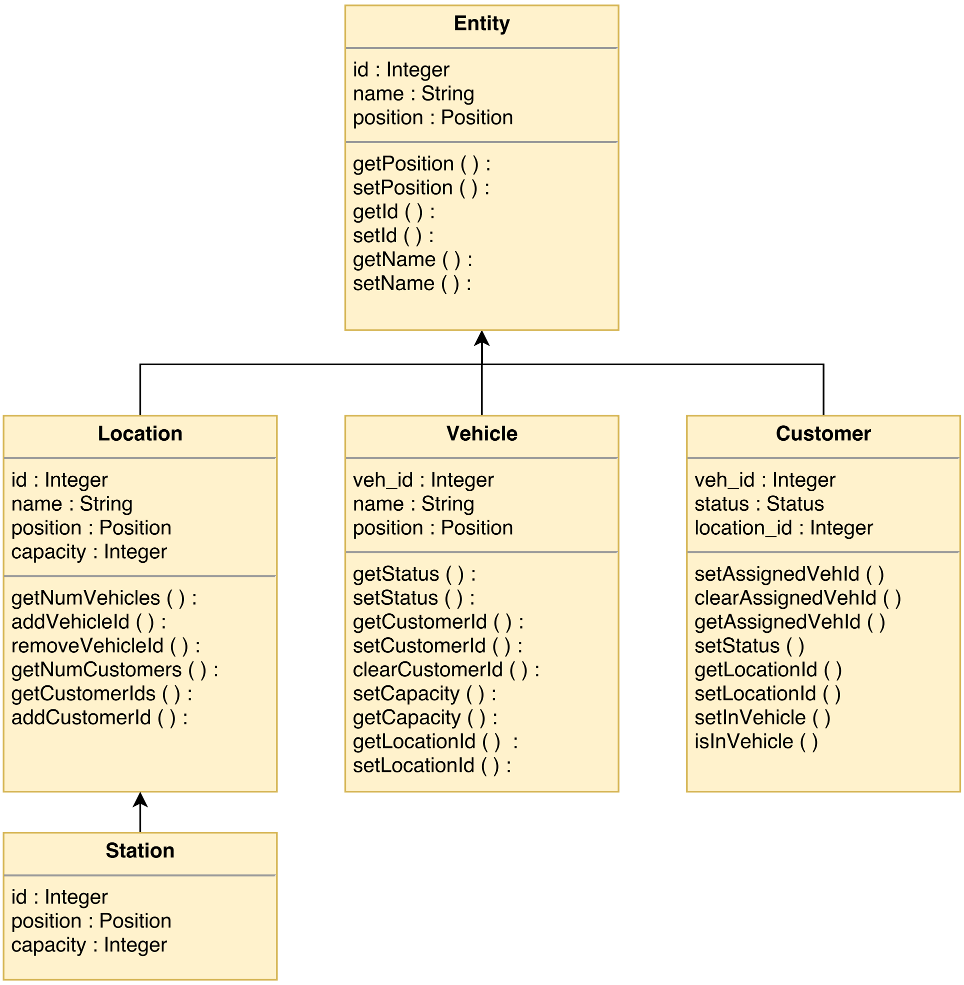


Figure C.1: Detailed class diagram for the physical elements of AMOD system.

Appendix D

Modeling Methodology within SimMobility

The demand generation process for AMOD trips is based on integration of SimMobility MT simulator with SimMobility ST simulator as described in details in our publications [Azevedo et al., 2016, Azevedo et al., 2017]. The supply models for AMOD vehicles are based on driver's models in SimMobility ST.

To analyze the impacts of specific AV technologies on travel patterns, SimMobility demand and supply simulation components were extended to account for dedicated AMOD service and vehicle access restrictions. For the case studies presented in Chapter 6, we limit the analysis to the ST and MT simulators, because extension of SimMobility LT is still in progress. SimMobility ST is responsible for advancing agents on the transportation network according to their respective behavioral and decision models. The behavioral and decision models within SimMobility ST are based on the open-source microscopic traffic simulation application MITSIM. SimMobility ST includes probabilistic model to capture drivers' route choice decisions and driving behavior parameters. These parameters are randomly assigned to each driver-vehicle unit. In case of AMOD the parameters are set to be the same for all vehicles. All vehicles within SimMobility are moved according to route choice, acceleration and lane changing models. The acceleration model captures drivers' response to neighboring conditions as a function of surrounding vehicles motion parameters. The lane changing model integrates mandatory and discretionary lane-changes in a single structure. The AV decision making models should, preferably, be based on the motion control algorithms used by the AV manufacturers. Due to the fact that AVs are still under research and development, we could not find any readily available specifications of the AVs' decisions making process. Therefore, for this implementation, the existing acceleration and lane-changing models in the vehicle flow model of SimMobility ST were adjusted to exclude human (driver's) heterogeneity factors and individual stochastic behavior. All AV behave the same way, and the safety margins in terms of gap acceptance, safety headway and reaction time were reduced and equal to 1.0s, 1.0s and 0.5s, respectively.

SimMobility MT simulates daily activities and travels at the individual level. It combines activity-based microscopic simulation on the demand side with mesoscopic simulation on the supply side. The demand side comprises two groups of behavioral models: pre-day models and within-day models. The pre-day models follow an enhanced version of econometric Day Activity Schedule approach predicts:

- the activity sequence (including home-based tours, work-based sub-tours, and intermediate stops),
- the trip destinations and modes, and
- the departure times (on half-hour slots).

This is based on a sequential application of hierarchical discrete choice models using a Monte Carlo simulation. Readers are directed to [Lu et al., 2015] to get details of the pre-day modeling framework. At the pre-day level, the implementation of a car-restricted area with AMOD service was assumed to affect directly the destination and mode choice. Mode availability for trips involving origins and/or destinations within the restricted area changed leading to multi-modal trips that combine private vehicles (in the case those are the modes restricted) outside the implementation area, and AMOD inside the implementation area. As transferring between modes is forced for these trips, it is necessary to properly model the agents' behavioral response. For this particular case, the utility specification of the AMOD mode was based on the individual preferences towards taxi due to the lack of AMOD-specific data for model estimation. For mandatory activities with fixed destination (such as going to work or school) the agents were only able to change modes, while for non-mandatory activities (such as shopping) they also had the possibility of changing destination. Once the daily activity schedules are obtained for all agents, the within-day models predict the routes for planned trips, transforming the activity schedule into actual trips. Depending on the traffic conditions and effective travel times, the agents could reschedule the remainder of the day, cancel an activity, re-route while traveling (including alighting a bus to change route), or run an opportunistic activity such as shopping while waiting. On the within-day level, the implementation of the restricted area with AMOD service affected the route choice, i.e., private vehicles had a smaller number of available paths which may lead to a change in congestion on alternative paths.

The cost of the AMOD service is assumed as 40% less than the regular taxi service in Singapore, resulting in an average cost of about $3 SGD within the CBD area. Reduction of cost for the AMOD service is due to different cost structure for the autonomous taxis (as compared to human driven ones). Our estimates—which are based on analysis of the ComfortDelgro Taxi data—suggest that if we eliminate the driver, the operator can make around 60% savings. Autonomous system, however, brings higher operational cost, i.e., control and management offices, which we assumed to be 20% higher than for human driven cabs.

