Imperial College of Science, Technology and Medicine
Department of Civil and Environmental Engineering
Centre for Transport Studies

OPERATIONAL RESEARCH AND SIMULATION METHODS FOR AUTONOMOUS RIDE-SOURCING

Renos Karamanis

A Thesis submitted in fulfilment of requirements for the degree of
Doctor of Philosophy of Imperial College London
January 2021

Declaration of Originality

I hereby certify that all material presented in this thesis which is not my own work has been properly acknowledged.

London, 2021

Renos Karamanis

Copyright Declaration

Renos Karamanis

“ If you hear a voice within you say you cannot paint, then by all means paint and that voice will be silenced.”

Vincent Willem van Gogh

Abstract

Ride-sourcing platforms provide on-demand shared transport services by solving decision problems related to ride-matching and pricing. The anticipated commercialisation of autonomous vehicles could transform these platforms to fleet operators and broaden their decision-making by introducing problems such as fleet sizing and empty vehicle redistribution. These problems have been frequently represented in research using aggregated mathematical programs, and alternative practises such as agent-based models. In this context, this study is set at the intersection between operational research and simulation methods to solve the multitude of autonomous ride-sourcing problems.

The study begins by providing a framework for building bespoke agent-based models for ride-sourcing fleets, derived from the principles of agent-based modelling theory, which is used to tackle the non-linear problem of minimum fleet size. The minimum fleet size problem is tackled by investigating the relationship of system parameters based on queuing theory principles and by deriving and validating a novel model for pickup wait times. Simulating the fleet function in different urban areas shows that ride-sourcing fleets operate queues with zero assignment times above the critical fleet size. The results also highlight that pickup wait times have a pivotal role in estimating the minimum fleet size in ride-sourcing operations, with agent-based modelling being a more reliable estimation method.

The focus is then shifted to empty vehicle redistribution, where the omission of market structure and underlying customer acumen, compromises the effectiveness of existing models. As a solution, the vehicle redistribution problem is formulated as a non-linear convex minimum cost flow problem that accounts for the relationship of supply and demand of rides by assuming a customer discrete choice model and a market structure. An edge splitting algorithm is then introduced to solve a transformed convex minimum cost flow problem for vehicle redistribution. Results of simulated tests show that the redistribution algorithm can significantly decrease wait times and increase profits with a moderate increase in vehicle mileage.

The study is concluded by considering the operational time-horizon decision problems of ride-matching and pricing at periods of peak travel demand. Combinatorial double auctions have been identified as a suitable alternative to surge pricing in research, as they maximise social welfare by relying on stated customer and driver valuations. However, a shortcoming of current models is the exclusion of trip detour effects in pricing estimates. The study formulates a shared-ride assignment and pricing algorithm using combinatorial double auctions to resolve the above problem. The model is reduced to the maximum weighted independent set problem, which is APX-hard. Therefore, a fast local search heuristic is proposed, producing solutions within 10% of the exact approach for practical implementations.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my supervisor, Dr Panagiotis Angeloudis, for his continuous support and direction through the years. Back in 2015, when he was an aspiring junior lecturer at Imperial College London, he offered me, a then inexperienced engineering consultant, the chance to study towards a PhD degree under his supervision. What struck me from our first meeting was his relentless motivation and determination to achieve great things. The onwards and upwards path of his research group, the Transport Systems and Logistics (TSL) Laboratory, is undoubtedly the proof that these character traits persist.

In his efforts to create a research group that sparks innovation, Dr Angeloudis managed to bring together people of great wit and compatible intellect. This environment proved valuable when I explored new ideas and was also a great source of entertainment during our vibrant TSL group dinner nights in pre-Covid London. Perhaps the most appreciated and notable act on his side was when he offered me the chance to do experience-worthy consultancy work in-between my PhD studies. My time under the guidance and mentoring of Dr Panagiotis Angeloudis helped to unravel directions for my career I never thought would have been possible before.

Besides my supervisor, I would like to thank Dr Marc Stettler, who I had the chance to work with and received valuable feedback on my publications throughout my PhD studies. Additionally, I could not leave out Dr Ali Niknejad, who was the first to teach me how to properly code in Object Oriented Programming and Dr Eleftherios Anastiasiades, the primary inspiration source and help for achieving my first journal publication. The list extends to other TSL laboratory members such as Dr Simon Hu and Dr Nils Goldbeck. I would also like to thank the departmental administrators Sarah Willis, Fionnuala Ni Dhonnabhain, Anna Hikel and Tina Mikellides for their continuous administrative support whenever needed. To mention the rest of the acknowledgements properly, I need to explain how the PhD journey started and where it ended.

My decision to pursue a PhD degree dates back in October 2014. Then, I was a fresh "out-of-the-box" engineer working for an engineering consultancy firm in the city of Leeds, having completed a Civil Engineering MEng degree at Imperial the previous summer. A "newbie" in the town, I spent a lot of time with my friend Demetra Flouri, then a PhD student on applied mathematics at Leeds University. My exposure to her efforts to achieve contributions in medical imaging was undoubtedly my first significant nudge towards research. Therefore, I am incredibly grateful that we did cross paths during that year; had we not, I would have been in a very different place now.

Nonetheless, it was not until the Imperial College graduation ceremony on the 22nd of October that year, where I decided to apply for my return to the South Kensington campus for a PhD. Upon hearing the news, my friends kept teasing me with the typical cliché of every returning Imperial student as a Stockholm syndrome sufferer. I insisted it was the Royal Albert Hall's awe-inspiring atmosphere that motivated me to do research, but perhaps I also liked Imperial College a bit too much.

However, returning to university for PhD studies after (even briefly) working in the industry was not an easy decision on paper. The switch was naturally accompanied by a significant decrease in salary and a substantial increase in accommodation costs for the next few years, the downside of studying in London. Naturally, I was considerably concerned over the viability of my plan: do a PhD at Imperial - achieve great things. Even so, I did make the switch. For that, I have to thank my parents, Maria and Panagiotis for pledging their full support behind my aspirations. Perhaps everything me and my sister Angelina will ever achieve in our lives is mostly down to their fierce determination to educate us the best way possible, both in skill and ethos.

In October 2015, I made my way back to London and Imperial College. Upon my return to the university, I was fortunate to reunite with people I have been close friends with since year 1 of my undergraduate studies. These include Nicholas Miscourides, Demetris Hadjigeorgiou, Constantinos Papayiannis and Kyriakos Nikiforou, who were also conducting their PhD studies at Imperial. The group was extended with the new PhD friends: Georgia Kouyialis, Christina Koutsoumba and Maria Grammatikopoulou to form the "lunch" gang.

Altogether, we held countless lunch and coffee breaks over the years of our PhD studies. During these breaks, typical discussions included (but were not restricted to) analysing our research thoughts or planning our weekend adventures. I am grateful to all them for being present during these years, as we all have been through the PhD struggle together, but we have had some fantastic moments in doing so.

My last year at the university was significantly different than the previous ones. All of my friends graduated having started their studies a year earlier, I made new friendships and experienced the Covid-19 pandemic. The "lunch" gang was reformed, with members of the Center for Transport Studies (CTS) and other PhD students, such as He-in Cheong, George Sialounas, Petrina Constantinou, Eduardo Candela, Alessandra Abeille, Jose Escribano, Clemence Le Cornec, Qiming Ye, Roger Teoh and Georgia Bateman. Altogether we continued socialising remotely during the difficult pandemic period. Special thanks go to my friends George and Petrina, who I have spent countless hours discussing research and personal matters with, and my good friend He-in who made my career as a quantitative researcher possible, by introducing me to Simudyne.

Outside the academic environment, from the start, I was blessed to have amazing friends that coloured the years of my studies with some beautiful memories, and have been, in part a mental push to keep me going and finish my degree. The list includes Eleni Christofides, Christoforos Chandriotis, Vasilis Kalogirou, George Christou, Charalambos Spanos, Pavlina Ellina, Nikoletta Kyranides and overlaps from previous mentions. The memories with them are countless, but perhaps highlights are the trip to New York and the vibrant Halloween parties we held in our house every year.

Looking back, my PhD was a beautiful journey, and I would do it again if I went back in time. I am grateful for everyone I crossed paths with during all these years as they played their own part in deciding where I am today.

Contents

List of Tables

List of Figures

Abbreviations

ABM	Agent-Based Model
AMoD	Autonomous Mobility on Demand
APX	Approximable
ATNC	Autonomous Transportation Network Company
AV	Autonomous Vehicle
BCMP	Baskett-Chandy-Muntz-Palacios
BC	Branch and Cut
CA	Cellular Automata
CDA	Combinatorial Double Auction
CMCF	Convex Minimum Cost Flow
DRS	Dynamic Ride-Sharing
DSGE	Dynamic Stochastic General Equilibrium
FCI	Fleet Coverage Index
FIFO	First-In-First-Out
GFP	Generalized First Price
ILP	Integer Linear Program
IP	Integer Programming
IPF	Iterative Proportional Fitting
IPO	Initial Public Offering
KPI	Key Performance Indicator
LP	Linear Programming
MILP	Mixed Integer Linear Programming
MIP	Mixed Integer Programming
MoD	Mobility on Demand
MWIS	Maximum Weighted Independent Set
NHS	National Healthcare Service
NP	Non-deterministic Polynomial time
NYC	New York City
OD	Origin-Destination
OOP	Object-Oriented Programming
SA	Simulated Annealing
SAEV	Shared Autonomous Electric Vehicle
SAV	Shared Autonomous Vehicle

TLC	Taxi and Limousine Commission
TNC	Transportation Network Company
TSI	Trade Surplus Index
UK	United Kingdom
VCG	Vickrey-Clarkes-Grove
WDP	Winner Determination Problem
WGC	Wild Goose Chase

Dedicated to my parents Maria and Panagiotis.

Chapter 1

Introduction

1.1 Landscape and Motivation

The recent proliferation of Transportation Network Companies (TNCs) has been facilitated by increasing demand for efficient, economic, and personalised urban mobility modes. TNCs have quickly captured a significant share of the urban mobility market, especially from their preceding taxi services and public transport, by providing a usually cheaper service than taxis, more convenient than public transport, and an effective alternative to private car ownership. Their success has been underpinned by the use of powerful algorithms and analytics, which helped reduce waiting times and increase fleet utilisation [3, 4, 5].

The standard TNC service, often referred to as ride-sourcing, is similar to taxis, since it also provides an on-demand, door-to-door transportation option to its customers. However, in contrast with traditional taxi street-hailing and dispatch centres; TNCs offer a more streamlined service via smartphone applications. Such smartphone applications act as platforms where TNCs match available drivers (often regarded as TNC partners instead of employees) with prospective customers who submit on-demand ride requests to the platform. In the setting described above, TNCs act as two-sided market makers, by setting a ride fare[1] and a driver commission rate[2] to attract two distinct sides of the transportation market, in their efforts to either maximise their revenues or social welfare.

The typical organisational structure of ride-sourcing firms precludes drivers' consideration as employees, thereby TNCs are not regarded as fleet owners. This organisational element implies that the capital requirement (or entry cost) for new TNCs to enter the market should be mainly dependent on the technology (i.e. software/product development). Consequently, the low asset cost (i.e. no vehicle purchasing required) for entering the market could hint for a competitive structure with many players (TNCs). Nonetheless, considering the local competition in urban markets across the world, one observes that it is usually described by a handful of players (oligopoly).

[1] Customers pay the ride fare to the drivers via the use of online payment frameworks.

[2] The platform collects a proportion of the ride fare, regarded as the commission rate.

The formation of these oligopolistic markets in ride-sourcing is better understood by dissecting the two-sided market structure which ride-sourcing companies implement. Essential characteristics of two-sided markets are the network effects that influence the interactions between parties [6]. According to [7], a network effect occurs when a service's value increases according to its user penetration. As an example, network effects in a TNC platform are present when the demand for rides is high, therefore incentivising the entry of drivers into the TNC market.

In essence, network effects induce the "chicken and egg" virtuous cycles described in economics, where more buyers (riders) in the platform incentivise more sellers (drivers) to join, and vice-versa. Consequently, to fuel such cycles, attract participants, and gain market share, new players in the TNC market initially invest considerable capital in subsidising both groups (drivers and customers) and incentivise them to use their platform [8]. Inevitably, this initial capital investment required to spark the friction between the two sides of the market, often in line with the local absence of regulation, is what induces oligopolistic TNC markets with cash-infused ride-sourcing companies.

Consequently, the few leading TNCs relishing the benefits of large urban markets report substantial revenues yearly. As an example, companies such as Uber, Didi Chuxing and Lyft, reported revenues of $20bn, $12.1bn and $2.7bn respectively in 2016 [9]. The absence of regulation for TNCs gives the freedom to manoeuvre with pricing strategies, such as lowering prices to gain higher transport market share. TNCs such as Uber, have been able to operate at a loss by raising cash via funding rounds and Initial Public Offerings (IPOs). The total reported funding of Uber was $24.2bn as of October 2018, which was raised over 21 funding rounds [10].

A significant driver behind the sizeable investment in ride-sourcing companies is the anticipated launch of autonomous vehicles (AVs) in their services [11]. An important proponent of this quest for autonomy, despite the substantial fleet purchasing costs, is that the redundancy of human drivers is expected to massively increase the profit margins for ride-sourcing platforms and induce economies of scale. In such a scenario, TNC platforms are likely to be transformed into one-sided markets, where they will enjoy complete control of the supply [12]. Research such as [13] and [14], anticipate that autonomy could also imply lower TNC prices for the public, thereby disrupting the current modal splits in urban transportation.

For reference, as outlined in the study in [15], the per-distance cost of a ride-sourcing trip is estimated to be significantly reduced if AVs are used in the service. Specifically, the authors estimate a cost of $2.728[3] per-kilometer for current ride-sourcing services, with 88% of the cost attributed to driver salaries. In the AV scenario, the authors estimate a cost of $0.407 per-kilometer with a significant increase in cleaning and maintenance costs. The study also estimates per-AV acquisition costs of $35000 for ride-sourcing platforms by assuming a commercial customer discount of 30% from car manufacturers due to substantial purchase volumes.

The digitisation of on-demand transportation mentioned above and anticipated deployment of AVs

[3] The authors in [15] used a 1:1 conversion rate between CHF and USD.

in TNC services to exploit economies of scale motivated a plethora of related studies in recent years [16]. Related research stretches in multiple fields, with studies aiming to optimise TNC revenue or public welfare, by introducing new methodologies and improving existing technology and operations. Furthermore, the ever-increasing urbanisation, which might be considered a proxy for congestion, and the ensuing consequence of global warming, act as amplifying motivators for TNC related research.

Researchers investigating ride-sourcing related problems have applied a wide range of scientific tools to reach conclusions in their studies. Studies which aim to improve the operational efficiency of TNCs made use of operational research approaches such as integer linear or non-linear programming and queuing theory. Furthermore, the increasing popularity of machine learning in recent years was not left out from TNC-related research, as many studies used practises such as reinforcement learning to identify best policies in time and meta-heuristics to find near-optimal solutions of computationally expensive problems. Additionally, pricing studies resorted to mathematical modelling approaches such as game theory or mechanism design to capture the interests of the various stakeholders involved in ride-sourcing systems. Finally, the dis-aggregated nature of trip planning and decision making in a complex system such as an urban transportation market, encouraged the use of Agent-Based Models (ABM) by many studies [16].

As most of the research on ride-sourcing operations, this study's grand motivator is to identify more efficient ways to offer this essential transportation service in the highly demanding and increasingly populated urban environments. To achieve the above objective, this study examines the existing literature on the operational aspects of TNC services in chapter 2 and identifies three niche research gaps that act as motivators for the research presented in the subsequent chapters. Specifically, the three pillars of motivation, as identified in the literature review in chapter 2 are:

I A lack of a modular framework describing the components and building sequence of ABMs for the different problems entangled in autonomous ride-sourcing, which is required to streamline such research.

II The omission of intelligent customer behavior when designing fleet management models and especially empty vehicle redistribution algorithms.

III The absence of an efficient methodology which applies assignment and pricing of shared rides in tandem at hours of peak travel, to reduce congestion and balance supply and demand of rides.

The lack of a modular ABM framework for autonomous ride-sourcing slows down the production of high-impact research in the area despite the extensive application of ABMs. In anticipation of labour-intensive software development to structure new models, practitioners and researchers might resort to inflexible ABM software packages and trivial methodologies. Alternatively, studies with sophisticated algorithms that utilize bespoke ABMs for testing, often lack clarity in their simulation structure, resulting in impediments when it comes to the reproducibility of their results.

Furthermore, the omission of intelligent customer behaviour when designing empty vehicle redistribution algorithms can overestimate the benefits of redistribution. Realistic ride-sourcing environments consider travellers choosing over a variety of similar transportation options. This effect of local competition, in combination with the notion of diminishing returns, implies a non-linear relationship between the amount of redistribution and its resulting benefit. Excluding these features in fleet management modelling can result in increased unjustifiable costs when compared to revenue.

Finally, when considering the assignment and pricing of shared rides at hours of peak travel demand, the points of interest in the modelling of such operations are to price rides to encourage sharing, and in doing so, match the shared trips and available vehicles efficiently. Combinatorial auctions provide a framework for deciding the above operations in tandem; however, the omission of trip detours in shared trips due to the added computational complexity can result in inefficient trip combinations. Such inefficient combinations of a vehicle and shared trips might imply increased detours or wait times for customers, or unfair pricing.

This study provides the simulation and mathematical modelling means to tackle the research gaps mentioned above. A central feature of this study is the gradual shift from strategic decisions of long-term impact, to tactical and operational decision problems of short-term online resolution. This feature equips the researchers, operators and regulators in the ride-sourcing domain with the perspective of the greater picture when pondering the questions of individual low-level problems such as the ones addressed in this study.

1.2 Contributions

To address the research gaps highlighted in the previous section, this study adopts a framework of problem labelling according to the relevant time horizon. Specifically, the study classifies problems according to a strategic, a tactical and an operational time-horizon (see Figure 1.1). This framework has been widely applied in research related to another shared mobility solution, namely that of car-sharing [1]. In a similar fashion, in this study, strategic problems attempt to answer decisions with long-term impact, such as fleet size (in the case of AVs) and infrastructure decisions (i.e. electric charging stations), tactical problems address medium term decisions such as fleet management and equilibrium static pricing strategies, and operational time-horizon problems address the shortest term decisions such as vehicle/rider assignments and dynamic pricing strategies.

1.2.1 Strategic Time Horizon Contributions

To address the literature gap outlined in motivation I, the study develops an agent-based modelling framework which identifies the core components of ABMs in autonomous ride-sourcing systems. The framework is modelled using the fundamentals of agent-based modelling and is presented alongside a proposed building sequence for structuring simulations of the various problems in autonomous ride-sourcing. To demonstrate the practicality of the ABM framework, the research in this study uses it

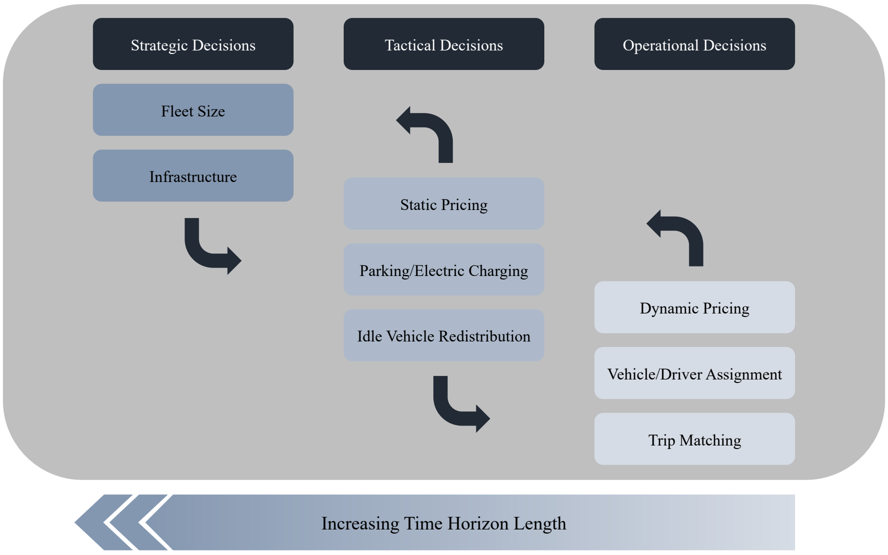


Figure 1.1: Conceptualization of the problem time-horizon labelling framework in ride-sourcing derived from the one consistently used in the car-sharing literature [1]. Arrows represent overlap of time-horizons.

to investigate the critical fleet size problem and compare the results with an aggregated model based on the theoretical properties of stable queues with multiple servers.

The contribution for motivation I is summarized as follows:

- The study identifies the core components of ABMs in autonomous ride-sourcing systems and presents them in clear-cut classifications based on the fundamentals of agent-based modelling.
- The study provides a model building sequence for tackling the different problems identified in autonomous ride-sourcing.
- An aggregated model is proposed which makes use of pickup wait times in identifying the minimum required fleet size for autonomous ride-sourcing fleets based on queuing theoretical implementations.
- A study performing simulations with an ABM is presented, which is structured using the proposed framework, to identify upper bounds of the critical fleet size and the dynamics of pickup wait times for the urban areas of Manhattan, San Francisco, Paris and Barcelona. The results are compared with the expected outputs of the aggregated model.

1.2.2 Tactical Time Horizon Contributions

To address the literature gap highlighted in motivation II, the study presents a mathematical programming formulation that accounts for alternative transport services offered and customer choice behaviour in the fleet management problem and specifically the empty vehicle redistribution problem. The derived vehicle redistribution problem is modelled as a non-linear minimum cost flow problem and the study proves that the model can have an optimal solution in a convex domain. The non-linear model is then transformed to the convex minimum cost flow problem and solved using an edge-splitting pseudo-polynomial algorithm.

The contribution for motivation II is summarised as follows:

- The study proposes a model which incorporates customer behaviour and local competition to account for the notion of diminishing returns in the vehicle redistribution problem.
- The study models the vehicle redistribution problem as a non-linear minimum cost flow problem.
- The work presents a derivation of a convex space for the problem and its transformation into a convex minimum cost flow problem, which is solved using a pseudo-polynomial edge splitting algorithm.

1.2.3 Operational Time Horizon Contributions

To bridge the research gap iterated in motivation III, the study presents a mathematical model that considers the effects of shared-ride detours through a winner determination process for the operational time horizon problems of peak-travel pricing and assignment. This implements a sealed-bid combinatorial double auction, with simultaneous driver-rider assignments that seek to maximise the total trade surplus. To reduce the problem search space, the work builds upon the concept of shareability networks [17], and transforms the formulation into a maximum weighted independent set problem, which is known to be APX-hard.

The contribution for motivation III is summarised as follows:

- The study proposes a winner determination problem modelled for dynamic ride-sharing assignment, implementing a combinatorial double auction while considering the effect of detours on the valuations of auction participants.
- A local search algorithm is provided which produces approximate results in polynomial time using greedy heuristic solutions as initializers.
- The effects of shill bidding on the proposed combinatorial double auction are identified, followed by a suggestion of a robust trip price determination methodology.

1.3 Thesis Structure

In accordance with the remarks made in section 1.1 and the contributions highlighted in section 1.2, the remaining chapters of this study are structured as follows: In chapter 2, the study presents a comprehensive literature review of the relevant theory and research on ride-sourcing operations, focusing on the applications of ABMs, the problems related to fleet management and idle vehicle redistribution, and the problem descriptions as well as solution approaches for ride pricing and vehicle/trip assignments in ride-sourcing.

Chapter 3 presents the ABM framework and methodology related to motivation I, as explained in 1.2.1. Chapter 4 showcases the fleet management approach inspired from motivation II, which forms the contribution highlighted in 1.2.2 above. Then, the methodological chapters conclude with chapter 5, which presents the peak-travel pricing and assignment methodology which forms the contribution outlined in 1.2.3 and inspired by motivation III. Finally the study concludes with an overview of the work and recommendations for future work in chapter 6.

Chapter 2

Literature Review

The areas of ride-sourcing and shared autonomous vehicle (SAV) operations encapsulate a diverse spectrum of disciplines which informs decision-making across the different levels of their services. As ride-sourcing companies aim to optimize their services in a competitive and disruptive transportation market, the scrutiny of their problems summons aspects of economics, mathematics, computer science and engineering. Wang and Yang [16] classify the problems related to ride-sourcing companies into four areas, namely:

- **Demand and pricing** - Patterns of demand and the design of analogous pricing schemes,
- **Supply and incentives** - Features of driver supply and the design of corresponding salary and incentive programs,
- **Platform operations** - Operational models to facilitate and enhance on-demand shared transportation services,
- **Competition, impact and regulations** - The impact of competition between multiple platforms on the urban transportation market and necessary regulation.

Similarly, Narayanan et al [18] in their comprehensive review for SAVs, reached to the following classification for the components of SAV modelling:

- **Demand** - Estimation of demand in the study area and mode share for SAV services,
- **Fleet** - Estimation of the required fleet size to serve a given demand,
- **Traffic assignment** - Route flows and travel times between origin and destination locations,
- **Vehicle assignment** - Methodologies for assignment of vehicles to customers,
- **Vehicle redistribution** - Redistribution of excess vehicles from low demand areas to to high demand areas,
- **Pricing** - Ride price estimation based on spatial and temporal features,
- **Charging** - Electric vehicle battery level monitoring and re-charging methodologies,
- **Parking** - Estimation of parking requirements and methodologies for online parking strategies.

There is a significant overlap between the current ride-sourcing platform problems and the ones related to SAVs. The majority of SAV modelling components outlined by Narayanan et al. [18] (other than demand) are sub-categories of the platform operations classification presented by Wang and Yang [16]. Furthermore, areas such as "demand and pricing" and "supply and incentives" might be listed as different problem categories. However, they are often encountered in problems related to ride-sourcing operations such as fleet management and vehicle redistribution, vehicles' assignment to customers, or the matching of trips for shared rides. Figure 2.1 outlines how the framework of relationships in ride-sourcing platforms and SAV operations is generally conceptualized in the literature outlined in this chapter. The positive and negative externalities in figure 2.1 pertain to an increasing and decreasing relationship between the variables respectively and also indicate causality.

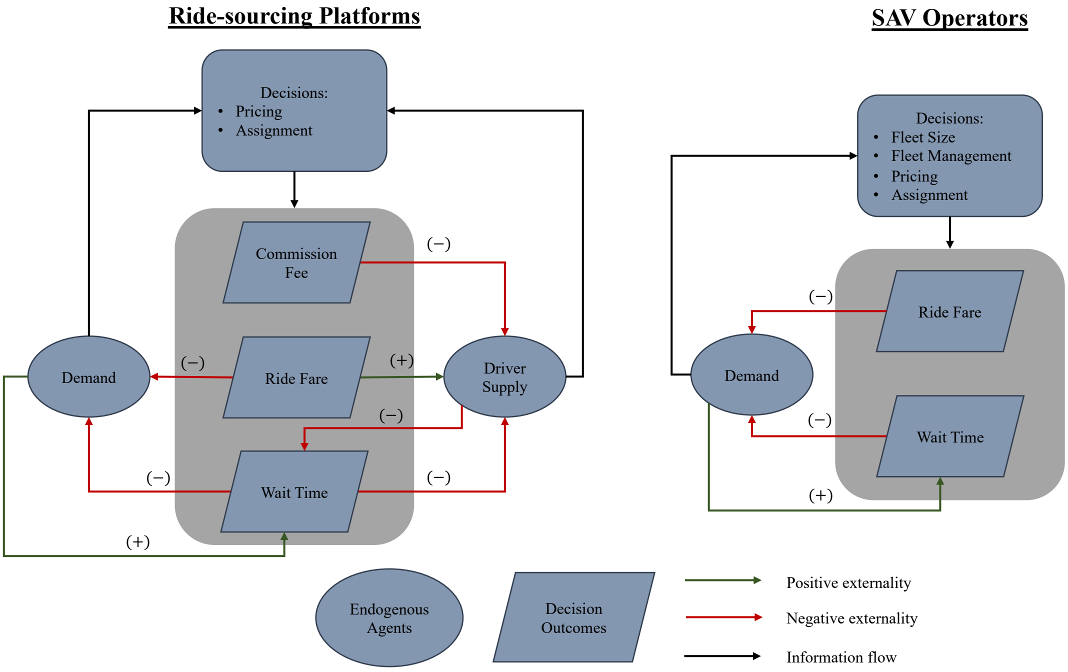


Figure 2.1: Conceptualization of relationships in ride-sourcing platforms and SAV operations identified across the literature.

This study focuses on practically implementable solutions to problems related to autonomous ride-sourcing operations and in identifying convenient and scalable means of simulating such services via ABMs. This chapter reviews the fundamentals of ABMs and their applications and the state-of-the-art on ride-sharing operations. Initially, section 2.1 covers the history, critical aspects and applications of ABMs in SAV platform design. Section 2.2 covers the areas of interest regarding SAV vehicle management. Then, in section 2.3, the chapter presents the principal strategies used for pricing rides in ride-sourcing platforms and SAV operations. The state-of-the-art on assignment operations such vehicle dispatching and trip matching is then presented in section 2.4, followed by a concluding summary in section 2.5 of the presented review across all sections and the identification of research gaps.

2.1 Agent-Based Modelling

The seminal work for agent-based modelling could be attributed to John von Neumann and Stanislaw Ulam when they created a grid-based model as a means for the design of the universal constructor; a machine capable of self-replication. Their grid-based model was comprised of a limited number of cells, each in one of a finite collection of states. The cells were assumed to alternate states in an iterative procedure, following a fixed rule. This type of discrete grid-based model was later termed cellular automata (CA) [19]. The concept of CA was subsequently incorporated in John Conway's Game of Life which is also one of the earliest applications of agent-based modelling [20].

What both Von Neumann's and Conway's experiments were showcasing, was that complex macro patterns could emerge by defining microscopic interaction rules between simple individual entities. These fundamental principles of CAs, extend to describe agent-based modelling, which, according to [21], is the practice of describing a system by defining a collection of self-determining agents. Unlike Conway's Game of Life; however, the agents are not necessarily restricted to simplistic behaviour, since the level of detail in the underlying system is entirely dependent upon the modeller. As such, ABMs constitute powerful tools for modelling highly complex systems to identify emergent phenomena by enabling heterogeneity in the agent population and non-trivial networks of communication.

2.1.1 Key Agent-Based Modelling Components

According to [22], ABMs have three key components:

- A set of agents.
- The network of connectedness between agents.
- The model's environment.

The agents are defined by their individual properties and their collective set of states. Behavioural definitions govern the alterations between states over time. Agent behaviour can range from fixed and simplistic (adhering to a finite set of rules), to adaptive and intelligent (perceptive with utility maximization). The authors in [22] also argue that agents need to be uniquely identifiable and modular. They explain modularity as the property which clearly defines a boundary on what constitutes an agent in a model.

The network of connectedness between agents is otherwise referred to as the topology of the model [21, 22]. It represents the mapping of possible social interactions. Each agent interacts with its neighbours, typically a subset of other agents in the model. The extent of neighbours (accessible agents), can vary according to the nature of the underlying topological model. In CA models such as the Game of Life, the topology is grid-based, with agents only connected to their immediate eight neighbouring cells. In models which consider a Euclidean topology, agents may only interact with others in their vicinity, up to a maximum Euclidean distanced radius. If the modeller considers a graph topology, agents (represented as nodes) are connected to their neighbours by graph edges.

The environment is used in order to represent the spatial and temporal dimensions of the model. It is, therefore, the boundary of the existence of everything within an ABM. As such, agents' locations in time are derived by the environment. Also, complex environments could be characterized by local properties, such as events or limited resources, which create restrictions and impact agents [23]. For example, in an ABM which simulates motor traffic, the environment would take the form of a road network. Consequently, road links would be subject to a limited capacity of vehicles, and an event such as a storm, might also reduce the utility of travel. Figure 2.2 illustrates the ABM components described in a road network environment.

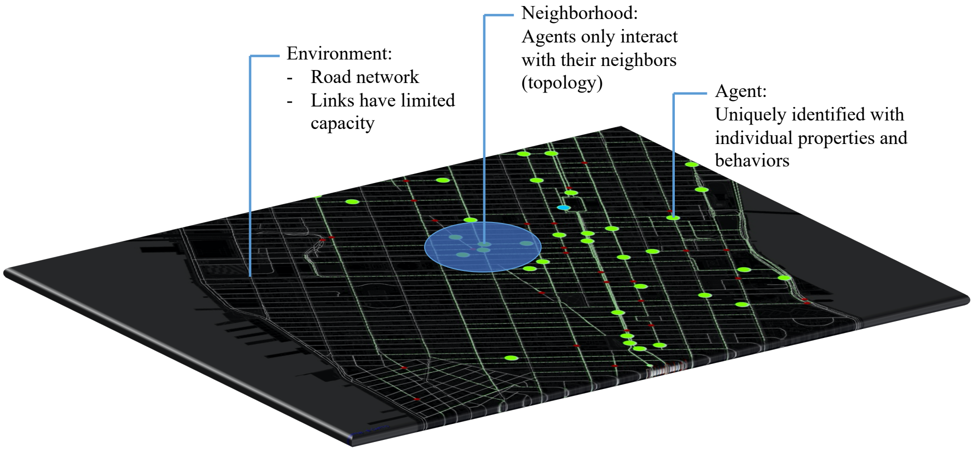

Environment:
- Road network
- Links have limited capacity
Neighborhood:
Agents only interact with their neighbors (topology)
Agent:
Uniquely identified with individual properties and behaviors

Figure 2.2: Illustration of ABM components in a road network environment.

2.1.2 Agent-Based Modelling Across Disciplines

The modularity and generality of the agent-based approach make it suitable for implementation in a variety of disciplines. Additionally, the substantial technological advancements in computing and the growing amount of data availability in the last decade has facilitated the emergence of numerous agent-based modelling projects, notably in areas such as transport, economics and epidemiology. In the following sub-sections, we offer an overview of some examples of agent-based methodologies applied in these areas.

Transport

Transportation planners are typically interested in evaluating the impact of new implementations, such as policy or infrastructure projects on transportation networks. These implementations could range from the introduction of new streets, congestion pricing or even public transit lines in existing networks. Traffic engineers traditionally derive key performance indicators of impact such as traffic flows, environmental footprint and cost-benefit ratios using aggregated four-step forecasting models[1]. Nonetheless, the emergence of intelligent transportation systems, as well as complex solutions such as intersection design, created the need for modelling tools which assess the impact of such systems from a dis-aggregated perspective [25].

Agent-based simulations have therefore been useful in complementing the traditional four-step models due to their decentralized nature. Software packages such as Vissim [26] allow micro-simulation of road networks where vehicles and pedestrians can act as agents in a realistic three-dimensional environment. The three-dimensional component of Vissim, however, prohibits the modelling of large scale networks due to computational limitations. Nevertheless, software such as TRANSIMS [27] and MATSim [28] can conduct agent-based simulations of city-scale models, by considering vehicles and travellers as agents, but omitting the low level three dimensional network detail of Vissim.

Economics

In macro-economics, traditional analysis utilised the dynamic stochastic general equilibrium (DSGE) model. In such models, economists aim to identify the effects of policy on growth and business cycles in economies. Their premise is to evaluate the evolution of econometric factors through time, assuming that the economy is subject to random shocks and also in Walrasian equilibrium [29, 30]. Such models have been criticised by various studies [31, 32, 33, 30] for invalid assumptions additional to equilibrium, such as population homogeneity and rationality[2].

The simplicity of the DSGE assumptions leads to their inability to describe highly non-linear effects which are results of emergent behaviour or wealth distribution in an economy. As a consequence, agent-based modelling in economics and finance is a useful tool to explore systems which are not in equilibrium and involve heterogeneous agents which act based on heuristics[3][35]. Some implementations of ABMs in finance consider the non-linear relationships of corporate bond trading in mutual funds [36] and the underlying agent interactions that govern the United Kingdom (UK) housing market[37].

Epidemiology

During the unfolding of the Covid-19 pandemic, especially in the early months of 2020, agent-based modelling was critical in policy implementation in the UK and the United States [38]. ABMs built in cooperation between engineers, mathematicians and epidemiologists, assisted in determining the footprint of the SARS-CoV-2 virus on the population. Studies such as the one by [39] influenced the

[1] Four stage models include trip generation, trip distribution, modal split and traffic assignment [24].

[2] DSGE models typically assume a representative agent who is entirely rational.

[3] In behavioural economics it has been argued that people use heuristics in decision making rather than exact optimisation[34].

UK government in implementing strict social distancing rules to prevent an anticipated sharp increase in hospitalization rates which would overwhelm the National Healthcare System (NHS). The input of such epidemiological ABMs has been data-heavy but also uncertain. Consequently, stochastic processes were incorporated in the models, and multiple runs of the simulations assured the consideration of numerous scenarios by the modellers.

The brief structure of models, such as the ones employed to inform governments, was comprised of agents clustered in individual households. Socio-economical characteristics from census studies were employed to characterize the properties of the agent population. Some of the most critical properties have been age, sex, health status, type of work and amount of contacts. Then, the modellers anticipated social mixing using data such as modes of travel and origin-destination matrices. The topology of these models was defined by agent geographic vicinity, as the transmission of the virus is only possible with the proximity of agents in the underlying environment. Additional data inputs, considered the biological description of the virus, such as the transmission rate and the viral impact on each agent [39, 40].

2.1.3 Agent-Based Modelling Applications in Ride-Sourcing

This section presents the breadth of ABM applications in AV ride-sourcing. To do so, the areas of application are split into the assignment problems of vehicle dispatching and ride-sharing, electric vehicle charging, fleet sizing, mode choice and pricing, and idle vehicle rebalancing. Table 2.1 summarizes the categorization of related literature on ABM applications for autonomous ride-sourcing.

Considering the dispatching problem, [41] used an ABM to evaluate different dispatching strategies to improve wait times and trip success rate of private trips. In terms of ride-sharing only, [42] used dynamic ride-sharing heuristics and a cell transmission model in an ABM to simulate the effects of ride-sharing on traffic flow. [43] applied an ABM to compare the impact between trip sharing in an Autonomous Mobility on Demand (AMoD) scheme to private vehicle ownership.