The cost structure of other modes is assumed to remain unchanged (i.e. buses and MRT kept their frequencies, fares and capacities and the taxi fleet and cost remained the same) using the year 2012 configuration as reference. No changes in the road network and traffic control systems is assumed for this case study. The impacts of AMOD parking locations within the CDB on land-use were ignored and residents of the CBD area were not given any privilege of driving their own vehicle within the restricted area. These strong assumptions allowed us to test the models and prove that the controller and simulator give us expected results.

With a growing interest in AV technology and mobility on demand systems, we are planning to relax some of these assumptions, however it falls beyond the scope of this thesis.

Appendix E

Demand Model within SimMobility

SimMobility consists of three pilars: long-term, mid-term, and short-term, each of them simulating the same agents at different spacial and temporal resolutions. The MT simulator takes as input a multi-modal network and a population (which may come from the LT simulator or other population data sets) that contains detailed characteristics of each agent. As an output, it passes accessibility measures (in the form of Logsums) from the pre-day component of MT simulator to the LT simulator. The MT simulator provides the ST simulator with trip chains as input demand to simulate smaller network regions in more detail. Short-Term simulator simulates vehicles movement at the operational stage. A snapshot of a mixed traffic within SimMobility Short-Term is presented in Figure E.1. We observe different classes of vehicles queuing at the intersection. Private vehicles are in red color, taxis are in blue color, AMOD vehicles are in green color.

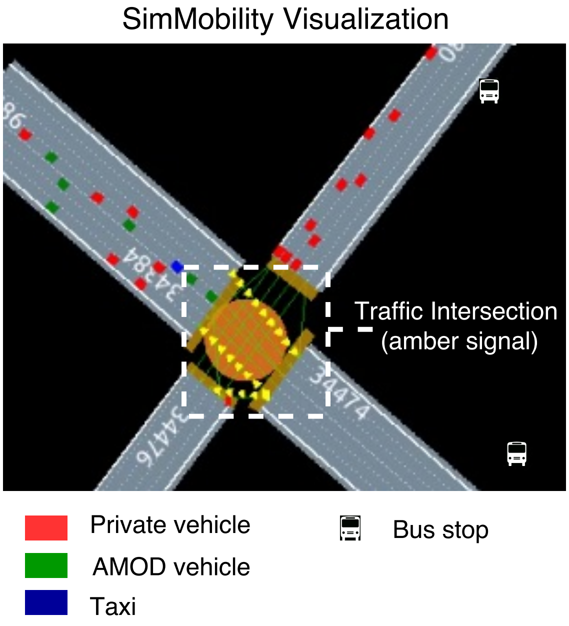


Figure E.1: A snapshot of visualization of SimMobility ST showing different classes of vehicles queuing at the intersection. Private vehicles are in red color, taxis are in blue color, AMOD vehicles are in green color.

Figure E.2 presents results for validation of baseline model within SimMobility Short-Term. The validation is done against HITS 2012 data.

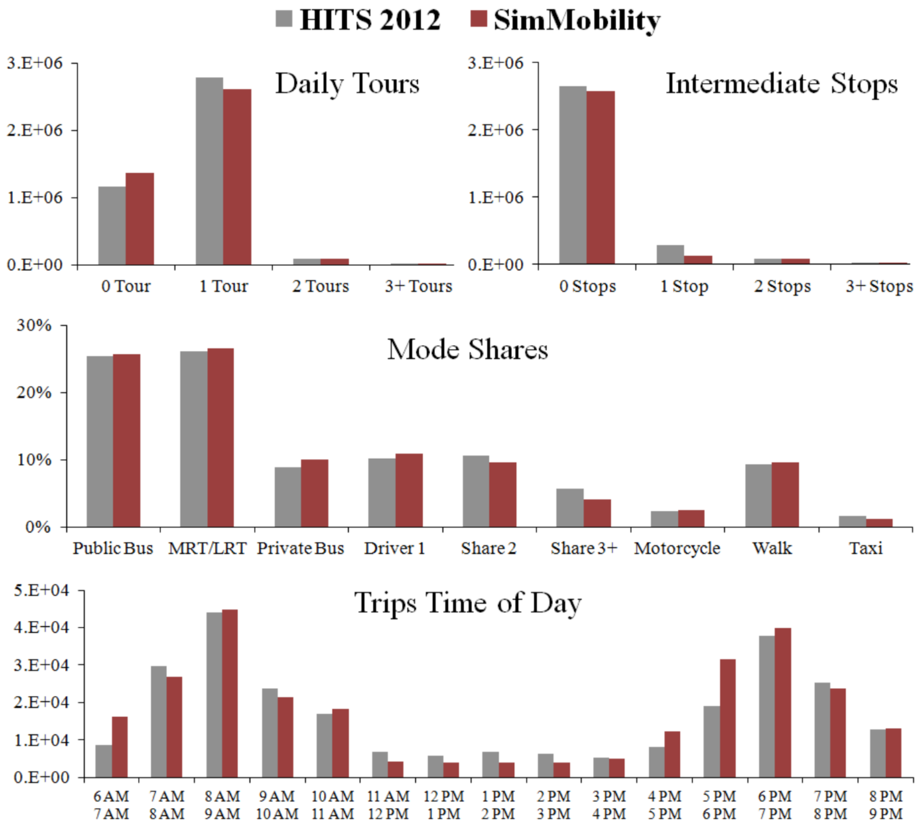


Figure E.2: Validation of baseline model in SimMobility based on our work published in [Azevedo et al., 2016] and [Azevedo et al., 2017].

Model validation and calibration within SimMobilityST applies several design heuristics to make modeling and development easier for a heterogeneous user base as described in [Azevedo et al., 2017]:

1. Entities are isolated from each other, and can only interact through properties that are shared among them, which is a property of agent-based simulations.
2. The simulator is location-agnostic regarding agents. In other words, an agent's interface does not change depending on where it is with respect to the network.
3. Time step within SimMobility is indivisible and agents are assumed to all tick forward at once.
4. SimMobility is hierarchical and provides sensible defaults, e.g., trip-chains can be filled in with more information as the agent's trip progresses.

SimMobility uses an activity-based demand formulation in the form of activity-schedules rather than the traditional Origin-Destination matrix definition. In such approach, trip chains are generated by individuals' daily schedules instead of aggregated traffic specific matrices. Demand parameters are calibrated through tuning of the OD flows by calibrating one parameter for each activity schedule [Azevedo et al., 2017]. Then the updated OD parameters are converted into trip chains by dis-aggregating through so-called killing-cloning process for each iteration. There are also parameters related to the route choice, which are calibrated as described in [Azevedo et al., 2017, Adnan et al., 2016]. Further details on SimMobility, in terms of modeling details, integration, and calibration can be found in [Lu et al., 2015, Adnan et al., 2016, Azevedo et al., 2016, Azevedo et al., 2017].

In the case-study of 14 km^2 CBD region, we have simulated the background traffic which includes bus rides, taxi rides, private vehicles rides and MRT rides (MRT were not directly simulated) alongside the AMOD vehicles.

As SimMobilility is still under development, there were limited data available. In the case-study of the Extended Central Business District and the entire Singapore, we had no background traffic information and therefore we performed analysis with AMOD trips only. Our future work, which is currently being undertaken, is to extend the simulations to simulate AMOD vehicles alongside other transportation modes in the entire Singapore.