The concept of SAVs, the autonomous extension of the car-sharing industry, which resembles AV TNCs, was introduced by [44]. The authors used a bespoke ABM to evaluate the impact of empty vehicle rebalancing in an SAV fleet in Austin, Texas. Their work was extended in [45], where they applied MATSim, an open-source ABM traffic simulation software, to evaluate the effects of different rebalancing strategies on customer wait times and fleet costs for SAV fleets. [46] also used MATSim to identify optimal fleet sizes for different levels of demand for an SAV fleet.

The impact of vehicle pooling strategies on mode choices was investigated by [47], [48] and [49] using ABMs. Other studies, such as [50] and [51], attempted to identify the SAV mode penetration in the market using different pricing schemes. [50] used a bespoke ABM whereas [51] used MATSim. The work in [52] extended the study in [50], to investigate the operational costs for Shared Autonomous Electric Vehicle (SAEV) fleets and vehicle rebalancing.

The problem of identifying a cost-efficient fleet size was investigated in combination with other TNC/SAV in studies such as [53], [54] and [55] using ABMs. [53] employed a large-scale micro-simulation of SAVs in Berlin and Barcelona via MATSim, to evaluate the effectiveness of dispatching strategies on the required fleet size to serve the trip data-set used. [54] also used MATSim to identify the effectiveness of pricing strategies on mode choice and fleet size in an SAEV fleet. [55] used an ABM to evaluate the impact of the level of ride-sharing and vehicle redistribution strategies on the required fleet size for an SAV operation.

Studies focused on sophisticated vehicle rebalancing strategies such as queuing theoretical models and aggregated optimization methodologies resorted to bespoke simulation models for validation rather than open-source ABM software such as MATsim. [56] used an ABM to validate their queuing theoretical vehicle redistribution model whereas [57] and [58] validated a reinforcement learning redistribution approach using an ABM. Similarly, [59], [60] and [61] validated their aggregated optimization approach for vehicle redistribution using an ABM.

2.2 Fleet Management

Fleet management considers the allocation of resources to tasks. Specifically, in a ride-sourcing setting, the resource refers to discrete vehicles or continuous vehicle hours. For a ride-sourcing operator, the tasks at hand denote the various types of service or states which a vehicle can be used throughout its operation. From an economic perspective, fleet management aims for the allocation of vehicles which stimulates an efficient service. Consequently, monetizing the effectiveness of fleet management strategies naturally leads to minimum cost or maximum profit optimization problems.

The variants of fleet allocation decision making might refer to the numbers of vehicles allocated for service, or parked in catchments across a network of operation at different times throughout a day. The impact of allocation is usually derived by the expected spatio-temporal variation of demand in the network. Deciding vehicle count thresholds throughout network catchments implicitly refers to idle vehicle redistribution typically from low demand areas to high demand areas [18]. Effective redistribution strategies require an in-depth understanding of the demand characteristics so that accurate predictions of incoming requests can drive the redistribution decision making [62].

As ride-sourcing offers rides in a two-sided market platform format, where independent drivers subscribing to the platform are matched to travel requests, this prohibits a centralized mandate for idle redistribution of vehicles. Nonetheless, when ride-sourcing firms introduce AVs to their services and potentially transform to fleet owners, such idle vehicle redistribution strategies can be explicitly implementable via a central fleet manager.

Research on taxi economics, such as the work in [63], has been seminal in influencing the implementation of spatio-temporal characteristics when modelling taxi markets. The authors in [63] did assume the flow of taxis in neighbouring areas; however, the work focused on evaluating system performance metrics for regulatory frameworks. Later studies also considered optimizing redistribution in shared mobility schemes such as bike-sharing or car-sharing[4] fleets [64, 65, 66, 67].

2.2.1 Idle Vehicle Redistribution in Ride-Sourcing

The structural differences[5] between ride-sourcing and traditional vehicle sharing schemes prohibited the direct application of vehicle sharing related research in the ride-sourcing market. Vehicle redistribution for ride-sourcing/taxi markets became popular alongside the concept of autonomy. As a consequence, researchers investigated vehicle redistribution in simulation studies implementing simplified redistribution heuristics for SAVs [68, 42]. These studies were critical in identifying the extra mileage and congestion, respectively, when redistributing empty SAVs.

Recent research focused on identifying redistribution strategies for ride-sourcing operations using demand predictions and linear Integer Programming (IP), queuing theoretical models or machine learning methods to identify relocation strategies. In [56], a closed Jackson network was used to simulate an AMoD service with passenger loss. The authors then solved the vehicle rebalancing problem using Linear Programming (LP). Using a queuing-theoretical implementation, they showed that congestion effects due to rebalancing could be avoided. The authors in [69] replaced the Jackson network with a Baskett–Chandy–Muntz–Palacios (BCMP) queuing network model and considered vehicle charging operations.

The authors in [57] and [58] used reinforcement learning to identify rebalancing actions in a Mobility on Demand (MoD) scheme and ride-sourcing platform respectively and showed that their methods achieve effective rebalancing strategies. A fluid-based optimization problem on a queuing network was used in [70] to identify an optimal routing policy with an upper bound for empty car routing in ride-sharing systems. The study in [71] considered a Markov decision process for the problem of vacant taxi routing with e-hailing. The authors solved their model using an iterative algorithm to maximize the expected long-term profit over a working period.

The authors in [59] and [60] used predictive algorithms to estimate incoming requests and an IP model to assign idle vehicles to clustered regions. They tested their models in a simulation of the New York City (NYC) taxi dataset and significantly reduced waiting times. Model predictive control for vehicle redistribution was also utilized in [61] to identify short term estimations of customer demand. The model in [61] achieved a significant reduction in waiting times when tested in simulation using Didi data.

[4] For a comprehensive review of vehicle redistribution algorithms for car-sharing we refer readers to [1]

[5] In traditional shared mobility schemes vehicles are usually parked at designated stations, and dedicated staff performs the relocation. Also, vehicles are usually booked in advance.

2.3 Ride Pricing

Ride sourcing firms, exhibit a two-sided market structure, acting as platforms which connect online drivers with prospective riders. The properties of two-side markets were highlighted in the seminal paper in [72], where the authors investigated the competition between a couple of two-sided market platforms. In essence, the ride-sourcing firm's main deciding parameters are the fare which riders should pay, and the commission fee retracted from the drivers as a proportion of the ride fare.

The earlier studies on taxi economics, such as [73] and [63] were used to identify steady-state static pricing policies for the taxi market. However, the substantial advancements in mobile technology and the ease of entry in the ride-sourcing market have shaped a competitive environment with high elasticity of riders to prices over the past decade. In line with that development, drivers are flexible workers on the system, and their market entry is guided by expected earnings, their sensitivity to commission rates and trip fares [16]. The endogenous nature of both supply and demand in the market stipulated the need for supply-demand balancing mechanisms to be applied by the market makers.

Such a mechanism is dynamic pricing, usually implemented in the form of surge price multipliers. Dynamic pricing is implemented at periods of peak travel, in areas of the network where the volume of incoming requests is such that cannot be served by online drivers. Increased fares discourage a portion of the demand to request rides and encourage new drivers to join the platform, or online drivers to relocate to areas where surge pricing is applied, in expectation of higher gains.

Consequently, dynamic pricing curbs demand and boosts supply, steering the service away from effects such as the Wild Goose Chase (WGC) [74]. According to [74], WGC is an effect occurring at times of scarce supply, where the few drivers are matched to distant riders, thereby producing excessively long pickup wait times. These wait times destabilise the assignment process and discourage new drivers from entering the system, thereby abruptly suppressing the platform's trip throughput.

As many major TNCs consider to deploy AVs in the near future, their platforms are likely to be transformed into one-sided markets, where they will enjoy complete control of the supply [12]. Existing dynamic pricing methods suggest new equilibrium prices to customers without having prior knowledge of their trip valuations. If these are considered, market equilibrium prices could be identified without approximation, transforming this process into an auction. Previous work on ride pricing using auction theory [75, 76, 77, 2] focused on the interactions between riders (bidders) and drivers (sellers) who are expected to declare their valuations and costs for prospective rides.

A TNC platform, taking the role of the auctioneer, would be responsible for determining the winner of each auction [2]. Possible auction settings might involve one or multiple drivers that are assigned to customers sequentially or simultaneously. Manipulations of the auctions by either side can be avoided by using mechanism design theory, and the analysis of participation incentives.

Combinatorial Double Auctions (CDAs) [78] can be used to allocate multiple drivers to riders simultaneously and efficiently[6] using linear programs that are commonly referred to as Winner Determination Problems (WDP) [76], which are known to be NP-hard. WDPs can be formulated as set packing problems that maximise the auctioneer's revenue (or social welfare) while taking into account the utilities of the participants [79].

2.3.1 Dynamic Pricing

Studies assumed queuing theoretical models to investigate the properties of dynamic pricing in two-sided markets. Notably, through market equilibrium identification, the studies presented in this section identify dynamic pricing benefits at peak demand hours.

Initially, the authors in [80] used a queuing theoretical model to find that dynamic pricing is not as profitable as static pricing, however, they concluded dynamic pricing is more robust to changes in system parameters. The study in [81] found that a revenue-maximising strategy is aligned with welfare maximisation when driver supply is scarce compared to demand. The authors in [82] developed a practically implementable dynamic pricing methodology using surge multipliers by identifying an equilibrium between supply throughput and incoming requests.

Other studies, investigated the benefits of price and commission fare alterations and identified that a commission fare alteration strategy is beneficial to ride-sourcing platforms [83, 84, 85]. The authors in [83] study three pricing contracts, a fixed (static), a dynamic commission contract, a dynamic price contract and an optimal contract where the platform varies both the fare and the driver commission rate. They conclude that the commission contract yields nearly the same profit as the optimal contract in most scenarios.

The study in [84] identified the WGC effect when supply is in shortage. Their geometric matching model suggested that a revenue maximising platform increasing the customer fare during supply shortages is not necessarily beneficial to customers. They instead propose a commission rate strategy instead of increasing ride fares to combat supply-demand inefficiencies. The authors in [85] also use a queuing model to analyse the impact of dynamic pricing in a ride-sourcing platform. They find that a variable commission fee is necessary for maximising platform profit.

Pricing policy variation across the spatial domain was also scrutinised in studies such as [86, 87, 88, 89]. The authors in [86] assume strategic driver behaviour and design an incentive-compatible dynamic pricing mechanism for drivers accounting for spatio-temporal supply-demand characteristics. Besbes et al [87], formulate the spatial pricing problem, assuming drivers choose where to relocate under equilibrium in prices and transportation costs.

Guda and Supremanian [88] investigated the effects of varying surge prices across zones in a network on driver flows in-between zones. They concluded that optimal vehicle flows across zones can be managed by identifying all zonal surge price levels in coordination. Finally, the authors in [89] consider

[6] Economically efficient auction allocations maximise social welfare.

differential pricing across different locations in a network. They conclude that a platform is more profitable when the demand profile is balanced. They define a demand profile as balanced if the potential count of trips originating at each area is similar to the number of trips dropping off. To mediate the effects of demand imbalances, they find it is beneficial for the platform to employ spatial price discrimination.

2.3.2 Auction Mechanisms

Considering auction mechanisms in ride-sourcing, research in dynamic ride-sharing (DRS) (carpooling), is particularly relevant, with several studies exploring the applicability of auction models between commuting drivers and riders [90, 91, 75]. In [75], the authors propose a CDA discounted trade reduction mechanism for DRS assignment and pricing. The proposed mechanism is found to be incentive-compatible [7], individually rational[8] and weakly budget-balanced[9]. A system of parallel DRS auctions was proposed in [90] aiming to identify rider-driver matches that minimise detours. A DRS model using mechanism design was presented in [91], demonstrating that maximum social welfare cannot be feasibly reached while incentivising commuters' participation and truthful reporting of trip reservation prices.

Lam [76] models the allocation of AV seats to customers as a combinatorial auction, using the Vickrey-Clarkes-Grove (VCG) mechanism to sequentially assign customers to vehicles and determine prices. Three types of service are considered: private rides shared rides and requests split over multiple vehicles. A separate study developed a CDA model for dial-a-ride AV fleets [2] where multiple customers and AV operators submit bids, while a platform determines allocations that maximise social welfare. The model is applied for three types of service as in [76], with prices computed using a relaxed version of the problem with Lagrangian multipliers. The algorithm is shown to be NP-hard, but optimal solutions can still be obtained for realistic problem instances in reasonable times. Another proposed technique [93, 77] involves a truthful DRS mechanism based on a second-price auction with reserve prices.

2.4 Vehicle and Trip Matching

Matching in ride-sourcing refers to either the assignment of vehicles to trips or the mixing of trips to form shared rides (trip matching). The assignment of vehicles to trip requests is a decision problem which ride-sourcing inherited from the preceding taxi market, known as the taxi dispatch problem. The trip matching problem has its roots in DRS, a form of peer-to-peer ride-sharing where both travellers and drivers have their own travel destinations.

[7] Incentive-compatible mechanisms ensure that every participant is incentivised to be truthful.

[8] Individual rationality ensures that no participant incurs a loss. [92]

[9] Weakly budget-balanced mechanisms ensure that auctioneer will not incur a loss [91]

The main objectives of both problems are to identify the assignments which improve fleet utilisation (proportion of occupied vehicle time) and the quality of service (reduce customer wait times or detours). Both vehicle dispatching and trip matching problems are combinatorial and usually represented either in simple heuristics or IP formulations. Furthermore, as requests usually are serviced on-demand in the ride-sourcing market, the scrutiny of matching algorithms is focused on the trade-offs between efficiency and computational tractability [16].

Both vehicle matching and the simplest versions of trip matching (i.e. assigning trips to occupied vehicles) are associated with the linear assignment problem. The linear assignment problem is a fundamental combinatorial optimisation problem. In its most general form, it is stated as the problem of assigning n workers to m tasks to minimise total cost, where each assignment is associated with a linear cost, each worker performs at most one task, and a task is performed by at most one worker. If the number of workers equals the number of tasks, the assignment problem is balanced; otherwise, it is referred to as unbalanced [94].

Therefore, the assignment problem is equivalent to that of finding a maximum matching in a bipartite graph after specific transformations. The unbalanced assignment problem transforms into the balanced assignment one in a bipartite setting. The transformation is achievable by adding new vertices to the vertex set with the lower cardinality and zero-cost edges until both vertex sets have the same number of nodes. The balanced assignment problem when assuming a set I of workers and a set J of tasks, is formulated as follows:

Linear Assignment Problem:

$$\text{minimize} \quad \sum_{(ij)\in I\times J} w_{i,j} x_{i,j} \tag{2.1a}$$

$$\text{subject to}$$

$$\sum_{j\in J} x_{i,j} = 1 \qquad \forall i \in I, \tag{2.1b}$$

$$\sum_{i\in I} x_{i,j} = 1 \qquad \forall j \in J, \tag{2.1c}$$

$$x_{i,j} \in \{0,1\} \quad \forall i \in I, j \in J \tag{2.1d}$$

The parameter $w_{i,j}$ in the objective function (2.1a) represents the weight (or cost) of assigning worker i to task j and variable $x_{i,j}$ is equal to one if worker i is assigned to task j and zero otherwise. The Hungarian algorithm solves the linear assignment problem in strongly polynomial time in the number of workers n [95]. Furthermore, under straight-forward transformations, the linear assignment problem can be transformed to the minimum cost flow problem, which can be solved efficiently using the network simplex algorithm. Figure 2.3, outlines how the vehicle matching problem can be conceptualized as a linear assignment problem via a bipartite graph. Edge weights in figure 2.3 might represent wait times or travel distance, profit from an assignment or other representations depending on the model.

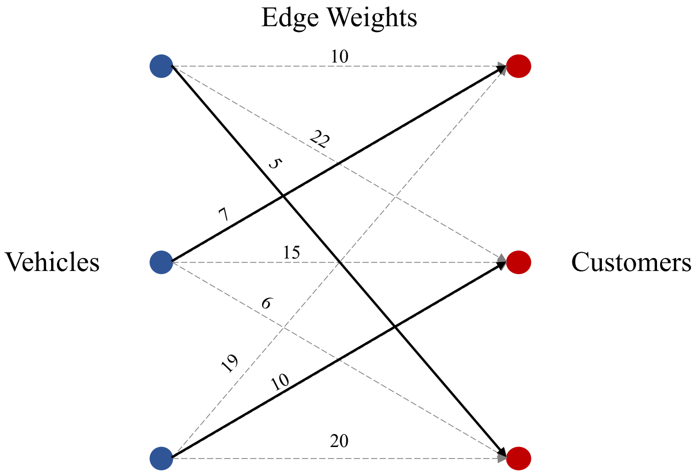


Figure 2.3: Conceptualization of the vehicle matching problem as a bipartite minimum cost matching problem.

2.4.1 Vehicle Dispatching

Vehicle dispatching in practice has been traditionally tackled using simple heuristics such as sequential assignments of customers to their closest available vehicles in terms of geographic proximity. In its simplest form, an operator or a ride-sourcing platform populates a list of trip requests upon submission and assigns each customer in the order of arrival. Such a heuristic is otherwise referred to as rule-based, due to the assignment process decided by a simple rule (i.e. closest vehicle, closest client). Rule-based heuristics have been generally preferred due to their simplicity and their ability to generate quick solutions. Studies applying rule-based heuristics include [52, 55, 96, 49, 97].

More sophisticated approaches solve vehicle dispatching in quasi-online fashion [98, 99, 100, 101]. Specifically, requests and available drivers are accumulated over a specified period, after which a mathematical program is solved to obtain multiple assignments at once. Modelling methods for the quasi-online vehicle dispatching problem frequently use network theory by transforming the assignment problem in a bipartite network. A critical decision upon implementing such algorithms is the length of the request/driver accumulation period. Such methods show improved performance when compared to rule-based (greedy) approaches.

Dickerson et al [98] model the vehicle dispatching problem as an online bipartite matching problem. They consider customer arrivals and vehicle availability in multiple time-steps using a look-ahead strategy. They solve their model using a tractable LP-based adaptive algorithm. Bertsimas et al [99] propose a maximum flow formulation to solve the vehicle dispatching problem in a quasi-online fashion. Their algorithm constructs an assignment network (arcs represent assignment) with random pickup times, which is solved offline multiple times. A new network is then populated with arcs from the maximum flow problem solutions, which is then re-optimised to obtain assignments in 15-second intervals.

Xu et al [100] use reinforcement learning to evaluate expected rewards of assignment decisions. They then solve the vehicle dispatching problem for multiple drivers and requests using combinatorial optimisation, maximising the expected rewards. Lyu et al [101] introduce a tractable online multi-objective maximisation optimisation problem for matching drivers to customers, considering pickup-distance, revenue and driver rating. They adjust each objective's weights based on the performance difference between the offline and online matching problems.

2.4.2 Trip Matching

Trip matching has similar properties with the vehicle matching problem as both are different versions of the assignment problem. In trip matching, two or more requests with different origin and destination locations are matched together forming a shared ride. In a shared ride, the vehicle will pick-up and drop-off matched customers according to an efficient schedule defined by the operator. As a consequence of the overlap in the service times of matched trips, the nominal variable which defines the quality of a shared ride is the detour experienced by each customer in the ride. A detour is defined across the literature as the deviation (in distance or time), of the shared trip experience from a private trip with the same origin and destination.