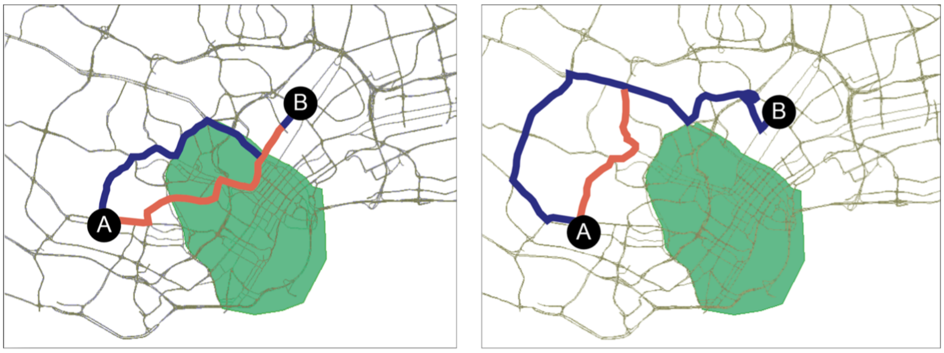
E.1 Analysis of Modal Shift within SimMobility

Analysis of the modal shift has been performed for the 14 km^2 CBD region. As described in our papers [Marczuk et al., 2015, Azevedo et al., 2016] , we evaluated policy of restricting private vehicles from entering the 14 km^2 zone and introducing AMOD within the region. The baseline traffic was validated against HITS 2012 data as presented in Figure E.2. In the case study, outside of the region all modes are allowed, while inside the zone we allow AMOD and public transport including taxis.

As we described ealrier in [Azevedo et al., 2016], we use SimMobility ST and AMOD Controller to test different fleet sizes for the AMOD service. After running simulations, we compare waiting and travel times within the restricted zone were we simulated demand in the form of trip chains from the SimMobility MT simulator, i.e., SimMobility ST simulator simulates the demand only within the restricted zone. This is done by dividing the private vehicle trips, which arrive or originate within the CBD region and have their origin or destination outside the CBD region into two sub-trips, i.e., one sub-trip is inside the CBD region and the second one is outside the CBD region. The inside CBD sub-trip is simulated using AMOD service. This is also true for those trips which have their origin and destination within the CBD region. Travel times on the links within Short-Term simulator are transferred back to the MT simulator. Within SimMobility Mid-term we combine the two sub-trips (inside and outside the CBD) into one complete trip. Mid-Term supply model stores trip travel times in an aggregated manner which is fed back to the pre-day component of SimMobility Mid-Term.

The pre-day model understands each private vehicle trip as a combined modal trip, i.e., private vehicle and AMOD, if part of the trip is inside CBD. This is done by modifying the utility specification of private vehicle mode in the mode choice model by adding waiting time into consideration. Parking prices for private vehicles are reduced due to the restriction for parking within the CBD region. The cost of the AMOD part of the trip is considered 40% lower than the similar taxi trip. Finally, Mid-Term Within-day simulated route choice of private vehicle trips considering unavailability of routes through the CBD region. With these assumptions, several iterations were run in two simulators.

This integration allows us to consider the impacts of introducing AMOD within the CBD region for an entire Singapore transportation network, along with behavioral changes in the individual's activity schedules. Our results show a significant change in the travel pattern due to this scenario, e.g., commuters destination choice for some trips changed, as some travelers showed preference to shop outside the CBD. The restricted zone affects route choice of through traffic and the performance of the road network. Based on the simulation output we conclude that private vehicles users shifted to public transport and taxis due to limited usage of private vehicles. Furthermore, we can see that the restricted zones causes some congestion outside of the region. We concluded that it is due to the vehicles which are taking detours and otherwise, would be traveling through the zone as shown in Figure E.3.



Path	Length	Travel Time	Split
1	10.6 km	17 min	13.0%
2	10.1 km	14 min	32.4%

Path	Length	Travel Time	Split
1	18.7 km	32 min	10.1%
2	15.4 km	29 min	25.3%

Figure E.3: Effects on through traffic: path attributes for the two selected paths from A to B, without (left) and with (right) the car-restricted area with AMOD.

We also conclude that after introducting AMOD service within restricted zone, number of private trips is reduced by 7% as compared to the case without AMOD.

Impact on mode choice for the restricted area is presented in Figure E.4.

We see that most of commuters replaced their trips on private vehicles by combined trips with AMOD.

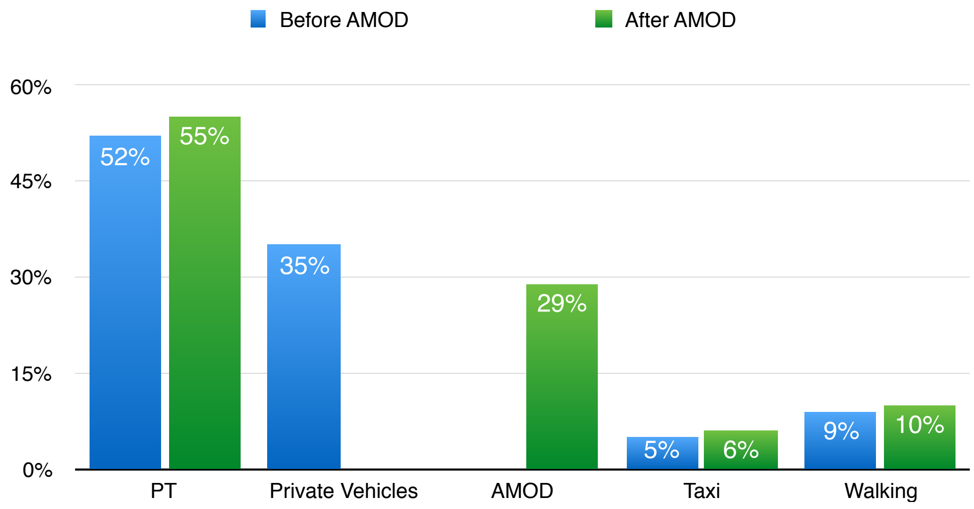


Figure E.4: Impact on the mode choice of the car-restricted area with AMOD.

E.1.1 Limitations

Impact of AMOD parking locations within the CDB on land-use were ignored and residents of the CBD area were not given any privilege of driving their own vehicle within the restricted area. ERP (Electronic Road Pricing, or toll system in Singapore) was not included in the analysis.

References

[Adnan et al., 2016] Adnan, M., Pereira, F., Azevedo, C., Basak, K., Lovric, M., Raveau, S., Zhu, Y., Ferreira, J., Zegras, C., and Ben-Akiva, M. (2016). Simmobility: A multi-scale integrated agent-based simulation platform. In *The 95th Annual Meeting, Transportation Research Board*.

[Agatz et al., 2012] Agatz, N., Erera, A., Savelsbergh, M., and Wang, X. (2012). Optimization for dynamic ride-sharing: A review. *European Journal of Operational Research*, 223(2):295–303.

[Agatz et al., 2011] Agatz, N. A., Erera, A. L., Savelsbergh, M. W., and Wang, X. (2011). Dynamic ride-sharing: A simulation study in metro atlanta. *Transportation Research Part B: Methodological*, 45(9):1450–1464.

[Aimsun, 2016] Aimsun (2016). Aimsun, TSS-Transport simulation systems. https://www.aimsun.com/. Accessed: 25-Nov-2016.

[Alshamsi et al., 2009] Alshamsi, A., Abdallah, S., and Rahwan, I. (2009). Multiagent self-organization for a taxi dispatch system. *Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems*.

[Amey, 2011] Amey, A. (2011). A proposed methodology for estimating rideshare viability within an organization, applied to the mit community. In *TRB Annual Meeting Procediings*, pages 1–16.

[Atallah and Blanton, 2009] Atallah, M. J. and Blanton, M., editors (2009). *Algorithms and Theory of Computation Handbook. Second Edition.* Chapman and Hall/CRC.

[Atasoy et al., 2015] Atasoy, B., Ikeda, T., Song, X., and Ben-Akiva, M. E. (2015). The concept and impact analysis of a flexible mobility on demand system. *Transportation Research Part C: Emerging Technologies*, 56:373–392.

[Audi Unite, 2016] Audi Unite (2016). Audi Unite. Join a cirle. Share an Audi. https://www.audiunite.com/. Accessed: 27-Aug-2016.

[Azevedo et al., 2016] Azevedo, C. L., Marczuk, K., Raveau, S., Soh, H., Adnan, M., Basak, K., Loganathan, H., Deshmunkh, N., Lee, D.-H., Frazzoli, E., and Ben-Akiva, M. (2016). Microsimulation of demand and supply of autonomous mobility on-demand. *Journal of the Transportation Research Board*.

[Azevedo et al., 2017] Azevedo, C. L., Oh, S., Deshmukh, N. M., Marimuthu, B., Marczuk, K., Soh, H., Basak, K., Toledo, T., Peh, L.-S., and Ben-Akiva, M. (2017). Simmobility short-term: An integrated microscopic mobility simulator. *Journal of the Transportation Research Board*.

[Balea, 2016] Balea, J. (2016). World's first driverless taxis kick off public trial in Singapore. https://www.techinasia.com/world-first-driverless-taxis-start-singapore. Accessed: 27-Aug-2016.

[Bank, 2012] Bank, W. (2012). Turn down the heat. why a 4c warmer World must be avoided. http://documents.worldbank.org/curated/en/865571468149107611/Turn-down-the-heat-why-a-4-C-warmer-world-must-be-avoided. A Report for the World Bank by the Potsdam Institute for Climate Impact Research and Climate Analytics.

[Barrios and Godier, 2014] Barrios, J. A. and Godier, J. D. (2014). Fleet sizing for flexible carsharing systems. simulation-based approach. *Transportation Research Record: Journal of the Transportation Research Board*, 2416:1–9.

[Barth and Todd, 1999] Barth, M. and Todd, M. (1999). Simulation model performance analysis of a multiple station shared vehicle system. *Transportation Research Part C: Emerging Technologies*, 7(4):237–259.

[Beasley, 1988] Beasley, J. E. (1988). An algorithm for solving large capacitated warehouse location problems. *European Journal of Operational Research*, 33(3):314–325.

[Bertsimas and Tsitsiklis, 1997] Bertsimas, D. and Tsitsiklis, J. (1997). *Introduction to Linear Optimization*. Athena Scientific and Dynamic Ideas, Belmont, Massachusetts, USA.

[Bonabeau, 2002] Bonabeau, E. (2002). Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences of the United States of America*, 99:7280–7287.

[Borshchev and Filippov, 2004] Borshchev, A. and Filippov, A. (2004). From system dynamics and discrete event to practical agent based modeling: Reasons, techniques, tools. In *The 22nd International Conference of the System Dynamics Society*, Oxford, England.

[Boyaci et al., 2015] Boyaci, B., Zografos, K. G., and Geroliminis, N. (2015). An optimization framework for the development of efficient one-way car-sharing systems. *European Journal of Operational Research*, 240:718–733.

[Brown, 2015] Brown, C. (2015). Carsharing: State of the market and growth potential. http://www.autorentalnews.com/channel/rental-operations/article/story/2015/03/carsharing-state-of-the-market-and-growth-potential.aspx. Accessed: 25-Aug-2016.

[Brownell and Kornhauser, 2014] Brownell, C. and Kornhauser, A. (2014). A driverless alternative. fleet size and cost requirements for a statewide autonomous taxi network in new jersey. *Transportation Research Record: Journal of the Transportation Research Board*, 2416:73–81.

[Bruglieri et al., 2013] Bruglieri, M., Colornia, A., and Lue, A. (2013). Bike sharing systems: Solving the static rebalancing problem. *Discrete Optimization*, 10:120–146.

[Bruglieri et al., 2014] Bruglieri, M., Colornia, A., and Lue, A. (2014). The vehicle relocation problem for the one-way electric vehicle sharing: an application to the Milan case. *Procedia of Social and Behavioral Sciences after the 16th Meeting of the EURO Working Group on Transportation*, 111:18–27.

[Burkard and Cela, 1999] Burkard, R. E. and Cela, E. (1999). Linear assignment problems and extensions. In *Handbook of combinatorial optimization*, pages 75–149. Springer.

[Caggiani and Ottomanelli, 2012] Caggiani, L. and Ottomanelli, M. (2012). A modular soft computing based method for vehicles repositioning in bike-sharing systems. *Procedia - Social and Behavioral Sciences*, 54:675–684.

[Caggiani and Ottomanelli, 2013] Caggiani, L. and Ottomanelli, M. (2013). A dynamic simulation based model for optimal fleet repositioning in bike-sharing systems. *Procedia - Social and Behavioral Sciences*, 87:203–210.

[Car2Go, 2016] Car2Go (2016). Car2go. https://www.car2go.com/. Accessed: 27-Aug-2016.

[Carlo et al., 2012] Carlo, H., Aldarondo, F., Saavedra, P., and Torres, S. (2012). Capacitated continuous facility location problem with unknown number of facilities. *Engineering Management Journal*, 24(3):24–31.

[Chafkin, 2016] Chafkin, M. (2016). Uber's first self-driving fleet arrives in Pittsburgh this month. http://www.bloomberg.com/news/features/2016-08-18/uber-s-first-self-driving-fleet-arrives-in-pittsburgh-this-month-is06r7on. Accessed: 31-Aug-2016.

[Chen et al., 2013] Chen, D., Kockelman, K., and Khan, M. (2013). The electric vehicle charging station location problem: A parking-based assignment method for Seattle. *Transportation Research Record*, 1254:28–36.

[Cheng and Nguyen, 2011] Cheng, S.-F. and Nguyen, T. D. (2011). Taxisim: A multiagent simulation platform for evaluating taxi fleet operations. In *International Conference on Intelligent Agent Technology*, pages 14–21.

[Church and Velle, 1974] Church, R. and Velle, C. R. (1974). The maximal covering location problem. *Papers in regional science*, 32(1):101–118.

[Ciari et al., 2014a] Ciari, F., Bock, B., and Balmer, M. (2014a). The bike sharing rebalancing problem: Mathematical formulations and benchmark instances. *Omega*, 45:7–19.

[Ciari et al., 2014b] Ciari, F., Bock, B., and Balmer, M. (2014b). Modeling station-based and free-floating carsharing demand. test case study for berlin. *Transportation Research Record: Journal of the Transportation Research Board*, 2416:37–47.

[Contardo et al., 2012] Contardo, C., Morency, C., and Rousseau, L.-M. (2012). Balancing a dynamic public bike-sharing system. Technical report, Rousseau and CIRRELT.