Similarly to vehicle dispatching, trip matching can be solved by either greedy heuristics or sophisticated optimization algorithms in a quasi-online fashion. However, in contrast to vehicle dispatching, trip matching is not necessarily a bipartite (two-dimensional) assignment. For example, if the matching problem aims to identify shared trip combinations of three (or more) customers, the two-dimensional assignment model is no longer valid. Instead, a tripartite (3D) matching model is the appropriate convention for identifying combinations of three trips. Higher customer counts in trip combinations increase the dimensionality of assignment accordingly. Inevitably, increasing dimensionality negatively impacts the computational tractability of optimization problems.

The above computational complexity considerations were addressed in [17]. The authors of the study modelled quasi-online matching of trip requests for different orders of matching. They did so by initially introducing the notion of the shareability network. In a shareability network, vertices account for unassigned individual trip requests, whereas edges between vertices denote their potential compatibility for matching. By calculating the most efficient detour for each potential match and removing edges which violate defined compatibility constraints, they show that 2D trip matching can be reduced to the maximum weighted matching problem, which can be solved efficiently in polynomial time. They also show that 3D matching is NP-complete and APX-complete, thereby it can only be solved quickly using greedy heuristics, and that any higher-order (4D or more) assignment is computationally intractable.

Studies such as [102, 103, 104] and [105] opted for sequential assignment instead of quasi online to achieve quicker computation times. Hosni et al [102] presented a Lagrangian decomposition approach to solve a Mixed Integer Programming (MIP) problem for matching seeking customers to vehicles with on-board customers. They also presented an incremental cost heuristic for matching each incoming customer to the vehicle, which incurs the minimum incremental cost if assigned. Ma et al [103] proposed a greedy trip matching strategy where incoming requests are matched to vehicles which are empty or serving on-board customers. Their method matches each request to the vehicle, which incurs the minimum increase in travel distance if assigned.

Peltzer et al [104] outlined a road-network partitioning algorithm to split the ride-sharing instance into mutually exclusive areas. They then used a greedy matching heuristic which assigns incoming requests to occupied vehicles sequentially by minimizing the total distance driven. Simonetto et al [105] assigned customers sequentially to vehicles using an LP approach. They also use a trip insertion heuristic to identify optimal schedules for each vehicle, fed into a local neighbourhood search heuristic for further improvement.

To improve the optimality of solutions compared to an offline assignment, studies such as [106, 107] and [108] used batch matching approaches, by performing assignments in regular intervals. Jung et al [106] used a simulated annealing method to simultaneously match multiple incoming requests to vehicles in a quasi-online fashion. Their quasi-online matching method improves the performance when compared to greedy nearest vehicle or trip insertion heuristics. To reduce the computational burden on their simulated annealing process, they introduce checks that ensure the random walks of their algorithm are only performed on feasible solutions.

Alonso Mora et al [107] applied the shareability networks proposed in [17] to reduce the instance space of the problem. They then employed greedy matching to assign batches of requests to available vehicles and improve the obtained solution by iteratively solving a constrained linear program. Qian et al [108] incorporated customer utilities and driver revenue thresholds in the matching process. They then formulate the problem as an Integer Linear Program (ILP) and convert it to an equivalent graph problem. Due to the NP-hard complexity of graph problem, they solve it using greedy heuristics.

Alternative approaches include the study in [82], where the authors optimized the matching interval and matching radius to improve ride-sharing efficiency. Also, in [109], the authors used spatio-temporal demand predictions as input to a trip matching problem for minimizing expected costs. The authors utilized a trip insertion heuristic after proving their demand away route planning problem is NP-hard. Finally, in [110], the authors narrowed the search space using shareability graphs and incorporated a discrete choice model so that the optimization for the matching result takes into account the customer utility. Bench-marking against time-window constraint approaches showed that their method yielded more attractive solutions for customers.

2.5 Summary

As observed in section 2.1 and outlined in table 2.1, researchers applied ABMs extensively to investigate the impacts of AV TNC operations. Nonetheless, in anticipation of time-consuming software development, many studies resort to using ABM software packages that are inflexible when it comes to modelling non-trivial operational strategies. By contrast, studies that apply bespoke ABMs to test sophisticated optimisation techniques often lack clarity in their simulation structure, which hinders reproducibility of their research and could potentially lead to significant omissions in the implementation. Therefore, it is evident that the existing state-of-the-art on AV TNC simulations lacks a modular framework describing the components and building sequence of such ABMs for the different problems entangled in autonomous ride-sourcing which is required to streamline such research.

Table 2.1: Summary of related literature on ABM applications for autonomous ride-sourcing.

ABM Studies	Dispatching, Sharing	Charging	Fleet Size	Mode Choice, Pricing	Rebalancing
Fagnant and Kockelman [44] Fagnant et al. [45] Zhang and Pavone [56] Wen et al. [57] Alonso-Mora et al. [59] Lin et al. [58] Wallar et al. [60] Iglesias et al. [61]					✓
Boesch et al. [46]			✓		
Shen and Lopes [41] Levin et al. [42] Dia and Javanshour [43]	✓				
Martinez and Crist [47] Martinez and Viegas [48] Hörl [49]	✓			✓	
Chen and Kockelman [50] Maciejewski and Bischoff [51]				✓	
Chen et al. [52]		✓			✓
Maciejewski et al. [53]	✓		✓		
Liu et al. [54]			✓	✓	
Fagnant and Kockelman [55]	✓		✓		✓

Additionally, when looking at the fleet operations literature outline in section 2.2, it is observed that the majority of relevant studies on vehicle redistribution (Table 2.2) do not consider acumen in customer behaviour. The two main approaches are, assuming unassigned customers abort the platform immediately (passenger loss) or setting a maximum customer wait time (maximum wait), after which all customers abort the service. However, in realistic ride-sourcing implementations, potential customers would encounter alternative travel options based on local competition, thereby deciding their mode choice based on several factors [111]. Consequently, oversimplified models of customer behaviour or nonexistent representation of competition do not reflect the real value and cost of redistribution, nor do they account for the notion of diminishing returns, since vehicle relocations do not necessarily result in guaranteed customers.

Table 2.2: Relevant studies on vehicle redistribution

Study	Method	Redistribution Cost	Customer Behavior
Zhang and Pavone [56], Iglesias et al. [69]	Queuing theory	Congestion	Passenger loss
Wen et al. [57]	Reinforcement Learning	Distance-based	Passenger loss
Lin et al. [58]	Reinforcement Learning	Fuel-based	Not-specified
Braverman et al. [70]	Queuing theory	Time-based	Passenger loss
Yu et al. [71]	Markov decision process	Time-based	Not-specified
Alonso-Mora et al. [59]	Demand prediction, integer programming	Passenger delay	Maximum wait
Wallar et al. [60]	Demand prediction, integer programming	Time-based	Maximum wait
Iglesias et al. [61]	Demand prediction, integer programming	Time and Distance-based	No passenger loss

Finally, when reviewing the literature on pricing and assignment in sections 2.3 and 2.4, one progressively manifests that they are, at the very least, implicitly connected operations. Furthermore, the relationship between pricing and assignment in ride-sourcing operations alters to a more explicit one when considering peak demand periods. As deduced in section 2.3, dynamic pricing is essential in periods of excessive request arrivals. Also, as observed in section 2.4, during such peak travel times, ride-sharing, which is more efficient if decided in batches, greatly relieves the supply burden. In hindsight, a method that combines all the aspects of pricing and assignment mentioned above is combinatorial auctions.

The majority of studies on ride-sharing auctions (Table 2.3) use two-dimensional models that perform one-to-one assignments between buyers and sellers. Nonetheless, DRS outputs inherently consist of one-to-many assignments for trips that contain at least three participants (one driver, two riders), whose trip-time utilities are interdependent. This limitation was partially addressed by previous studies [76, 2] which, however, did not consider detour effects. An alternative approach [77] utilised sequential rider-vehicle matches, but without accounting for the effect of detours on valuations, assignments and pricing estimates.

Table 2.3: Auction studies on ride-sharing assignment

Study	Problem	Auction	Assignment	Detours
Zhang et al. [75]	DRS	CDA	one-to-one	No
Kleiner et al. [90]	DRS	Vickrey	one-to-one	Yes
Zhao et al. [91]	DRS	CDA	one-to-one	Yes
Lam [76]	DARP	VCG	one-to-many	No
Yu et al. [2]	DARP	CDA	one-to-many	No
Ashgari and Shahabi [77], Ashgari et al. [93]	DARP	VCG	one-to-one	Yes

Chapter 3

An Agent-Based Modelling Framework for Autonomous Ride-Sourcing

3.1 Introduction

The introduction of AVs in TNC services could transform TNC platforms into fleet operators [12]. Consequently, the typical list of decisions for a TNC, which includes assignment and pricing of private or shared rides, would be extended with fleet management functions, such as empty vehicle redistribution to areas of excess demand, as well as electric charging and parking operations. As TNCs are interested in running efficient services, the multitude of their decision problems is governed by mathematical models, usually in the form of optimisation problems. Furthermore, TNC mathematical models are either structured as aggregated representations of the system or consider specific sub-problem instances in the fleet operation [18].

Developers of TNC platforms are usually interested in system-wide Key Performance Indicators (KPIs) to quantify the impact of their models. These KPIs might not always be identifiable from the mathematical models. Some examples of such metrics include the velocities of vehicles across areas in the network (congestion), the average wait or detour times of travellers, fleet utilisation, or the level of sharing in a pooling service. Due to the complexity of the TNC operations, non-linearity of some relationships and heterogeneity of the client population are not always accounted in models. As a result, mathematical models, are not always capable of capturing the highly complex dynamics of the TNC environment and are subject to further validation.

To complement the scrutiny of their algorithms and capture higher-order effects, modellers resort to digital twins or simulations, using Agent-Based Models (ABMs). Through the use of ABMs, researchers can apply their algorithms in a realistic environment, observe and quantify the system's emergent behaviour in time. ABMs are bottom-up models, which simulate the behaviour of actors (clients, vehicles, operators) in the system by defining them as software objects (agents), which interact according to behaviours set by the modeller. As a result, they are dis-aggregated representations of reality, which allow for heterogeneous characteristics between agents, and are capable of simulating highly complex and non-linear systems [22].

An example of an AV TNC problem which requires the use of ABMs is critical fleet sizing. Critical fleet sizing refers to the minimum fleet size required for the rate of incoming requests to be sustainably serviced. Specifically, suppose incoming requests join a queue until an available vehicle is assigned to them. In that case, fleet sizes below the critical, contribute to increasing queue lengths with incoming requests, resulting in spiking assignment wait times for customers. This problem has been traditionally tackled in research by using queuing theoretical models with multiple servers by using parameters such as the rate of request arrivals and the average service time. However, the underlying network structure, the variation of velocities throughout the network and the spatial distribution of origin-destination locations contribute to non-homogeneous service times, making aggregation a challenging task. Furthermore, as the pickup time after assignment varies with fleet size, it results in an additional non-linear component in the service time. Consequently, the more reliable route for estimating critical fleet sizes in TNC systems is the use of simulations that capture the complexities mentioned above.

As identified in the relevant literature in chapter 2, a framework describing the structure ABMs for the different problems encountered in autonomous ride-sourcing is imperative to streamline such research. The ABM framework introduced in this chapter is structured to facilitate a guided approach in creating ABMs for the relevant problems in autonomous ride-sourcing highlighted in Section 2.1.3. In this chapter, initially, all the framework's core components are identified in line with the fundamental description provided in Section 2.1.1. Then, a model building sequence is presented in Section 3.3. The study initially introduces an aggregated model of the critical fleet size problem based on the theoretical properties of stable queues with multiple servers in 3.4. Subsequently, the study compares the results between the aggregated model and a simulation structured using the ABM framework to demonstrate its practicality in Section 3.5.

3.2 Framework Components

Following the terminology in the work of [22], the framework composition is split between the components of agents, their respective topologies and the environment in which they exist.

3.2.1 Types of Agents

The research outlined in Section 2.1.1 was used to derive the core agents in an AV TNC system. In doing so, three main types of agents were identified; the traveller, the vehicle and the operator. It is assumed that each type of agent is characterised by some distinct properties and behaviours. However, to ensure uniqueness and modularity, all agents across all types have a unique identifier property and a list of states. The list of states as well as the mechanism description for alternating between states is termed as state logic for convenience.

Furthermore, a list of KPIs for each agent is identified which could serve as a model output for each agent. Table 3.1 presents these core agents, their properties, behaviours and KPIs. Properties, behaviours and KPIs in the framework are non-exhaustive and also non-binding for model building but represent the typical ones encountered in the literature.

Table 3.1: Description of core agents' properties, behaviours and key performance indicators (KPIs).

Agent	Properties	Behaviours	KPIs
Traveller	- Unique identifier - Origin and destination - Request time - Utility function - Reservation price	- Mode choice - State logic	- Utility - Wait time - Travel time - Detour - Cost - Trips shared
Vehicle	- Unique identifier - Location - Capacity - Velocity - Range and charge - Schedule - Revenue function - Cost function	- Charging station choice - Parking station choice - State logic	- Mileage - Revenue - Cost - Trips served - Service rate
Operator	- Unique identifier - Fleet (vehicles) - Assignment strategies - Pricing strategies - Routing algorithm	- Assignment/pricing choice - Fleet management strategy - State logic	- Mileage - Revenue - Cost - Trips served - Service rate

Travellers are potential clients who submit requests to the TNC for rides. Consequently, throughout the ride-sourcing literature, the terms of clients, requests or riders have been interchangeable [16]. Their goal is to travel from an origin location to a destination. As such, their origin and destination coordinates are static[1] properties. It is assumed that travellers enter the system at their static request time and exit after they are delivered at their destination or when they abort the TNC service by cancelling their request (if this is possible in the model). Multiple travellers could also be bundled into the same request, with the number of persons in the request to be a property description.

To assist the acumen of traveller agents (if required), a utility function for each individual is set, that could take inputs which differentiate the quality of a TNC option amongst other options in the model (if any). Consequently, it could have deterministic inputs such as the prospective wait time, travel time and service price. The capability to support heterogeneous populations in ABMs, allows for monetising parameters in the utility function, such as the value of travel time, wait time, service price

[1] Static properties do not change in the model.

or other inputs, to be specific to individual travellers. Modellers could also define other traveller specific properties such as a reservation price, which reflects the maximum amount a traveller is willing to pay for a travel option.

Vehicles represent the AVs in the system, and their goal is to serve travellers. They can have properties such as the total and current capacity, as well as the total and current range[2]. Since vehicles are self-moving agents, they are also characterised by velocity and location properties, which depend on the environment and vary during the simulation. It is also assumed that operators own the vehicles due to their autonomous nature. Consequently, their activity schedule and revenue function are both governed by the assignment and pricing strategies of their operator. The type of vehicle also defines its cost function, which could take as input the mileage and velocity fluctuations during the simulation.

Operators are agents which denote the AV fleet owners in the ABM. Depending on the model, there could be one or multiple operators in the system (monopolistic vs competitive scenarios). They dictate the assignment, pricing and routing methods which vehicles follow during the simulation. Operators are not explicitly regarded as agents in the literature outlined in Section 2.1.3; however modelling them as software entities allows for extensible properties which enable the simulation of scenarios that emulate adaptive behaviour by TNCs. Such scenarios could be the response to a congestion charge by a regulator agent or the fluctuation of pricing in competition.

As mentioned earlier, all agents in the framework are assumed to have an arbitrary set of states. These states alternate during the simulation for each agent depending on discrete events, processes (i.e. assignment heuristic) or decisions. The complexity of decision making can be varying. The majority of ABMs cited in Section 2.1.3 assume travellers always request and follow an assignment from a TNC unless some thresholds such as wait time, detour time or a reservation price are exceeded. Other studies [50, 54] assume more sophisticated models, in which each option's utility is an input to a discrete choice model. Discrete choice models have not been widely applied on AV TNC simulation studies, mainly due to the lack of necessary population information to validate such models.

Vehicle behaviours are governed by operator properties, such as assignment and pricing strategies, as well as the routing algorithm. Typically, a vehicle might be preliminarily assigned to a traveller by the operator using some assignment heuristic. The traveller then would either accept or abort the TNC service. Upon acceptance, the vehicle will follow a route to the traveller origin based on the routing algorithm specified by the operator. In cases where a vehicle needs to recharge or park, vehicles could decide which charging or parking station to visit using simple heuristics such as the nearest available station.

The ABM framework in this study does not intend to define the agents' behaviours but rather collectively classify them as sub-systems. This approach is followed to maintain a flexible and extensible nature in modelling. Nonetheless, in Table 3.2, some basic states for travellers and vehicles are provided based on the models presented in Section 2.1.3. Table 3.2 does not provide any example states for operators

[2] Potential mileage based on electric charge level.

due to the lack of applications in research. It is also noted that any vehicle states related to electric charging were not included, but these states could be implemented as extensions of the core state logic.

Table 3.2: Core state description for travellers and vehicles.

Agent	State	Description
Traveller	Waiting for assignment	The traveller has requested a ride but has not been assigned to a vehicle.
	Waiting for pick-up	A vehicle has been assigned to the traveller and the traveller waits to be picked up.
	In trip	The traveller is in the vehicle.
	Served	The traveller has been delivered to the destination location.
	Aborted	The traveller aborted the service before pick-up.
Vehicle	Idle	The vehicle does not have any tasks.
	Travelling to origin	The vehicle is travelling to a traveller's origin location.
	Loading	The traveller is boarding the vehicle at the origin location.
	Travelling to destination	The vehicle is travelling to the traveller's destination location.
	Unloading	The traveller is exiting the vehicle at the destination location.

3.2.2 Topologies

Topology has been previously defined in Section 2.1.1 as the infrastructure enabling interactions between agents. Possible topology settings were identified which are based on euclidean distance or graph structures; however, in this AV TNC modelling framework, multiple levels of topologies are proposed which are also more complex than their one-dimensional counterparts mentioned in Section 2.1.1.

The interactions between types of agents in a realistic TNC service are initially highlighted, to accentuate the underlying topologies in a TNC ABM. The typical process involves a traveller submitting a request to an operator. The operator, in turn, assigns a vehicle and informs the traveller of the expected pick-up time. If the traveller accepts the assignment, the vehicle is then instructed by the operator to pick-up the client. It is therefore apparent that the operator brokers any agent interaction. Travellers do not directly interact with nearby vehicles, nor do vehicles decide which clients to select, but the operators instead instruct them.

It is also evident that different topologies are relevant at different times for different agents. On request submission, travellers can access all TNC operators. During the assignment, the operator structures a graph topology which outlines all possible assignments between travellers and vehicles (and also travellers and travellers in the case of ride-sharing). The assignment graph topology could depend on traveller requirements, vehicle properties, as well as the geographical vicinity between these agents. Furthermore, the aforementioned topology includes both travellers and vehicles, but it is only actionable by the operator. Upon preliminary assignment, each operator informs the travellers of their option, and then if travellers accept the offer, the operator instructs the vehicles to move to their assigned traveller origin locations.

The processes described above, highlight an open-access topology between travellers and operators, which is relevant at request submission, and upon traveller choice when offers are presented. A preliminary topology between vehicles and travellers is also in use during the assignment. The assignment topology is termed as preliminary, since it offers no access between vehicles and travellers, but is only visible to the operator. Derived from the preliminary topology, a one-to-one topology between travellers and their assigned vehicles is also identified after acceptance by the travellers.

Due to the technological nature of request submission to a TNC platform, no direct interaction between travellers and nearby vehicles is assumed, unless the operator decides an assignment. Nonetheless, such a simplified geographical topology could be implemented if modellers aim to test competition between TNC platforms and street hailing taxis. It also assumed that no direct interaction between the same types of agents takes place, such as traveller-traveller[3] or vehicle-vehicle[4] connectedness.

The implementations of assignment strategies in the research outlined in Section 2.1.3 vary from simple First-In-First-Out (FIFO) heuristics to aggregate optimization assignments. In a FIFO heuristic, travellers enter the end of a queue of unassigned travellers upon request submission. Then, the operator assigns the closest vehicle to each traveller sequentially, from first to last in the queue. Aggregate optimization assignment procedures were used mainly in studies proposing idle vehicle redistribution methodologies [59, 58, 60, 61]. Assignments using optimization, are decided in intervals in a quasi-online approach. During these intervals, travellers' requests and available vehicles accumulate a list, and a cost minimization optimization program decides the optimal assignment at the end of the interval.

To reduce the instance size of optimization problems for assignment, [17] proposed the use of shareability networks. Although the initial application was presented for use in identifying potentially shareable trips in a ride-sharing scheme, the concept can be extended to an assignment between travellers and vehicles for both shared and private trips. In such networks, nodes could represent traveller or vehicle agents, and edges between them could represent a capability for assignment subject to constraints. More sophisticated assignment strategies also incorporate pricing into the creation of such networks by using auctions [112, 113]. The shareability networks or more simplified techniques, such as FIFO, constitute the preliminary topology visible to the operator.

[3] Traveller-traveller connectedness could be apparent during a ride-sharing assignment process, but there is no communication between travellers.

[4] Vehicle-vehicle connectedness could be useful in the connected-autonomous vehicle (CAV) problem for traffic purposes but lies beyond the AV TNC scope.

3.2.3 Environment

The road-network is set as the core environment in this ABM framework. The presence of the operator is abstract (not physical); nonetheless, travellers and vehicles physically exist in a road-network via origin/destination and current location, respectively. Some simulations might ignore the urban road-network and instead use grid-like environments representing physical locations or even euclidean space. However, using these alternative environments might be an over-simplification as vehicle properties such as velocity and schedule are derived from road-network information. Furthermore, processes such as assignment, pricing and routing utilize road-network information via shortest path algorithms.

The main road-network components are road-nodes and road-links. It is assumed that road-nodes have the fundamental property of coordinates, and edges have the fundamental properties of an inbound and outbound road-node, as well as a geometry indicated by a list of coordinates. Information such as road-node neighbours, or the length of each road-link could be derived from the fundamental properties above. Additional to these properties (but not necessarily essential), could be a specific velocity or a velocity vs traffic density profile in each road-link. Simplified models could assume an average velocity over an entire road-network instead. Background traffic could also be implemented as a randomized input in a velocity-traffic density profile for realistic ABM implementations.

By assuming a velocity-traffic density profile for each edge, a modeller can define space in each link as a limited resource. Furthermore, additional environmental features on top of the core road-network structure could be capacitated parking or charging stations, situated in selected road-nodes across the road-network. Such stations also constitute a time-varying limited resource in an ABM. Additional environmental features which enhance complexity could include networks for different modes (public transport, cyclic, walking) or external events which impact the properties of the environment (i.e. extreme weather).

3.3 Model Building

The various components introduced in Section 3.2 serve as the raw materials required in ABM building for autonomous ride-sourcing problems. The amount of necessary detail in agents, topologies and environment composition could vary significantly depending on the ABM scope. Therefore, deciding the model contents as well as the sequence in which the building blocks should be created is essential in avoiding computational burden and over-design of the system.

First, it is highlighted that the various versions of agents, as well as the environment's building blocks, can contain information in the form of properties. Examples of properties for agents include the origin and destination locations of travellers and the locations and capacities of vehicles. Furthermore, environmental components such as road-nodes and charging stations have the property of coordinates. Agents also utilize behavioural processes, such as mode choice, charging/parking station choice, routing and assignment. Consequently, the preceding descriptions of agents and environment components conveniently fit the description of software objects with Object-Oriented Programming (OOP) principles.

In OOP, objects are software entities (classes) which hold properties and processes (methods). In that respect, OOP languages are regarded as ideal for building ABMs. Objects are classified into agents and non-agents since only agents have behavioural properties. Furthermore, processes are also categorized into decision problems and functions with scalar outputs. Functions with scalar outputs define scalar agent properties described in Table 3.1, such as traveller utility, vehicle revenue and cost, and serve as prerequisites to decision problems such as mode choice, routing and assignment. Table 3.3 summarizes the categorization of objects and processes in line with the descriptions presented in Section 3.2.

Table 3.3: Categorization of objects and processes.

Objects		Processes	
Agents	Non-Agents	Decision Problems	Scalar Functions
- Travellers - Vehicles - Operators	- Road-network - Road-node - Road-link - Infrastructure	- Mode choice - Routing - Assignment - Charging/Parking choice - Fleet management	- Utility - Revenue - Cost

The categories proposed above also reflect the objects' complexity and processes, as well as the proposed sequence for model building. Decision problems present higher complexity than functions outputting scalar variables; agents are more complex than non-agents due to their respective behavioural and adaptive properties. It is also apparent that decision problems need the outputs of scalar functions as inputs, and that agents have properties derived from non-agent objects (environment components). As a consequence, this model building sequence proceeds from simple to complex software entities. Figure 3.3 outlines how the different layers of complexity blend together to create a realistic ABM for ride-sourcing. We, therefore, propose the following sequence for model building, as shown in Figure 3.1.

The sequence of operations during a simulation run is also non-trivial. In practice, objects which exist throughout the simulation time should be instantiated before the iterator starts. As such, infrastructure objects (non-agents) should be instantiated first, followed by operators, and their vehicles in locations across the network (derived from infrastructure objects). Travellers could then be created during the iterations at their request times. Once new travellers are added to the system, a state logic step should be performed for each agent at each iteration step until all iterations are completed.

It is assumed that different models might require a different state logic step sequence when considering all types of agents. For example, at the beginning of an iteration step, a traveller requests a trip from an operator. The operator, in turn, might trigger the assignment process in the same iteration step. If at least a vehicle is available and the assignment process is not computationally expensive, it is reasonable to assume that the assignment would be instantaneous. In that case, the operator could report the

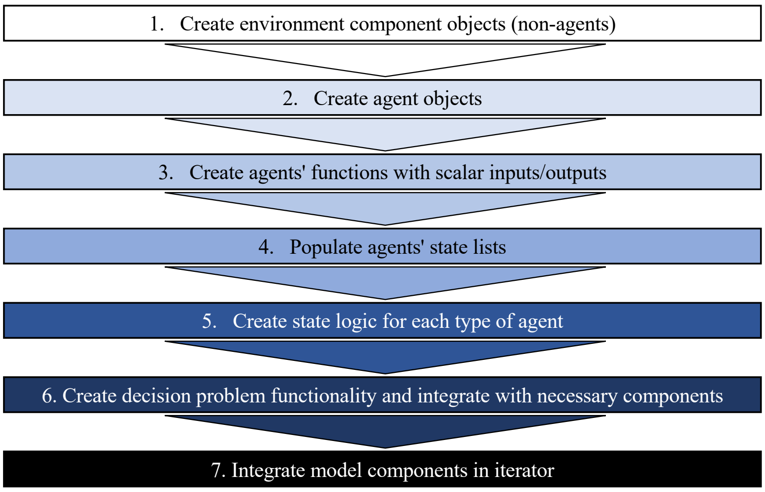

1. Create environment component objects (non-agents)
2. Create agent objects
3. Create agents' functions with scalar inputs/outputs
4. Populate agents' state lists
5. Create state logic for each type of agent
6. Create decision problem functionality and integrate with necessary components
7. Integrate model components in iterator

Figure 3.1: Proposed model building sequence.

assignment to the traveller, with the traveller choosing the preferred mode in the same iteration step as the request.

However, in cases where the assignment is indeed expensive computationally, such as in specific ride-sharing settings [113], the events of triggering the assignment process and obtaining a result do not occur at the same time even in realistic situations. During this period, from triggering a computationally expensive process to obtaining a result in a ride-sourcing environment, other operations might be underway. For example, vehicles not involved in that process might be travelling to a destination, and travellers involved in a similar assignment process do experience wait time which could be essential in their final choice. As a consequence, the use of multi-threading programming to run such computationally expensive processes in parallel with the main iteration might be required.

The proposed ABM framework offers the capability of creating simulations with complexity according to the appetite of the modeller, by following a modular approach. Figure 3.2 outlines the components of the proposed framework and its modular/extensible nature. Figure 3.3 outlines how the different layers of complexity blend together to create a realistic ABM for ride-sourcing.

3.4 Critical Fleet Size Modelling

Various studies used queuing theory to model the operation of ride-sourcing services and successfully focused on identifying optimal idle vehicle redistribution strategies across networks [56, 69, 70]. Such studies, model the arrival of travellers as a Poisson process in time, in which the travellers join a queue upon arrival and exit the system after they are served by vehicles. Service times are usually modelled by an Exponential distribution. The studies in [59] and [60] also focused on solving the empty vehicle redistribution problem, but instead applied aggregated optimization methods in combination with demand prediction to solve the problem.

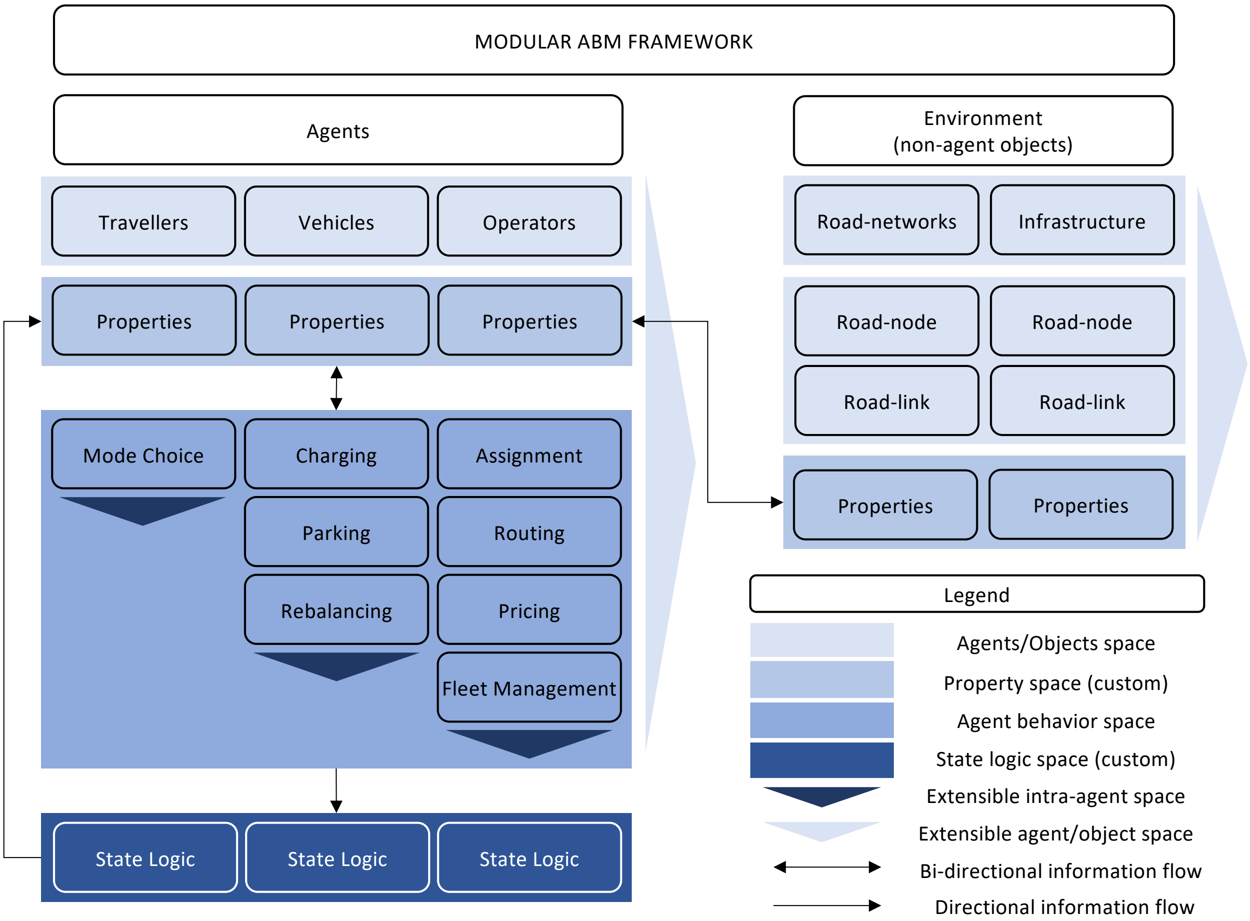


Figure 3.2: Graphical illustration of the modularity of the proposed ABM framework.

A key index of such vehicle redistribution studies is the average wait time which customers experience until pickup, which is traditionally regarded as a quality of service metric in demand-responsive transport [63]. Customer wait time is split in two categories, specifically, the waiting time from request to assignment, and the waiting from assignment to pickup. As such, assignment wait is governed by the number of vacant vehicles whereas waiting for pick up relates to the vicinity of those vacant vehicles. Consequently, understanding how the components of wait time vary with demand, vehicle availability and the spatio-temporal properties of the underlying network can be a useful input in fleet management models.

The model in this section assumes that customers request trips through a central ride-sourcing fleet operator at a quasi-constant rate. Requests are added to a single queue upon arrival and are assigned to the closest available autonomous vehicle in the fleet using a FIFO setting. For clarity of the demonstration of this ABM framework and identifying the properties of wait time variation, acumen in customer behaviour is not considered. Consequently, customers exit the system once they are served, even if they experience significant wait times. It is noted that although this setting is unrealistic, in this case, is useful in identifying the fleet size limitations.

An $M/M/c$ queue model is assumed as a way to represent the operation of the ride-sourcing fleet

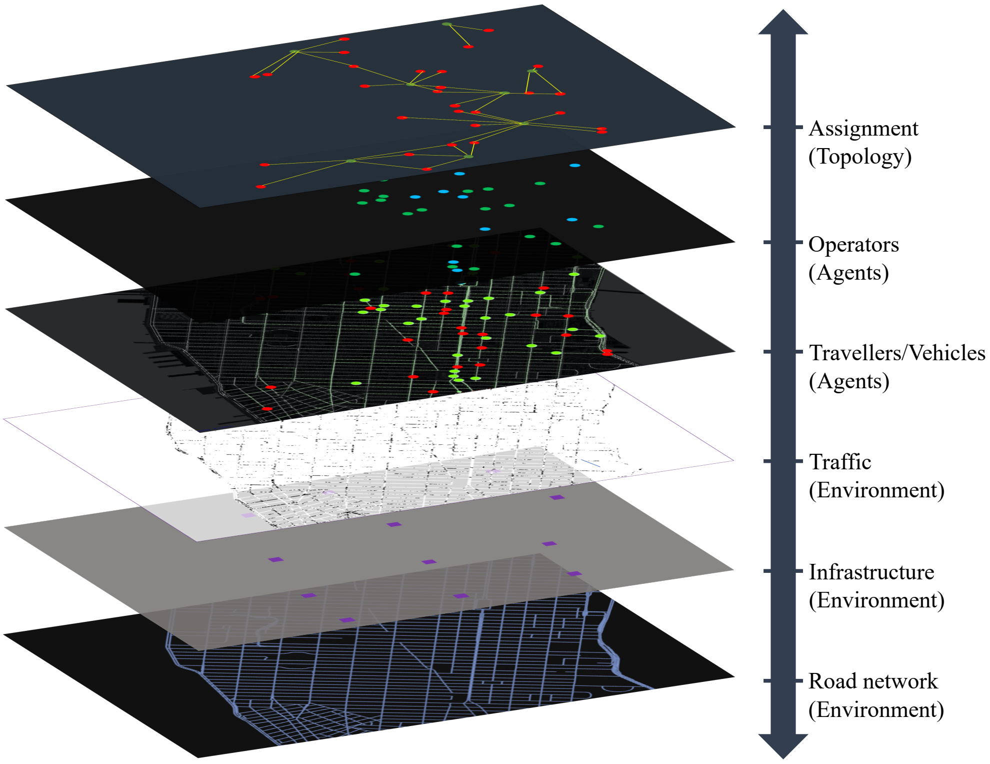


Figure 3.3: Illustration of the extensible simulation layers in the ABM framework.

operator. The first and second M's in the notation stand for Poisson generated customer arrivals and exponentially distributed service times, respectively. Furthermore, c denotes the number of parallel homogeneous servers (fleet size). As such, customer arrivals occur at a rate of λ per unit time and the average service time is set at $\bar{t}$ units of time. Consequently, the service rate μ is equal to $1/\bar{t}$. The utilization rate ρ of the fleet is therefore defined as follows:

$$\rho = \frac{\lambda}{c\mu} \tag{3.1}$$

As observed in (3.1), $c\mu$ denotes the number of customers the fleet can serve in a period. As such, for $\rho > 1$, the customer arrivals surpass the fleet's capacity, and the queue becomes unstable, with the average wait time in the queue W_q growing to infinity. Consequently, for $\rho = 1$ c is equal to the minimum fleet size c_0 which can sustain the queue, given λ and μ. Therefore:

$$c_0 = \frac{\lambda}{\mu} \tag{3.2}$$

The mean number of customers in the queue is defined as L_q (length of queue); similarly, the mean number of customers in the system (including customers being served) and the average time in the system are defined as L and W respectively. Using Little's law [114], the following relationships hold:

$$L = \lambda W \tag{3.3}$$

$$L_q = \lambda W_q \tag{3.4}$$

The following relationship between wait time in the queue the total time of customers in the system is also expected:

$$W = W_q + \frac{1}{\mu} \tag{3.5}$$

For $M/M/c$ queues the queue length can be found using the following [114]:

$$L_q = \frac{P_0(\frac{\lambda}{\mu})^c \rho}{c!(1-\rho)^2} \tag{3.6}$$

where P_0 in (3.6) is the probability of zero customers in the system, otherwise known as the Erlang C formula [115, 114]:

$$P_0 = \frac{1}{1 + (1-\rho)\frac{c!}{(c\rho)^c}\sum_{m=0}^{c-1}\frac{(c\rho)^m}{m!}} \tag{3.7}$$

As observed in equation (3.7), any terms including the number of agents c, such as the factorial and power terms produce extremely large numbers as c becomes large. As such, a transformation which assists the calculation of P_0 with large numbers is defined. To do so, function $f(c,\rho)$ is set as follows:

$$f(c,\rho) = \frac{c!}{(c\rho)^c}\sum_{m=0}^{c-1}\frac{(c\rho)^m}{m!} \tag{3.8}$$

It is noted that equation (3.8) contains terms of the cumulative distribution function (CDF) of a Poisson distribution. As such, equation (3.8) is transformed as follows:

$$f(c,\rho) = \frac{c!}{(c\rho)^c} e^{c\rho} F_{Poisson}(c-1, c\rho) \tag{3.9}$$

Where $F_{Poisson}(c-1, c\rho)$ in (3.9) is the CDF value of a Poisson distribution with mean $c\rho$ and cumulative probability $P(X \leq c-1)$. Using exponents of natural logarithms and the gamma function[5], $f(c,\rho)$ results in the following form:

[5] For a Gamma function $\Gamma(n)$, $\Gamma(n) = (n-1)!$.

$$f(c,\rho) = \exp\big[\ln(\Gamma(c+1)) - c\ln(c\rho) + c\rho\big]F_{Poisson}(c-1, c\rho) \tag{3.10}$$

Consequently the function $f(c,\rho)$ in equation (3.10) is used to transform function P_0 to a more convenient version for large values of c, as shown below:

$$P_0 = \frac{1}{1+(1-\rho)f(c,\rho)} \tag{3.11}$$

The average wait time in the queue W_q models the wait time until assignment. As such, the wait time from assignment to pickup is embedded in the service time. Nonetheless, more vehicles in the fleet would produce less pickup wait times on average. That is, assuming vehicles and requests are uniformly distributed in space. Consequently, by embedding the pickup wait time in the service time, the service time cannot be regarded as a constant (i.e. mean of an exponential distribution).