[Cordeau and Laporte, 2003] Cordeau, J.-F. and Laporte, G. (2003). A tabu search heuristic for the static multi-vehicle dial-a-ride problem. *Transportation Research Part B: Methodological*, 37(6):579–594.

[Cordeau et al., 2007] Cordeau, J.-F., Laporte, G., Potvin, J.-Y., and Savelsbergh, M. W. (2007). Transportation on demand. *Handbooks in operations research and management science*, 14:429–466.

[Correia and Antunes, 2012] Correia, G. H. d. A. and Antunes, A. P. (2012). Optimization approach to depot location and trip selection in one-way carsharing systems. *Transportation Research Part E*, 48:233–247.

[Corwin et al., 2015] Corwin, S., Vitale, J., Kelly, E., and Cathles, E. (2015). The future of mobility. How transportation technology and social trends are creating a new business ecosystem. https://dupress.deloitte.com/dup-us-en/focus/future-of-mobility/transportation-technology.html. Accessed: 15-Nov-2016.

[DriveNow, 2016] DriveNow (2016). DriveNow. Carsharing by BMW. https://de.drive-now.com/en/. Accessed: 27-08-2016.

[Dumas et al., 1991] Dumas, Y., Desrosiers, J., and Soumis, F. (1991). The pickup and delivery problem with time windows. *European Journal of Operational Research*, 54(1):7–22.

[Duncan, 2011] Duncan, M. (2011). The cost saving potential of carsharing in a US context. *Transportation*, 38:363–382.

[Edmonds, 1965] Edmonds, J. (1965). Paths, trees, and flowers. *Canadian Journal of mathematics*, 17(3):449–467.

[Ehlers and Erdil, 2010] Ehlers, L. and Erdil, A. (2010). Efficient assignment respecting priorities. *Journal of Economics Theory*, 145(3):1269–1282.

[Erdogan et al., 2012] Erdogan, G., Laporte, G., and Calvo, R. W. (2012). The one-commodity pickup and delivery traveling salesman problem with demand intervals. *Working paper*.

[Erlenkotter, 1978] Erlenkotter, D. (1978). A dual-based procedure for uncapacitated facility location. *Operations Research*, 26(6):992–1009.

[Espada et al., 2010] Espada, I., Luk, J., and Yoo, Y. (2010). Guidelines for selecting techniques for the modelling of road network operations. In *24th ARRB Conference—Building on 50 years of road and transport research.*

[Fagnant and Kockelman, 2015] Fagnant, D. J. and Kockelman, K. (2015). Preparing a nation for autonomous vehicles: opportunities, barriers and policy recommendations. *Transportation Research Record: Journal of the Transportation Research Board*, 77:167–181.

[Fan, 2013] Fan, W. (2013). Management of dynamic vehicle allocation for carsharing systems. stochastic programming approach. *Transportation Research Record: Journal of the Transportation Research Board*, 2359:51–58.

[Fan et al., 2008] Fan, W., Machemehl, R. B., and Lownes, N. E. (2008). Carsharing dynamic decision-making problem for vehicle allocation. *Transportation Research Record: Journal of the Transportation Research Board*, 2063:97–104.

[Farahani et al., 2012] Farahani, R. Z., Asgari, N., Heidari, N., Hosseininia, M., and Goh, M. (2012). Covering problems in facility location: A review. *Computers and Industrial Engineering*, 62:368–407.

[Farahani et al., 2014] Farahani, R. Z., Asgari, N., Heidari, N., Hosseininia, M., and Goh, M. (2014). Dynamic vehicle redistribution and online price incentives in shared mobility systems. *IEEE Transactions on Intelligent Transportation Systems*, 15(4):1567–1578.

[Farahani and Hekmatfar, 2009] Farahani, R. Z. and Hekmatfar, M., editors (2009). *Facility Location. Concepts, Models, Algorithms and Case Studies.* Springer-Verlag Berlin Heidelberg.

[Firnkorn and Muller, 2011] Firnkorn, J. and Muller, M. (2011). Selling mobility instead of cars: New business strategies of automakers and the impact on private vehicle holding. *Business Strategy and the Environment*, 21:264–280.

[Fishman, 2015] Fishman, E. (2015). Bikeshare: A review of recent literature. *Transport Reviews*, 36(1):92–113.

[Fishman et al., 2013] Fishman, E., Washingtonab, S., and Hawortha, N. (2013). Bike share: A synthesis of the literature. *Transport Reviews*, 33(2):148–165.

[Fishman et al., 2014] Fishman, E., Washingtonab, S., Hawortha, N., and Mazzei, A. (2014). Barriers to bikesharing: an analysis from melbourne and brisbane. *Journal of Transport Geography*, 41:325–337.

[Forma et al., 2015] Forma, I. A., Raviv, T., and Tzur, M. (2015). A 3-step math heuristic for the static repositioning problem in bike-sharing systems. *Transportation Research Part B*, 71:230–247.

[Fradea and Ribeiro, 2014] Fradea, I. and Ribeiro, A. (2014). Bicycle sharing systems demand. *Procedia—Social and Behavioral Sciences*, 111:518–527.

[Frazzoli, 2014] Frazzoli, E. (2014). Can we put a price on autonomous driving? http://www.technologyreview.com/view/525591/can-we-put-a-price-on-autonomous-driving/. Accessed: 15-Nov-2014.

[Fricker and Gast, 2014] Fricker, C. and Gast, N. (2014). Incentives and redistribution in homogeneous bike-sharing systems with stations of finite capacity. Technical report, INRIA Paris-Rocquencourt Domaine de Voluceau.

[Furuhata et al., 2013] Furuhata, M., Dessouky, M., Ordóñez, F., Brunet, M.-E., Wang, X., and Koenig, S. (2013). Ridesharing: The state-of-the-art and future directions. *Transportation Research Part B: Methodological*, 57:28–46.

[Galil, 1986] Galil, Z. (1986). Efficient algorithms for finding maximum matching in graphs. *ACM Computing Surveys (CSUR)*, 18(1):23–38.

[Gao et al., 2016] Gao, P., Kaas, H.-W., Mohr, D., and Wee, D. (2016). Disruptive trends that will transform the auto industry. http://www.mckinsey.com/industries/high-tech/our-insights/disruptive-trends-that-will-transform-the-auto-industry. Accessed: 2016-Aug-25.

[Garcia-Palomares et al., 2012] Garcia-Palomares, J. C., Gutierrez, J., and Latorre, M. (2012). Optimizing the location of stations in bike-sharing programs: A gis approach. *Applied Geography*, 35:235–246.

[Garcia-Palomares et al., 2014] Garcia-Palomares, J. C., Gutierrez, J., and Latorre, M. (2014). A survey of simulation platforms for the assessment of public transport control systems. In *International Conference on Advanced Logistics and Transport*, pages 85–90.

[Gauthier et al., 2014] Gauthier, A., Hughes, C., Kost, C., Li, S., Linke, C., Lotshaw, S., Mason, J., Pardo, C., Rasore, C., Schroeder, B., and Trevino, X. (2014). Bike-sharing planning guide. Technical report, Institute for Transportation and Development Policy, USA.

[Geisberger et al., 2009] Geisberger, R., Luxen, D., Neubauer, S., Sanders, P., and Volker, L. (2009). Fast detour computation for ride sharing. Proceedings of the 10th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems.

[Geron, 2016] Geron, S. (2016). A brief history of Autolib'. http://www.paristechreview.com/2016/03/03/a-brief-history-of-autolib/. Accessed: 2016-Aug-27.

[Gibbs, 2016] Gibbs, N. (2016). Car2Go poised to top 1-million users. http://goo.gl/MwwYLA. Accessed: 2016-Aug-25.

[Google, 2016] Google (2016). Google self-driving car project. https://www.google.com/selfdrivingcar/. Accessed: 2016-Nov-15.

[Guerriero et al., 2012] Guerriero, F., Miglionico, G., and Olivito, F. (2012). Revenue management policies for the truck rental industry. *Transportation Research Part E*, 48:202–214.

[Hars, 2016] Hars, A. (2016). Driverless car market watch. http://www.driverless-future.com/. Accessed: 2016-Nov-16.

[Helbing, 2012] Helbing, D. (2012). *Social Self-Organization. Agent-Based Simulations and Experiments to Study Emergent Social Behavior*. Springer-Verlag Berlin Heidelberg.

[Hopcroft and Karp, 1973] Hopcroft, J. E. and Karp, R. M. (1973). An nˆ5/2 algorithm for maximum matchings in bipartite graphs. *SIAM Journal on computing*, 2(4):225–231.

[Horn, 2002] Horn, M. E. (2002). Fleet scheduling and dispatching for demand-responsive passenger services. *Transportation Research Part C*, 10:35–63.

[Jalloh, 2014] Jalloh, M. (2014). Taxi industry: pros and cons of Uber and other e-hail apps. http://www.investopedia.com/articles/investing/110614/taxi-industry-pros-cons-uber-and-other-ehail-apps.asp. Accessed: 2016-Aug-31.

[Jaw et al., 1986] Jaw, J.-J., Odoni, A. R., Psaraftis, H. N., and Wilson, N. H. (1986). A heuristic algorithm for the multi-vehicle advance request dial-a-ride problem with time windows. *Transportation Research Part B: Methodological*, 20(3):243–257.

[Ji et al., 2014] Ji, S., Cherry, C. R., Han, L. D., and Jordan, D. A. (2014). Electric bike sharing: simulation of user demand and system availability. *Journal of Cleaner Production*, 85:250–257.

[Jorge and Correia, 2013] Jorge, D. and Correia, G. (2013). Carsharing systems demand estimation and defined operations: a literature review. *European Journal of Transport and Infrastructure Research*, 13(3):201–220.

[Jorge et al., 2014] Jorge, D., Correia, G., and Barnhart, C. (2014). Comparing optimal relocation operations with simulated relocation policies in one-way carsharing systems. *IEEE Transactions on Intelligent Transportation Systems*.

[Karp, 1972] Karp, R. (1972). Reducibility among combinatorial problems. *Complexity of Computer Computations*, pages 85–103.

[Kek et al., 2009] Kek, A. G., Cheub, R. L., Mengc, Q., and Fung, C. H. (2009). A decision support system for vehicle relocation operations in carsharing systems. *Transportation Research Part E: Logistics and Transportation Review*, 54(1):149–158.

[Khuller et al., 1994] Khuller, S., Mitchell, S., and Vazirani, V. (1994). On-line algorithms for weighted bipartite matching and stable marriages. *Theoretical Computer Science*, 127:255–267.

[Kleiner et al., 2011] Kleiner, A., Nebel, B., and Ziparo, V. (2011). A mechanism for dynamic ride sharing based on parallel auctions. In *22th International Joint Conference on Artificial Intelligence*, pages 266–272.

[Kuhn, 1955] Kuhn, H. W. (1955). The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97.

[Kuhnimhof et al., 2012] Kuhnimhof, T., Buehler, R., Wirtz, M., and Kalinowska, D. (2012). Travel trends among young adults in Germany: increasing multimodality and declining car use for men. *Journal of Transport Geography*, 24:443–450.

[Kumar and Bierlaire, 2012] Kumar, P. and Bierlaire, M. (2012). Optimizing locations for a vehicle sharing system. Technical report, Ecole Polytechnique Federale de Lausanne, Switzerland.

[Lam et al., 2014] Lam, A., Leung, Y.-W., and Chu, X. (2014). Electric vehicle charging station placement: Formulation, complexity, and solutions. *IEEE Transactions on Smart Grids*, 5(6):2846–2857.

[Land Transport Statistics, 2015] Land Transport Statistics (2015). Singapore land transport statistics in brief. `http://www.lta.gov.sg/content/ltaweb/en/publications-and-research.html`. Singapore Land Transport Authority. Accessed: 2016-02-16.

[Larsen et al., 2013] Larsen, J., Patterson, Z., and El-Geneidy, A. M. (2013). Build it. but where? the use of geographic information systems in identifying locations for new cycling infrastructure. *Journal of Sustainable Transportation*, 7(4):299–317.

[Le Vine and Polak, 2015] Le Vine, S. and Polak, J. (2015). Autonomous cars: The tension between occupant experience and intersection capacity. *Transportation*, 42:407–411.

[Le Vine et al., 2014] Le Vine, S., Zolfaghari, A., and Polak, J. (2014). Carsharing: Evolution, challenges and opportunities. Technical report, Centre for Transport Studies, Imperial College London.

[Le Vine et al., 2015] Le Vine, S., Zolfaghari, A., and Polak, J. (2015). Introduction to special issue: new directions in shared-mobility research. *Transportation Research Part C*, 52:1–14.

[Lee et al., 2004] Lee, D.-H., Wang, H., Long Cheu, R., and Teo, S. H. (2004). Taxi dispatch system based on current demands and real-time traffic conditions. *Journal of Transportation Research Board*, 1882:193–200.

[Levy et al., 2010] Levy, J., Buonocore, J., and Stackelberg, K. (2010). Evaluation of the public health impacts of traffic congestion: a health risk assessment. *Environmental Health*, 9:65.

[Lin and Yang, 2011] Lin, J.-R. and Yang, T.-H. (2011). Strategic design of public bicycle sharing systems with service level constraints. *Transportation Research Part E*, 47:284–294.

[Litman, 2015] Litman, T. A. (2015). Evaluating carsharing benefits. Technical report, Victoria Transport Policy Institute.

[Liu, 2009] Liu, B. (2009). *Theory and practise of uncertain programming*, volume 239 of *Studies in fuzziness and soft computing*. Springer.

[Liu et al., 2013] Liu, Z., Wen, F., and Ledwich, G. (2013). Optimal planning of electric-vehicle charging stations in distribution systems. *IEEE Transactions on Power Delivery*, 28(1):102–110.

[Lorenz, 2016] Lorenz, L. (2016). Global carsharing services revenue is expected to reach $6.5 billion in 2024. http://goo.gl/ieB429. Accessed: 27-Aug-2016.

[Lu et al., 2015] Lu, Y., Adnan, M., Basak, K., Pereira, F. C., Carrion, C., Saber, V. H., Loganathan, H., and Ben-Akiva, M. E. (2015). SimMobility mid-term simulator: A state of the art integrated agent based demand and supply model. *TRB 93rd Annual Meeting. Washington DC.*

[Maciejewski, 2014] Maciejewski, M. (2014). Benchmarking minimum passenger waiting time in online taxi dispaching with exact offline optimization methods. *Archives of Transport*.