Instead, pickup wait time is modelled by exploiting the spatial characteristics of the underlying network. By assuming all available vehicles are uniformly distributed in space, the total network area A is divided by the number of idle vehicles V. As such, each vehicle has a coverage area of $\frac{A}{V}$. For a single idle vehicle with velocity v, the pickup time of an assigned request can be identified using $t_p = \frac{x}{v}$. Where x is the distance from the location of the available vehicle to the pickup spot.

Assuming the number of idle vehicles can vary, t_p can be defined using the average velocity in the network $\bar{v}$ and a derived average pickup distance. If an incoming request can be located in any of the coverage areas $\frac{A}{V}$ in the network, then the average distance between a request location and any available vehicle in the fleet needs to be defined. To do so, it is assumed that each vehicle covers a square area[6]. Since requests and available vehicles are uniformly distributed in the network, the average distance between any two points in a square coverage area $\frac{A}{V}$ needs to be identified. However, to account for non-uniform dispersion of vehicles across area A, factor $\psi \in (0,1]$ is introduced, such that the coverage area is transformed to $\frac{A}{\psi V}$.

Assuming the coverage areas to be squares, the average distance between two uniformly distributed points in a square domain is considered. [116] proved that the distance between any two such points in a square is approximately equal to $0.52\sqrt{a}$, where a is the square's side length. By considering a factor ϕ analogous to taxicab geometry, the average distance between a request and total available vehicles V in a network with area A is equal to $0.52\phi\sqrt{\frac{A}{\psi V}}$. The value of ϕ is identified by considering the average ratio of a network path over the haversine distance of between two road nodes. Consequently, the average pickup wait time $t_p(V)$ for requests is defined using the following equation:

$$t_p(V) = \frac{0.52\phi}{\bar{v}}\sqrt{\frac{A}{\psi V}} \quad \forall V > 0 \tag{3.12}$$

[6] This error of this approximation reduces with increasing vehicle availability.

The pickup wait time derived in equation (3.12) is in line with estimates derived in the works of [84] and [117]. Identifying the variation of V with respect to the fleet size c is a non-trivial task due to a variation of V with time after changes in parameters such as the rate of request arrivals or the average velocity during specific hours. To understand the nature of the fluctuation, the change of V on each frame in time can be inspected. By assuming a discrete-time step t and a set of time steps T, the number of idle vehicles V_t at time step t is governed by the following equation:

$$V_t = V_{t-1} + V_t^I - V_t^O \quad \forall t \in T \tag{3.13}$$

Where V_t^I and V_t^O in equation (3.13) represent the incoming idle vehicles and the newly outgoing/occupied vehicles respectively at time step t. The value of V_t^O can be regarded as the constant value of customer arrivals per time step t over the period T; however, the value of incoming idle vehicles depends, not only on the constant average trip time but also on the value of the average pickup wait time. Furthermore, V_t^I at time step t is decided by the average pickup wait time and by extension the number of idle vehicles of previous time steps t'.

As an example, consider a per-minute time step t, where the average trip time is 10 minutes, and the average pickup wait time at t_0 is 2 minutes. With these values, all the outgoing vehicles at t_0 would be added to the incoming idle vehicles of time step $t_0 + 12$. If the pickup wait time at $t_0 + 1$ is reduced to 1 minute with the trip time remaining constant, then the outgoing vehicles $V_{t_0+1}^O$ would also be added to the incoming idle vehicles of $t_0 + 12$. Considering this complex behaviour, the value of minimum fleet size c_0 could be identified, either by using non-linear integer programming or by assuming a continuous representation of idle vehicles V and time t. In both scenarios, one would seek to identify the lowest value of V_0 (which can be set to c), for which equation (3.13) is still feasible across all time steps t with V_t^O assumed to be constant.

As a consequence of this model, the minimum required fleet size c_0 is larger if pickup wait time is accounted. Furthermore, since the form of P_0 depends on ρ, changes on the structure of utilization ρ must be reflected on equations (3.6) and (3.7) as well. Nonetheless, the derivation of such formulas is non-trivial and beyond the scope of this ABM framework. However, it is evident in section 3.5 that due to the size of fleets in practical implementations, the $M/M/c$ queue model converges to $M/M/\infty$, which implies no waiting in the queue (assignment wait).

Since the value of c_0 is governed by the value V_0, it is expected that for c_0, $V(t)$ would initially drop to a minimum, and the reach a steady-state value as formerly occupied vehicles start returning from trips. Consequently, the value of idle vehicles V in equation (3.12) for pickup wait time $t_p(V)$ could be greater than V_t^O at the critical fleet size case if the steady state is above the minimum. However, for $c < c_0$, it is expected that the value of $t_p(V)$ will be equal to the maximum, with V equal to V_t^O. This would imply a possible discontinuity in the measurement of pickup wait times t_p and utilization ρ for $c < c_0$ and $c \geq c_0$.

By assuming the service rate introduced in equation (3.1) takes into account the steady-state pickup wait time $\bar{t}_p$, μ could be expressed using the following equation:

$$\mu = \frac{1}{\bar{t} + \bar{t}_p} \tag{3.14}$$

As such, the fleet utilization rate ρ in equation (3.1) transforms to the following equation:

$$\rho = \frac{\lambda}{c}(\bar{t} + \bar{t}_p) \tag{3.15}$$

The discontinuity in pickup wait times t_p at c_0, prohibits the identification of c_0 by setting the utilization ρ in equation (3.15) equal to 1.

3.5 Discussion

In section 3.4, the function of a ride-sourcing fleet was modelled as a queue of multiple servers to identify the critical size of the fleet. However, by analysing the effects of the variation of pickup wait times with idle vehicles in time, it was deduced that identifying a minimum value for fleet size is an increasingly complex problem. As such, the ABM framework introduced in Sections 3.2 and 3.3 is used to identify the minimum fleet size for ride-sourcing operations under specific parameters.

3.5.1 Model Structure

An ABM with the three basic types of agents highlighted in Table 3.1 is assumed. For the traveller and vehicle agents, the basic stage logic outlined in Table 3.2 are used, but the Aborted state for the traveller agents is omitted. Instead, it is assumed that travellers never abort the service so that the relationships highlighted in Section 3.4 are adequately captured. A single active state for the operator agent is also defined. Therefore, a simplified model is used, where travellers submit requests for private trips at the operator at their request times and are assigned to their closest idle vehicle using a FIFO assignment routine. Travellers exit the system once they are served by a vehicle. The operator also routes vehicles through the network using the A^* algorithm.

To identify a value for minimum fleet size relevant to the models in Section 3.4, complex behaviours such as mode choice, charging and parking station choice, pricing and fleet management strategies are ignored. The inclusion of a mode choice model would defeat the purpose of identifying unstable queues as customers would always choose a different mode if they are subjected to long wait times. Electric charging behaviour is expected to slightly increase the minimum fleet size during a simulation hour due to more unavailable vehicles, but that is solely dependent on the choice of battery and the velocity fluctuation. Furthermore, it is assumed a parking selection and fleet management model would be necessary only if the minimum required fleet size was known in advance.

Algorithm 1 Simulation Initialization

```
Create road-network object $G$.
$A \leftarrow \emptyset$                              ▷ Empty set of agent types
$A \leftarrow A \cup O$                               ▷ Add operator agent
$C \leftarrow \emptyset$                              ▷ Empty set of vehicles
$R \leftarrow \emptyset$                              ▷ Empty set of travellers
$A \leftarrow A \cup R$                               ▷ Add traveller set to agent types
for $i \leftarrow 1, c$ do
    $C \leftarrow C \cup 1$                           ▷ Add $c$ vehicle agents.
end for
$A \leftarrow A \cup C$                               ▷ Add vehicle set to agent types.
Initiate non-empty set of time steps $T$
Outputs: $G, T, A = (O, R, C)$
```

Algorithm 2 Simulation Process

```
Inputs: $G, T, A = (O, R, C)$
for $t \in T$ do
    $R \leftarrow R \cup r(t)$                        ▷ Generate new travellers at time $t$.
    for $a \in A$ do                                  ▷ For each set of agent types.
        for $a_i \in a$ do                            ▷ For each agent in the agent type set.
            $LogicStep(a_i)$                          ▷ Progress agent logic.
        end for
    end for
end for
```

To initiate and perform the simulation, algorithms 1 and 2 are respectively followed. In algorithm 1, the process instantiating non-agent (road-network) and agent objects in the simulator are followed. It is noted that this process does not refer to the model/code building procedure described in Section 3.3 but refers to the prerequisite step of initiating instances of all objects required for the simulation to run. During initialization, the road network G, the set of agent types A and the set of simulation time steps T are created. It is also noted that the set of travellers R in A is an empty set at the end of the initialization, as travellers are generated during the simulation time.

In algorithm 2, any new traveller requests are added at the start of each time step t of the simulation. Then, the simulator iterates through all the agent sets to progress the state logic of each agent. The FIFO assignment is performed as part of the logic step of the operator, whereas the A^* routing algorithm is performed by any vehicle which enters a travelling state such as Travelling to origin or Travelling to destination (Table 3.2). Furthermore, agent KPIs and properties are updated at each time step of the simulation. To achieve an acceptable precision in the representation of this agent-based model, the time step size t was set to be one second.

3.5.2 Model Instances

To test the methodology presented in Section 3.4, four urban areas were selected to create case study networks; namely the island of Manhattan in NYC, and the areas within the city boundaries of San Francisco, Paris and Barcelona. The underlying road networks and link travel times were obtained using the OSMnx library [118]. Endogenous congestion was omitted in the model by assuming a small proportion of traffic accounts to ride-sourcing vehicles. However, a 20% reduction was applied to the free-flow speeds in residential and motorway link segments, and 40% elsewhere, to account for exogenous congestion. Figure 3.4 shows the extends and the geometries of the road-networks used for the analysis.

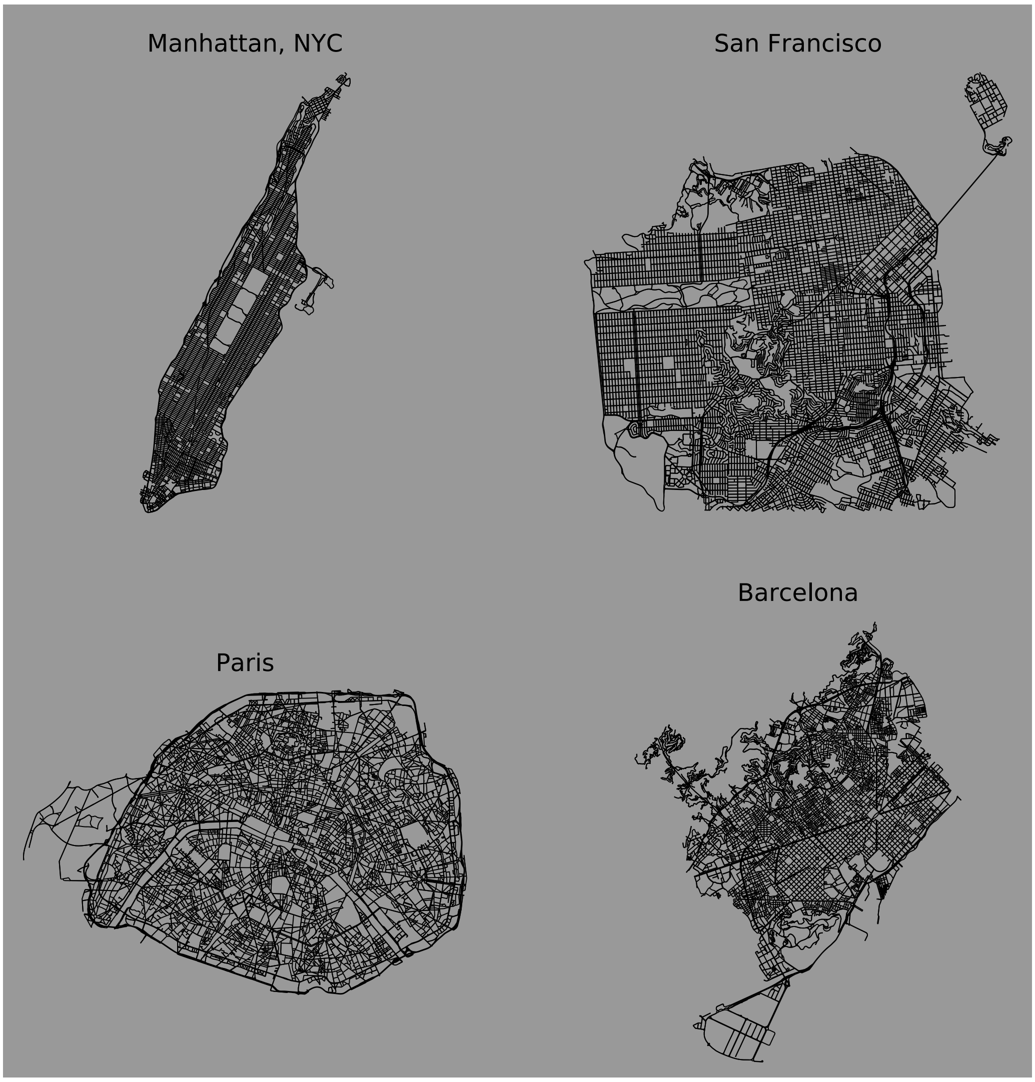


Figure 3.4: Road network geometries for urban areas used in the analysis.

Using the dataset provided by [119] and the [120], typical demand profiles were created for morning peak hours in Manhattan and San Francisco, respectively. While the trip dataset for Manhattan already included trip times, the San Francisco dataset only included inbound and outbound zonal trip counts. As such, the Iterative Proportional Fitting (IPF) algorithm [121] was used to generate a synthetic zone

Table 3.4: Descriptions of urban area instances used in the analysis.

Instance	A	ϕ	$\bar{v}$	λ	$\bar{t}$
	$[km^2]$		$[km/h]$	[/h]	[h]
Manhattan	59.1	1.36	24.5	11607	0.11
San Francisco	121.4	1.30	25.4	9454	0.20
Paris	105.4	1.31	23.2	10000	0.27
Barcelona	101.9	1.41	24.5	7000	0.28

Origin-Destination (OD) matrix for San Francisco.

Using the zone polygons provided by [120], the San Francisco road-network and the resulting OD matrix, random OD nodes were sampled within the zone polygons for each OD zone pair. In the absence of ride-sourcing trip datasets for the cities of Paris and Barcelona, origin and destination node pairs were randomly sampled for 10000 and 7000 trips per hour respectively. To achieve a realistic dispersion of the trips in time for each city other than Manhattan[7], inter-arrival times were sampled using an exponential distribution for three hours. Table 3.4 outlines the properties of model instances used to test the simulation.

3.5.3 Analysis

To identify the critical fleet size for each instance, multiple simulations of the ride-sourcing service were performed, varying the fleet size in each simulation run. In doing so, the length of the FIFO list on each time step was recorded, which is interpreted as the queue length introduced in Section 3.4. Any instances with fleet sizes that produce departures from a constant queue length are regarded as unstable and below the critical fleet size.

Observing figure 3.3 for each modelled city and instance, it is deduced as expected, that if the fleet size is greater than the critical fleet size, the average queue length has the same constant value for all fleet sizes. This constant value is equal to the number of travel requests per second, which implies that there is no wait in the queue and travellers are assigned to vehicles immediately.

The zero queue wait for stable queues is a direct implication of the volume of requests per hour λ. Revisiting equation (3.13), one can see that even for $V_{t-1} = 0$, due to constant non-zero traveller arrivals per time step t, the value of V_t^I must be at least equal to the traveller arrivals per second, for stability. The required abundance of idle vehicles is also expected when using equations (3.1)-(3.11) by assuming a converged low-value pickup the pickup-wait time. This directly implies that the $M/M/c$ queue model converges to an $M/M/\infty$ model for $\rho \geq 1$.

To further scrutinize the shift from unstable to stable queues, the pickup wait times of each served traveller in the simulations were also recorded as shown in Figure 3.6. To achieve steady-state pickup

[7] Trip times for Manhattan are available in the dataset [119]

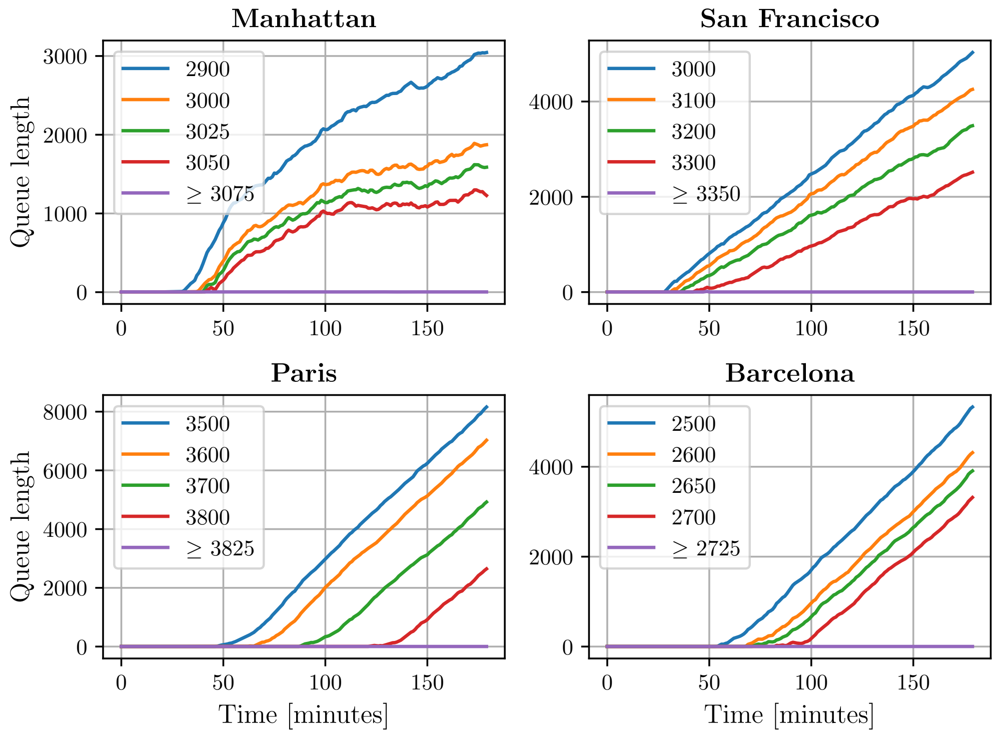


Figure 3.5: Queue lengths for each instance over the simulation period.

wait times in all cases, only pickup wait times in the hours after the queue becomes unstable are recorded (see Figure 3.5). Table 3.5 provides estimates of maximum pickup wait times using Equation (3.12) and the values from Table 3.4 for $\psi = 1$ (homogeneous dispersion). The number of idle vehicles V in each case is set, as an estimate, to the value of incoming requests per second, transformed from λ in Table 3.4.