[Maciejewski and Nagel, 2014] Maciejewski, M. and Nagel, K. (2014). A microscopic simulation approach for optimization of taxi services. Technical report, Poznan University of Technology and Berlin Institute of Technology.

[Marczuk, 2010] Marczuk, K. A. (2010). Pedestrian traffic simulation at the bemowo-ratusz bus-tram interchange. B.Eng. Thesis.

[Marczuk, 2015] Marczuk, K. A. (2015). Predictive rebalancing for an autonomous mobility on demand system. presented for the Singapore-MIT Alliance for Research and Technology Symposium.

[Marczuk et al., 2015] Marczuk, K. A., Soh, H. S. H., Azevedo, C. M. L., Adnan, M., Pendleton, S. D., Frazzoli, E., and Lee, D. H. (2015). Autonomous mobility on demand in simmobility: Case study of the central business district in singapore. *IEEE 7th International Conference on Cybernetics and Intelligent Systems and Robotics, Automation and Mechatronics*, pages 167–172.

[Marczuk et al., 2016] Marczuk, K. A., Soh, H. S. H., Azevedo, C. M. L., Lee, D. H., and Frazzoli, E. (2016). Simulation framework for rebalancing of autonomous mobility on demand systems. *5th International Conference on Transportation and Traffic Engineering*.

[Martin et al., 2010] Martin, E., Shaheen, S. A., and Lidicker, J. (2010). Impact of carsharing on household vehicle holdings. results from north american shared-use vehicle survey. *Transportation Research Record: Journal of the Transportation Research Board*, 2143:150–158.

[Martinez et al., 2014] Martinez, L. M., Caetano, L., Eiro, T., and Cruz, F. (2014). An optimisation algorithm to establish the location of stations of a mixed fleet biking system: an application to the city of lisbon. *Procedia–Social and Behavioral Sciences*, 39:94–112.

[Martinez-Barbera and Herrero-Perez, 2012] Martinez-Barbera, H. and Herrero-Perez, D. (2012). Multilayer distributed intelligent control of an autonomous car. *Transportation Research Part C*, 54:513–524.

[MATSim, 2016] MATSim (2016). Matsim, agent-based transport simulation. http://www.matsim.org/. Accessed: 25-Nov-2016.

[Meghjani and Marczuk, 2016] Meghjani, M. and Marczuk, K. (2016). A hybrid approach to matching taxis and customers. *IEEE TENCON (to appear)*.

[Micali and Vazirani, 1980] Micali, S. and Vazirani, V. V. (1980). An o (v| v| c| e|) algoithm for finding maximum matching in general graphs. In *Foundations of Computer Science, 1980., 21st Annual Symposium on*, pages 17–27. IEEE.

[Mills-Tettey et al., 2007] Mills-Tettey, G. A., Stentz, A., and Dias, M. B. (2007). The dynamic hungarian algorithm for the assignment problem with changing costs.

[Mitsim, 2016] Mitsim (2016). Mitsimlab: Technical information. https://its.mit.edu/software/mitsimlab/technical-information. Accessed: 25-Nov-2016.

[National Highway Traffic Safety Administration, 2016] National Highway Traffic Safety Administration (2016). National statistics. https://www.techinasia.com/world-first-driverless-taxis-start-singapore. Accessed: 27-Aug-2016.

[Nourinejad and Roorda, 2014] Nourinejad, M. and Roorda, M. (2014). A dynamic carsharing decision support system. *Transportation Research Part E*, 66:36–50.

[Nourinejad and Roorda, 2015] Nourinejad, M. and Roorda, M. J. (2015). Carsharing operations policies: a comparison between one-way and two-way systems. *Transportation*, 42(3).

[Ozimek, 2014] Ozimek, A. (2014). The massive economic benefits of self-driving cars. http://www.forbes.com/sites/modeledbehavior/2014/11/08/the-massive-economic-benefits-of-self-driving-cars/. Accessed: 15-Nov-2016.

[Paraboschi et al., 2015] Paraboschi, A., Santi, P., and Ratti, C. (2015). Modeling urban-level impact of a shared taxi market. In *14th International Conference on Computers in Urban Planning and Urban Management*, Cambridge, MA, USA. Massachusetts Institute of Technology.

[Paramix, 2016] Paramix (2016). Quadstone paramics. http://www.paramics-online.com/. Accessed: 25-Nov-2016.

[Parzen et al., 2016] Parzen, J., Randall, C., Feigon, S., and Frisbie, T. (2016). Shared-use mobility. reference guide. http://sharedusemobilitycenter.org/what-is-shared-mobility/. Accessed: 23-Aug-2016.

[Paul and Russell, 2016] Paul, D. and Russell, M. (2016). The bike sharing blog. The bike-sharing world—year end data 2015. http://bike-sharing.blogspot.sg/2016/01/the-bike-sharing-world-year-end-data.html. Accessed: 23-Aug-2016.

[Pavone, 2015] Pavone, M. (2015). *Autonomes Fahren*, chapter Autonomous mobility-on-demand systems for future urban mobility, pages 399–416. Springer Berlin Heidelberg.

[Pavone et al., 2011] Pavone, M., Smith, S., Frazzoli, E., and Rus, D. (2011). Load balancing for mobility-on-demand systems. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA.

[Pendleton et al., 2015] Pendleton, S., Uthaicharoenpong, T., Chong, Z. J., Fu, G. M. J., Qin, B., Liu, W., Shen, X., Weng, Z., Kamin, C., Ang, M. A., Kuwae, L. T., Marczuk, K., Andersen, H., Feng, M., Butron, G., Chong, Z. Z., Ang Jr., M. H., Frazzoli, E., and Rus, D. (2015). Autonomous golf cars for public trial of mobility-on-demand service. In *IEEERSJ International Conference on Intelligent Robots and Systems (IROS)*.

[Poultney, 2015] Poultney, L. (2015). The 10 top car tech trends for 2016. http://www.t3.com/news/the-top-ten-automotive-tech-trends-for-2016/. Accessed: 27-Aug-2016.

[Privat, 2015] Privat, L. (2015). Carsharing: 26m users worldwide in 2020. http://www.gpsbusinessnews.com/Carsharing-26M-Users-Worldwide-in-2020_a5752.html. Accessed: 27-Aug-2016.

[Psaraftis, 1980] Psaraftis, H. N. (1980). A dynamic programming solution to the single vehicle many-to-many immediate request dial-a-ride problem. *Transportation Science*, 14(2):130–154.

[Psaraftis, 1983] Psaraftis, H. N. (1983). An exact algorithm for the single vehicle many-to-many dial-a-ride problem with time windows. *Transportation science*, 17(3):351–357.

[Pursula, 1999] Pursula, M. (1999). Simulation of traffic systems—an overview. *Journal of Geographic Information and Decision Analysis*, 3(1):1–8.

[Raviv and Kolka, 2013] Raviv, T. and Kolka, O. (2013). Optimal inventory management of a bike-sharing station. *IIE Transactions*, 45(10):1077–1093.

[Raviv et al., 2013] Raviv, T., Tzur, M., and Forma, I. A. (2013). Static repositioning in a bike-sharing system: models and solution approaches. *EURO Journal on Transportation and Logistics*, 2(3):187–229.