In that respect, it is observed in Table 3.5, that the maximum simulation pickup wait times for unstable queues are not always closely approximated by the pickup wait time model in equation (3.12). Inaccuracies in the calculation might be down to origin locations dispersion (for Manhattan and San Francisco) and shape irregularities (network areas are not square). Nonetheless, with appropriate calibration of parameters to minimize deviation from observed behaviour, the model can serve as a reasonable approximation in aggregate optimization models.

Using the average trip times and pickup wait times of each instance, the fleet utilization was calculated as shown in equation (3.15). Similarly to the wait times, in figure 3.7, in all models, a jump in values from unstable to stable queues is identified. This discontinuity in pickup wait times and fleet utilization is expected and attributed to the fluctuation of idle vehicles in the simulation, as explained in Section 3.4.

Table 3.5: Maximum pickup wait time values using approximation and simulation.

Instance	V	t_p (Eq(3.12)) [min]	t_p (Simulation) [min]	Absolute Error [%]
Manhattan	3.28	7.35	8.43	12.8
San Francisco	2.63	10.85	10.71	1.3
Paris	2.78	10.84	14.15	23.4
Barcelona	1.94	13.01	15.75	14.4

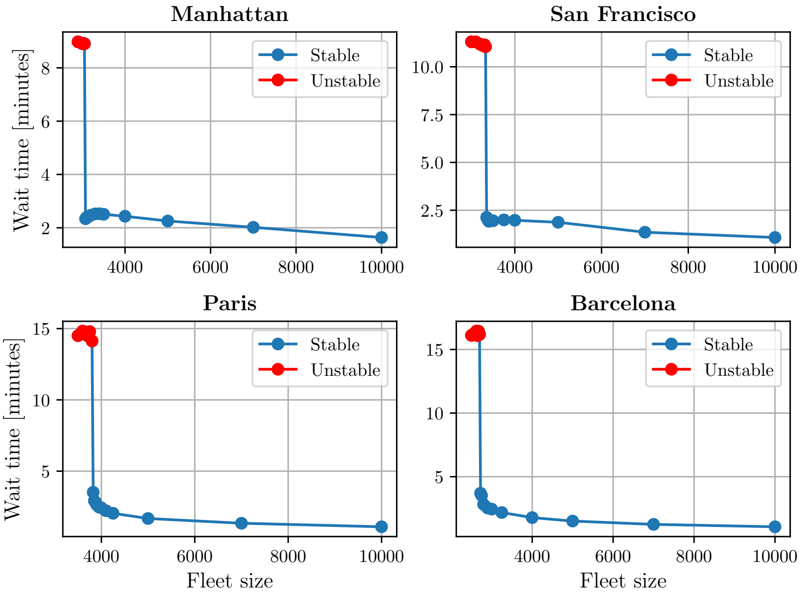


Figure 3.6: Pickup wait time variation for each instance.

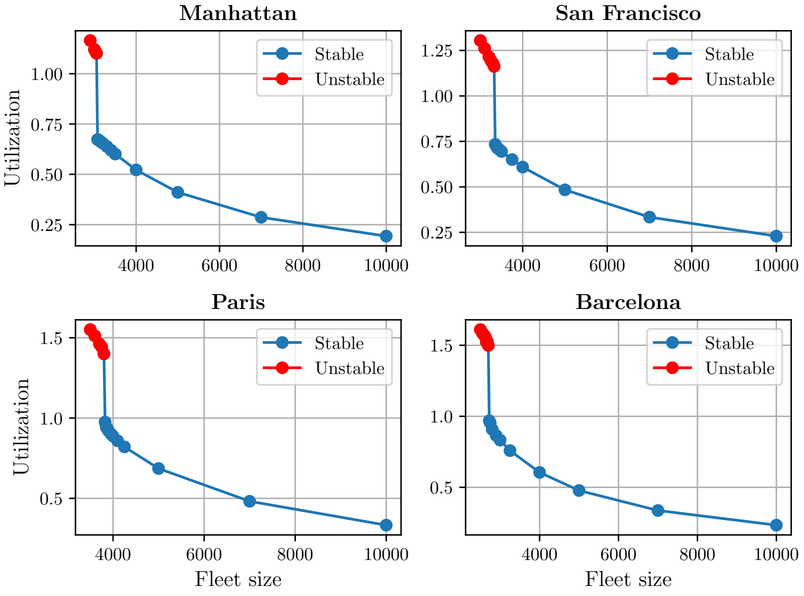


Figure 3.7: Utilization variation for each instance.

In figures 3.8 and 3.9, the fluctuation of idle vehicles and pickup wait times experienced by travellers is outlined for cases of unstable and a case of a stable queue respectively for all cities. It is observed that in the case of unstable queues (figure 3.8), the value of idle vehicles decreases at an exponential rate and never recovers after reaching the minimum. When observing figure 3.9, for the case of the stable queue, as expected, after an initial dip, the idle vehicles stabilize to a constant rate. The opposite trend is identified for pickup wait time, with the wait time reaching a maximum as the idle vehicles decrease and thereby also reaching a steady state in line with a constant flow of idle vehicles. The above observations are more prominent the instances of Manhattan and San Francisco.

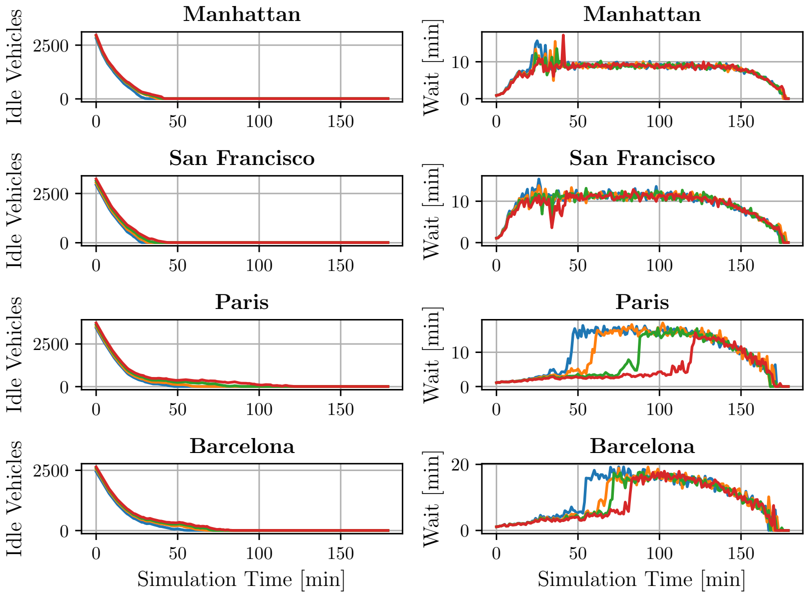


Figure 3.8: Variation of idle vehicle counts and pickup wait time over unstable queue settings.

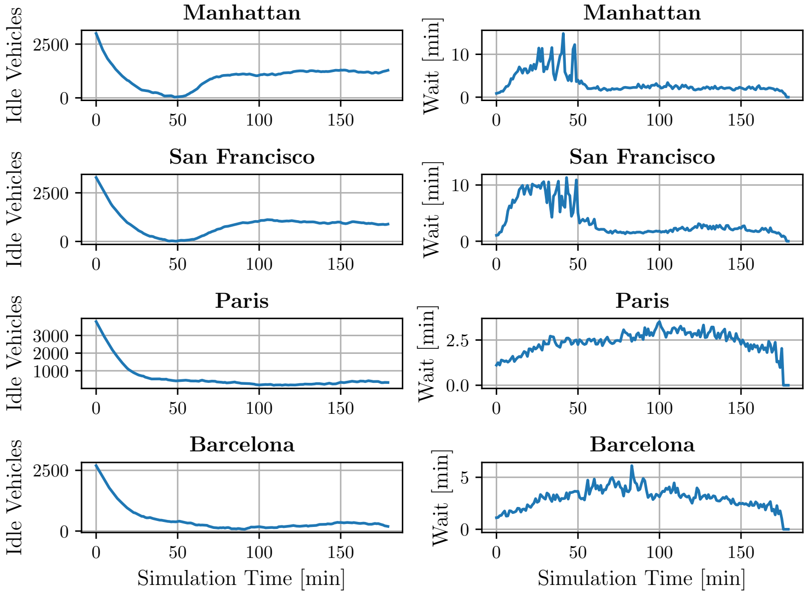


Figure 3.9: Variation of idle vehicle counts and pickup wait time over a stable queue setting.

3.5.4 Model Validity

Through sections 3.5.1 to 3.5.3, a bespoke ABM was structured using the framework proposed in section 3.2 to identify minimum AV TNC fleet sizes for various networks. The aim of this section is to scrutinize the validity of the model and discuss how the ABM framework of section 3.2 could have been used to solve different problems. The proposed model, although simplistic, offers a swift introduction on ABM structuring and a solid starting implementation for investigating AV TNC system dynamics.

In section 3.5.1, no traveller acumen or competition in the market were assumed. As a consequence, travellers never abort the service and the requests pile up in the queue if the fleet size is unsustainable. The objective of this simplification is to acquire an upper bound on the minimum fleet size in each network implementation. Additionally, the model did not account for endogenous congestion in the system, as the relevant fleet sizes are only a negligible fraction of the background traffic in the network.

In realistic scenarios, intelligent travellers with wait times beyond the ones they deem acceptable, would abort the service and resort to other solutions, thereby not contributing to long unsustainable queues. Furthermore, an AV TNC would increase ride prices during peak travel via the use of dynamic pricing strategies, to sway excess demand to other modes and ensure a stable service. This system response would also imply complex and intelligent customer and operator agents in the ABM.

Nonetheless, even in the absence of such complicated ABM scenarios discussed above, the minimum sustainable fleet size results found in the previous section do serve as upper bounds for the service. Furthermore, the model used serves as an extensible basis for more complicated problems, such as optimal pricing strategies, assignment operations and fleet management decision making (i.e. electric charging, parking, maintenance, idle redistribution).

To comprehend the required extensible features for the basic ABM structure to be used in more complicated problems, the relationships of the different operations in the AV TNC system identified in chapter 2 need to be revisited, and what the differences are in the switch from conventional ride-sourcing platforms to autonomous services (figure 2.1). In conventional TNC systems, both demand for rides and supply of drivers is endogenous, with travellers and drivers making online decisions depending on wait time and pricing specifics. The ride-sourcing platform requires a good understanding of both sides, so as to set system parameters which will lead to traveller/driver behavior that will maximize revenue or profits. In the case of AVs, TNCs have complete control of the supply and need to choose their strategies so as to efficiently attract and serve demand.

As a consequence of the above description, the most important extension in the ABM model in this section to tackle more complicated problems, is intelligent traveller behaviour. This extension is regarded as essential, especially in testing assignment and pricing strategies which are heavily reliant on traveller choice. Furthermore, realistic implementations of traveller behavior would require choice models calibrated using TNC ride data or stated preference surveys. Additional extensions could be endogenous congestion and the use of a car following model in vehicle movement. Nonetheless, these extensions only become relevant when the fleet size is comparable to the background traffic of a network, so as to have a sizeable impact on the link velocities.

3.6 ABM Module Extensions in Subsequent Chapters

As discussed in section 3.5.4 the ABM components used to address the critical fleet size problem in this chapter, do not reflect the sophistication of a realistic ride-sourcing environment. Nonetheless, the selected relaxations in the ABM structure were such so as to identify an upper bound with regards to the critical fleet size problem and the estimation of maximum wait times. For clarity, a modified version of figure 3.2 which outlines the visualised ABM framework is shown in figure 3.10. Figure 3.10 highlights that the active components in the agent behavior for the critical fleet size ABM in the chapter are only the assignment of vehicles to customers and their routing throughout the network.

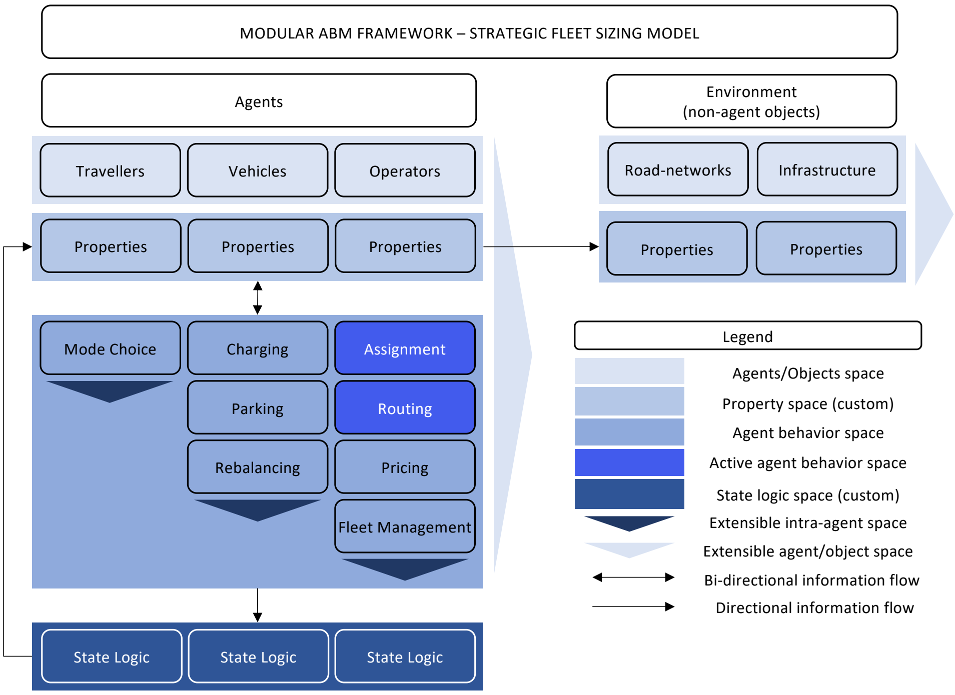


Figure 3.10: Active agent behavior modules in the critical fleet size ABM.

In subsequent chapters, the focus of the study will shift to decision problems relating to shorter time-horizons with the aim of providing practically implementable solutions at the end of each chapter. Through this time-horizon refinement, the ABM framework will be used to test the effectiveness of proposed methodologies. As such, the use of a more sophisticated module composition to attribute for more realism in the simulation environment will become more relevant.

Specifically, Chapter 4 will examine the fleet management problem with a focus on centralised idle vehicle rebalancing and Chapter 5 will investigate the problems of pricing and assignment of shared rides during peak travel times. Consequently, to guide the vehicle rebalancing, pricing and assignment operations, the ABM framework presented in this chapter will be extended to include pricing, fleet management and mode choice behaviors for operators, vehicles and customers respectively. Figure 3.11 outlines the ABM agent behavior modules which will be activated in subsequent chapters.

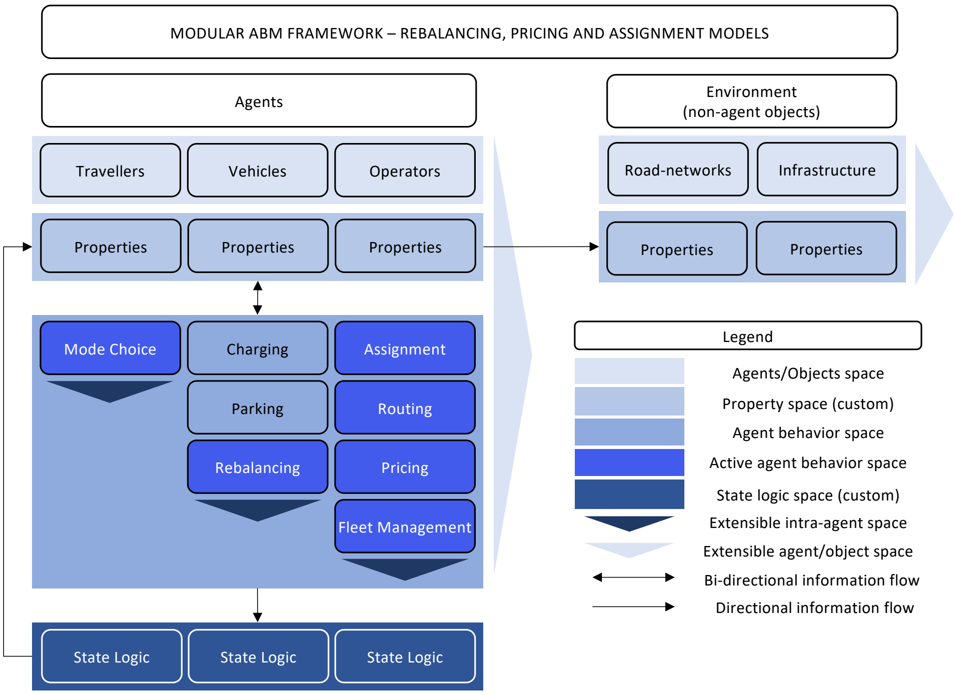


Figure 3.11: Active agent behavior modules in the idle vehicle rebalancing, assignment and pricing ABM tests.

3.7 Summary

In this chapter, the study proposed a framework for ride-sourcing simulations founded from the fundamentals of agent-based modelling. The study defined the basic building blocks of ride-sourcing simulations, by using explicit ABM terminologies, such as definitions for the environment, the agents, and topology. The chapter then elaborated on the various forms the above components could take and offered baseline modelling examples, such as vehicles and customers as agents, and their respective states.

The study then suggested a modular approach for creating bespoke simulations using this framework with the aid of object-oriented programming. The model building approach's main proposed feature is the practice of building ABM components in a modular way, moving from simple to more complex implementations of the model.

The study also considered the validation of an aggregated queuing theoretical model of a ride-sourcing platform, to tackle minimum fleet size. The fleet size model validation showcased the agent-based modelling framework's necessity and outlined the complexity of ride-sourcing systems. In doing so, the study explored the capabilities of queuing theory to accurately estimate the critical fleet size and maximum wait time, as aggregated measures of a ride-sourcing service.

Four case study applications were selected to test the validity of the queuing theoretical model and highlight the limits of aggregated representations of the system across various road-network structures; the urban areas of Manhattan, San Francisco, Paris and Barcelona. The results imply $M/M/\infty$ queues govern the operation of the ride-sourcing service across the entire urban network for the selected demand volumes. The work also justifies that the inclusion of pickup waiting time in modelling ride-sourcing operations via queuing theory is essential in identifying the minimum fleet size.