[Rayle et al., 2014] Rayle, L., Shaheen, S., Chan, N., Dai, D., and Cervero, R. (2014). App-based, on-demand ride services: Comparing taxi and ridesourcing trips and user characteristics in san francisco. Technical report, University of California Transportation Center (UCTC). UCTC-FR-2014-08.

[ReachNow, 2016] ReachNow (2016). ReachNow. Carsharing by BMW. http://www.bmwcarsharing.com/. Accessed: 27-Aug-2016.

[Repoux et al., 2015] Repoux, M., Boyaci, B., and Geroliminis, N. (2015). Simulation and optimization of one-way car-sharing systems with variant relocation policies. In *Transportation Research Board 94th Annual Meeting*, volume 15–1907.

[Rodriguez-Valencia, 2014] Rodriguez-Valencia, A. (2014). Taxicab operation in bogota, colombia. empirical findings in day versus night operations. *Journal of the Transportation Research Board*, 2416:92–99.

[Savelsbergh and Sol, 1998] Savelsbergh, M. and Sol, M. (1998). Drive: Dynamic routing of independent vehicles. *Operations Research*, 46(4):474–490.

[Sayarshad et al., 2012] Sayarshad, H., Tavassoli, S., and Zhao, F. (2012). A multi-periodic optimization formulation for bike planning and bike utilization. *Applied Mathematical Modelling*, 26:4944–4951.

[Sethi and Sorger, 1991] Sethi, S. and Sorger, G. (1991). A theory of rolling horizon decision making. *Annals of Operations Research*, 29(1):387–415.

[Shaheen and Guzman, 2011] Shaheen, S. A. and Guzman (2011). Worldwide bikesharing. *ACCESS Magazine at the University of California*, 39:22–28.

[Shaheen et al., 2010] Shaheen, S. A., Guzman, S., and Zhang, H. (2010). Bikesharing in Europe, the Americas, and Asia. past, present, and future. *Transportation Research Record: Journal of the Transportation Research Board*, 2143:159–167.

[Shu et al., 2010] Shu, J., Chou, M., Liu, Q., Teo, C.-P., and Wang, I.-L. (2010). Bicycle-sharing system: deployment, utilization and the value of re-distribution. Working paper.

[Singapore Department of Statistics, 2016] Singapore Department of Statistics (2016). Population trends, 2015.

[Smith et al., 2013] Smith, S., Pavone, M., Schwager, M., Frazzoli, E., and Rus, D. (2013). Rebalancing the rebalancers: optimally routing vehicles and drivers in mobility-on-demand systems. *American Control Conference paper.*

[Smove, 2016] Smove (2016). Smove. https://www.smove.sg/. Accessed: 27-Aug-2016.

[Spieser et al., 2014] Spieser, K., Treleaven, K., Zhang, R., Frazzoli, E., Morton, D., and Pavone, M. (2014). Toward a systematic approach to the design and evaluation of automated mobility-on-demand systems. A case study in Singapore. In *Road Vehicle Automation*, pages 229–245. Springer International Publishing.

[Sumo, 2016] Sumo (2016). SUMO, simulation of urban mobility. https://sourceforge.net/projects/sumo/. Accessed: 25-Nov-2016.

[Sun et al., 2014] Sun, J., Gwee, E., Chin, L. S., and Low, A. (2014). Journeys. Sharing urban transport solutions. Technical report, Singapore Land Transport Authority. Accessed: 16-Nov-2016.

[Teal and Berglund, 1987] Teal, R. and Berglund, M. (1987). The impacts of taxicab deregulation in the USA. *Journal of Transport Economic and Policy*, pages 37–56.

[Tesla, 2016] Tesla (2016). Tesla autopilot. https://www.tesla.com/presskit/autopilot. Accessed: 15-Nov-2016.

[Thorpe et al., 1988] Thorpe, C., Herbert, M., Kanade, T., and Shafer, S. (1988). Vision and navigation for the Carnegie-Mellon Navlab. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(3):362–373.

[Uber, 2016] Uber (2016). Uber. Find a city. https://www.uber.com/cities/. Accessed: 31-Aug-2016.

[Van Roy, 1986] Van Roy, T. (1986). A cross decomposition algorithm for capacitated facility location. *Operations Research*, 34(1):145–163.

[Verter, 2011] Verter, V. (2011). Uncapacitated and capacitated facility location problems. *Foundations of Location Analysis*, 155:25–37.

[Vissim, 2016] Vissim (2016). Ptv vissim. http://vision-traffic.ptvgroup.com/en-us/products/ptv-vissim/. Accessed: 25-Nov-2016.

[Vogel and Mattfeld, 2004] Vogel, P. and Mattfeld, D. C. (2004). Approximation algorithms for facility location problems. Technical report, Research Institute for Discrete Mathematics, University of Bonn, Germany.

[Vogel and Mattfeld, 2010] Vogel, P. and Mattfeld, D. C. (2010). Modeling of repositioning activities in bike-sharing systems. In *12th World Conference on Transport Research.*

[Wang et al., 2010] Wang, H., Huang, Q., Changhua, Z., and Xia, A. (2010). A novel approach for the layout of electric vehicle charging station. In *International Conference on Apperceiving Computing and Intelligence Analysis*, pages 64–70.

[Wang, 2013] Wang, X. (2013). *Optimizing ride matches for the dynamic rde-sharing systems*. PhD thesis, Georgia Institute of Technology.

[Wang et al., 2016] Wang, X., Yuen, C., Ul Hassan, N., An, N., and Wu, W. (2016). Electric vehicle charging station placement for urban public bus systems. *IEEE Transactions on Intelligent Transportation Systems*, PP(99):1–12.

[Wang et al., 2013] Wang, Z., Peng, L., Cui, J., Xi, Y., and Lei, Z. (2013). Research on quantitative models of electric vehicle charging stations based on principle of energy equivalence. *Mathematical problems in engineering*, 2013.

[Xing et al., 2009] Xing, X., Warden, T., Nicolai, T., and Herzog, O. (2009). SMIZE: a spontaneous ride-sharing system for individual urban transit. In *German Conference on Multiagent System Technologies*, pages 165–176. Springer.

[Xu et al., 2003] Xu, H., Chen, Z.-L., Rajagopal, S., and Arunapuram, S. (2003). Solving a practical pickup and delivery problem. *Transportation science*, 37(3):347–364.

[Xu et al., 2015] Xu, H., Ordóñez, F., and Dessouky, M. (2015). A traffic assignment model for a ridesharing transportation market. *Journal of Advanced Transportation*, 49(7):793–816.

[Yang et al., 2010] Yang, H., Xu, W., He, B.-S., and Meng, Q. (2010). Road pricing for congestion control with unknown demand and cost functions. *Transportation Research Part C*, 18:157–175.

[Yang et al., 2000] Yang, Q., Koutsopoulos, H. N., and Ben-Akiva, M. E. (2000). Simulation laboratory for evaluating dynamic traffic management systems. *Transportation Research Record: Journal of the Transportation Research Board*, 1710:122–130.

[Zhang and Pavone, 2015] Zhang, R. and Pavone, M. (2015). Understanding spatiotemporal patterns of biking behavior by analyzing massive bike sharing data in Chicago. *PLoS ONE*.

[ZipCar, 2016] ZipCar (2016). ZipCar. Wheels when you want them. http://www.zipcar.com/. Accessed: 27-Aug-2016.