The chapter concludes that agent-based modelling is required to fully capture the complex dynamics that govern the critical fleet sizes in ride-sourcing systems. As decision problems in ride-sourcing extend beyond the strategic domain, using the proposed ABM framework is also relevant in the tactical and operational time-horizons. Consequently, in the next chapter, the study will investigate how the proposed ABM framework and building sequence could be used to optimize fleet management operations within the tactical time-horizon.

Chapter 4

Integer Programming Methodologies for Tactical Fleet Management Decisions

4.1 Introduction

As observed in Chapter 3, pickup wait times can have a sizeable impact on the utilization of a fleet. Specifically, lower pickup wait times can translate to more trips per vehicle, leading to a more profitable operation. Consequently, fleet redistribution strategies and sophisticated assignment algorithms can improve a fleet's operational performance. However, the current structure of TNCs is not in line with centralized fleet management to reduce pickup wait times and improve service quality. Drivers in TNC platforms act as independent entities; therefore, the decision making in terms of the vehicle resource across a network is decentralized.

Even so, TNCs currently have mechanisms which alleviate imbalances in the supply and demand of rides. Such imbalances can increase customer wait times in areas where there is an under-supply of drivers, thereby decreasing the quality of service and the popularity of the platform. To mediate this effect, TNCs apply dynamic pricing strategies, usually in the form of variable surge pricing multipliers [13]. By design, these dynamic pricing strategies motivate drivers to redistribute to under-served areas and suppress demand from customers whom their willingness to pay is exceeded [80].

As mentioned in the previous chapters, the anticipated launch of autonomous vehicles in TNC services to cut operational costs could transform TNCs from matching platforms to fleet operators having complete control of the supply [12]. In such a scenario, currently implemented dynamic pricing strategies would still suppress demand [122], but TNCs, as fleet owners, would need to decide vehicle redistribution operations. Generally, in the absence of drivers entering the market proactively by knowing historical surge pricing patterns [13], autonomous vehicle ride-sourcing operators would need to manage their fleet effectively, to alleviate any asymmetries of demand across road-networks.

Fleet management, and especially empty vehicle redistribution, although not prevalent in ride-sourcing markets, has been an established practice in shared mobility (bike and car-sharing) [64, 65]. In existing shared mobility schemes, vehicle redistribution is carried out by dedicated staff. In the case of autonomous ride-sourcing, platforms would instruct vehicles to self-relocate, thereby avoiding dedicated staff costs. TNCs could also proactively decide vehicle redistribution operations by exploiting the diverse area of predictive algorithms and existing data.

Nonetheless, this seamless autonomous vehicle relocation would endure mileage costs. Increased fleet mileage can induce externalities such as congestion subject to fleet adoption rates [42]. Furthermore, as ride-sourcing markets are competitive, and travellers encounter alternative options, redistributed vehicles in an area are not guaranteed an assignment. Consequently, vehicle redistribution models which take account of relocation costs, local market structure and travel behaviour are paramount in assessing the value of vehicle redistribution to autonomous ride-sourcing platforms.

As such, this Chapter focuses on identifying an effective fleet management strategy. To do so, the Chapter utilizes mathematical programming models, such as linear and convex integer optimization. Initially, in Section 4.2, presents an example Mixed Integer Linear Programming (MILP) formulation for maximizing daily profits, derived using the relevant car-sharing literature outlined in Section 2.2. The model in Section 4.2 accounts for aggregated charging and maintenance costs, as well as mileage costs.

Considerations about computational complexity and lack of traveller acumen in the example model presented in Section 4.2, lay the groundwork for the more sophisticated fleet management optimization model presented in Section 4.3 which fits an established computational complexity framework and assumes intelligent traveller decisions. The model is implemented on a large scale urban case study and compared with other benchmarks in Section 4.4 to demonstrate the feasibility and efficiency of the algorithm presented in Section 4.3.

4.2 A Linear Programming Model for Fleet Allocation

The purpose of the example model presented in this section is to highlight the interplay of relevant variables when optimizing the operations of an AV ride-sourcing fleet. In this regard, the fleet size and maintenance infrastructure are regarded as variable quantities to be decided by the model. Following the model description, a test case of the Sioux Falls network is implemented for validation. The limitations of the model are discussed at the end of the Section, which serve as a preamble for Section 4.3.

4.2.1 Model Description

The model assumes a ride-sourcing fleet operating over a network $G = (V, E)$, with V and E representing the sets of graph vertices and edges respectively. Requests can be accommodated by electric AVs of the fleet, when enough available vehicles exist at the origin locations. Furthermore, some candidate depots are created across the network which provide electric charging, parking and maintenance facilities. The operations are split into time intervals/steps. Operations may start and finish at different time steps. The system supply is based on stochastic demand throughout the duration of the process. At each time step, an OD matrix is produced, drawn from an expected OD matrix. OD values are modelled using a Poisson distribution and identified by a matrix of mean OD values at each period. A congestion factor is applied to the travel time for each trip.

The vehicles can be identified within four operations, rental/trip, empty relocation, depot related operations, or being idle/unused at a node. Considering the trip operation, at each time step, a number of requests between nodes is created. Depending on the number of vehicles available and the number of trips, an amount of vehicles is assigned to trips at each time step. With regards to relocation, the system tries to predict demand and distribute the vehicles accordingly at different time intervals. Hence, vehicles which might be empty or engaged to a trip will relocate according to the expected demand across the request catchments. The AVs are required to visit depots for charging and maintenance requirements. A daily charging and maintenance quota exists depending on the size of the fleet. As a result, the model ensures hat the quota is at least matched, with visits of vehicles to depots. If a vehicle is not assigned to a trip or relocation, neither it is involved in any depot related movement at a time step, it is then left idle/unused at the node it was located at the beginning of that time step.

The assignment of vehicles within operations based on the expected and actual demand is made so as to maximise the objective function of the MILP, which is formed by subtraction of the total costs from the revenue. Each trip generates units of revenue for the operator based on the ride charge the customer pays. The ride charge has a fixed and a variable part, both of which are functions of trip duration and distance. The operator costs are as follows:

- **Depot Establishment Cost**: A fixed cost is assumed for each depot establishment, with an additional charge for each parking spot established within each depot.
- **Vehicle Establishment Cost**: A fixed cost is assumed for each vehicle establishment.
- **Charging and Maintenance Cost**: A daily quota for charging and maintenance is assumed depending on the fleet size. The battery range is assumed to be high enough so that one daily visit to a depot per AV is required. This serves the model choice of not tracking individual vehicles through the network to avoid additional computational burden in the model.
- **Relocation Cost**: A cost for relocation is assumed, which varies according to the congestion and distance of relocation.
- **Depot Movement Cost**: A cost for when a vehicle is moving to and from a depot is assumed, which varies according to the congestion and distance of movement.
- **Unmet Demand Penalty**: A penalty(cost) is assumed to occur to the operator for any amount of demand which is not served at each time step.

The sets, indices, functions and parameters used in the model are summarised in Table 4.1, whereas the decision and auxiliary variables assumed are summarised in Table 4.2. Furthermore, Table 4.3 outlines the any preliminary expressions used in the MILP.

Table 4.1: Sets, indices, functions and parameters used in the model.

Sets and Indices	Description
$q \in Q$	(candidate) Station index
$i, j \in V$	Node indices
$t, u \in T$	Time step indices
Functions	**Description**
$dti(t, i, j)$	Time steps required to travel from node i to node j at time step t
$dtq(t, i, q)$	Time steps required to travel from node i to depot q at time step t
$intseti(t, i, j)$	Earliest time step a vehicle started travelling from node i to node j and has not arrived at time t
$intsetq(t, i, q)$	Earliest time step a vehicle started travelling from node i to depot q and has not arrived at time t
Parameters	**Description**
DC	Depot establishment cost
PC	Parking spot establishment cost
AVC	Vehicle establishment cost
CMC	Daily charging and maintenance cost/quota per vehicle
B_f	Base fare for hiring a vehicle
T_f	Time fare rate for hiring a vehicle
D_f	Distance fare rate for hiring a vehicle
$TFUD$	Time penalty rate for unmet demand
$DFUD$	Distance penalty rate for unmet demand
Δ_t	Congestion level factor for time step t
P	Maximum number of candidate parking spots available for establishment at each depot
N	Maximum number of open depots
AV	Maximum number of vehicles
MC	Charging and maintenance cost per time step per car
ED_{ij}^t	Expected demand for trips from node i to node j at time step t
AD_{ij}^t	Actual demand for trips from node i to node j at time step t
TR_{ij}	Travel time from node i to node j
D_{ij}	Travel distance from node i to node j
TQ_{iq}	Travel time from node i to depot q
DQ_{iq}	Travel distance from node i to depot q

Table 4.2: Decision and auxiliary variables assumed in the model.

Decision Variables	Description
x_q	Binary variable indicating if candidate depot q is open
p_q	Number of established parking spaces at station q
c	Fleet size
Auxiliary Variables	Description
a_i^t	Number of vehicles left idle in node i from previous time step at time t
e_{ij}^t	Number of booked vehicles starting starting a trip from node i to node j at time t
$\bar{e}_{ij}^t$	Number of served requests arriving at node i from node j at time t
m_{ij}^t	Number of unserved requests from node i to node j requested at time t
r_{ij}^t	Number of vehicles to relocate from node i to node j starting at time t
$\bar{r}_{ij}^t$	Number of relocating vehicles arriving to node i from node j at time t
cs_{iq}^t	Number of vehicles to depart from node i to depot q at time t
$\bar{cs}_{iq}^t$	Number of vehicles arriving to depot q from node i at time t
sc_{qi}^t	Number of vehicles to depart from depot q to node i at time t
$\bar{sc}_{qi}^t$	Number of vehicles arriving to node i from depot q at time t
pv_q^t	Number of vehicles at depot q at the beginning of time t

Table 4.3: Auxiliary expressions used in the model.

Expression	Description
$sup_i^t = \sum_j \bar{e}_{ij}^t + \sum_j \bar{r}_{ij}^t + \sum_q \bar{sc}_{qi}^t + a_i^t \quad \forall t$	Number of vehicles available at node i at the start of time step t
$uv_t = \sum_i \left(sup_i^t - \sum_j e_{ij}^t - \sum_j r_{ij}^t - \sum_q cs_{iq}^t\right) \quad \forall t$	Number of unused vehicles at the start of time step t
$ev_t = \sum_i \sum_j \sum_{u=intseti(t,i,j)}^{t} e_{ij}^u \quad \forall t$	Number of vehicles occupied in trips at time step t
$rv_t = \sum_i \sum_j \sum_{u=intseti(t,i,j)}^{t} r_{ij}^u \quad \forall t$	Number of vehicles relocating at time step t
$csv_t = \sum_i \sum_q \sum_{u=intsetq(t,i,q)}^{t} cs_{iq}^u \quad \forall t$	Number of vehicles travelling to depots at time step t
$scv_t = \sum_i \sum_q \sum_{u=intsetq(t,i,q)}^{t} sc_{iq}^u \quad \forall t$	Number of vehicles travelling from depots at time step t
$dv_t = \sum_q pv_q^t \quad \forall t$	Number of vehicles at depots at time step t
$rev = \sum_t \sum_i \sum_j e_{ij}^t (B_f + T_f \times \Delta_t \times TR_{ij} + D_f \times D_{ij})$	Total income from served requests
$cde = \sum_q \left(DC + PC \times p_q\right) x_q$	Costs from established depots and parking spots
$cve = AVC \times c$	Cost for establishing vehicles
$ccm = MC \sum_t \sum_q pv_q^t$	Costs for visits to depots for charging and maintenance
$cr = \sum_t \sum_i \sum_j r_{ij}^t (T_f \times \Delta_t \times TR_{ij} + D_f \times D_{ij})$	Cost of relocations
$cdm = \sum_t \sum_i \sum_q \left(cs_{iq}^t + sc_{iq}^t\right)(T_f \times \Delta_t \times TRQ_{iq} + D_f \times DQ_{iq})$	Cost of moving to and from depots
$cud = \sum_t \sum_i \sum_j m_{ij}^t (TFUD \times \Delta_t \times TR_{ij} + DFUD \times D_{ij})$	Cost of unmet demand

Using the information from Tables 4.1, 4.2 and 4.3, the following MILP is formulated:

$$\text{maximize} \quad rev - cde - cve - ccm - cr - cdm - cud \tag{4.1a}$$

subject to

$$\sum_i a_i^1 = c, \tag{4.1b}$$

$$p_q \leq Px_q \qquad \forall q \in Q, \tag{4.1c}$$

$$\sum_q x_q \leq N, \tag{4.1d}$$

$$\sum_q x_q \geq 1, \tag{4.1e}$$

$$c \leq AV, \tag{4.1f}$$

$$cs_{iq}^t \leq AVx_q \qquad \forall t \in T, i \in V, q \in Q, \tag{4.1g}$$

$$sc_{iq}^t \leq AVx_q \qquad \forall t \in T, i \in V, q \in Q, \tag{4.1h}$$

$$AD_{ij}^t = e_{ij}^t + m_{ij}^t \qquad \forall t \in T, i, j \in V, \tag{4.1i}$$

$$\sum_j ED_{ij}^t \leq sup_i^t - \sum_j r_{ij}^t - \sum_q cs_{iq}^t \qquad \forall t \in T, i \in V, \tag{4.1j}$$

$$a_i^{t+1} = sup_i^t - \sum_j e_{ij}^t - \sum_j r_{ij}^t - \sum_q cs_{iq}^t \qquad \forall t \in T, i \in V, \tag{4.1k}$$

$$c = uv_t + ev_t + rv_t + csv_t + scv_t + dv_t \qquad \forall t \in T, \tag{4.1l}$$

$$pv_q^t \leq p_q \qquad \forall t \in T, q \in Q, \tag{4.1m}$$

$$pv_q^t = pv_q^{t-1} + \sum_i \bar{cs}_{iq}^t - \sum_i sc_{iq}^t \qquad \forall t \in T, q \in Q, \tag{4.1n}$$

$$\sum_i sc_{iq}^t \leq pv_q^{t-1} \qquad \forall t \in T, q \in Q, \tag{4.1o}$$

$$c \times CMC \leq MC \sum_t \sum_q pv_q^t, \tag{4.1p}$$

$$\sum_i \bar{cs}_q^t = \sum_i sc_{iq}^{t+CMC/MC} \qquad \forall t \in T, q \in Q, \tag{4.1q}$$

$$e_{ij}^t = \bar{e}_{ji}^{t+dti(t,i,j)} \qquad \forall t \in T, i, j \in V, \tag{4.1r}$$

$$r_{ij}^t = \bar{r}_{ji}^{t+dti(t,i,j)} \qquad \forall t \in T, i, j \in V, \tag{4.1s}$$

$$cs_{iq}^t = \bar{cs}_{iq}^{t+dtq(t,i,q)} \qquad \forall t \in T, i \in V, q \in Q, \tag{4.1t}$$

$$sc_{iq}^t = \bar{sc}_{iq}^{t+dtq(t,i,q)} \qquad \forall t \in T, i \in V, q \in Q, \tag{4.1u}$$

$$r_{ij}^t = 0 \qquad \forall t \in T, i = j \in V, \tag{4.1v}$$

$$x_q \in \{0, 1\} \qquad \forall q \in Q, \tag{4.1w}$$

$$a_i^t, c, p_q, cs_{iq}^t, sc_{iq}^t, e_{ij}^t, m_{ij}^t, rm_{ij}^t, r_{ij}^t, pv_q^t, \bar{e}_{ij}^t, \bar{r}_{ij}^t, \bar{sc}_{iq}^t, \bar{cs}_{iq}^t \in \mathbb{N} \quad \forall t \in T, i, j \in V, q \in Q \tag{4.1x}$$

The model aims to maximize the profit, with a subtraction of the costs from the revenue. Equation (4.1a) also contains costs such as relocation, the movement to and from depots and also a penalty for unmet demand. By incorporating these costs, the model ensures there is no excess movement to other nodes or depots. Furthermore, the unmet demand cost captures the customer perspective in the same objective.

Constraint (4.1b) ensures that initially all the vehicles are available and dispersed across the nodes. Constraint (4.1c) makes sure that parking spots are only assigned to open depots and are always less than or equal to the maximum allowed. Constraints (4.1d) and (4.1e) ensure the highest number of open depots is less than or equal the maximum allowed and also that there is at least one open depot. Constraint (4.1f) sets the fleet size to be less than or equal the maximum allowed. Constraints (4.1g) and (4.1h) establish that there are no trips to and from closed depots.

Furthermore, constraint (4.1i) is related to supply and demand of vehicles at nodes. Constraint (4.1i) states that at every time step, the demand for trips between a pair of nodes is split to served and unserved requests. Constraint (4.1j) ensures that the vehicles available to serve requests at each node and time step are at least as much as the predicted demand. Constraint (4.1k) establishes the continuity between time steps at nodes, with any unused vehicles after departures to be considered idle at those nodes in the next time step. Constraint (4.1l) ensures that the sum of vehicles at different states for each time step is always the same as the fleet size.

Constraints (4.1m)-(4.1p) related to depot capacity. Constraint (4.1m) makes sure that the parked vehicles at each depot never exceed the depot capacity. Constraints (4.1n) and (4.1o) establish continuity at depots between time steps. Also, constraint (4.1p) ensures that the charging and maintenance quota is spent. Constraint (4.1q) ensures that vehicles which visit depots for charging and maintenance always leave the depot after they receive services. Therefore CMC/MC is set to an integer so that it can be accounted as a time index for the duration of a charging and maintenance session.

Constraints (4.1r)-(4.1u) ensure that vehicles change states after the respective time needed for each type of trip to be completed. Finally, constraint (4.1v) ensures there are no assigned relocations between the same nodes.

4.2.2 Preliminary Test Case

The example MILP proposed in 4.2.1 was tested on a modified version of the Sioux Falls network as observed in Figure 4.1. The network consisted of 24 nodes, 38 bi-directional links and six candidate locations for depots. Depots are geographically located on network nodes, however are assumed to be additional nodes to their geographical counterparts. The choice of candidate stations was manually performed with the objective that every node could have a connection to a candidate station using at most two links.

The network as shown in figure 4.1 and OD matrix were obtained from [123]. By assuming a 2.4% mode share for the fleet the daily OD values were divided by 12000 [1] to obtain an estimate for AV trip requests for five minute intervals during the day. The resulting five-minute OD flows were then multiplied by a congestion factor which varied through the 24 hours. The final 3D matrix was regarded as the expected demand per time step (ED_{ij}^{t}). The congestion factor used assumed two scenarios; one for weekdays which utilised a 6th order polynomial with two peaks throughout the day, and a weekend scenario which utilised a 3rd order polynomial with one peak throughout the day. The choice of polynomial functions was performed intuitively based on the expected pattern of peak time traffic in urban areas. The values of ED_{ij}^{t} were then used as means of a Poisson distribution for the number of trips between two nodes within five-minute intervals and were used to obtain random values for AD_{ij}^{t}.

Table 4.4 outlines the parameters used for both scenarios and figures 4.2a and 4.2b show the trip distributions for weekdays and weekends, with AD_{ij}^{t} to vary with each run of the model. The starting time step was chosen at 07:00am. The input arrays for the model as well as the functions were created in MatLab R2015b. The mathematical model was composed and solved in IBM ILOG Cplex Studio (version 12.6.1). The runs were performed on a modern workstation with a 6-core Xeon CPU (at 3.60 GHz) and 32GB RAM.

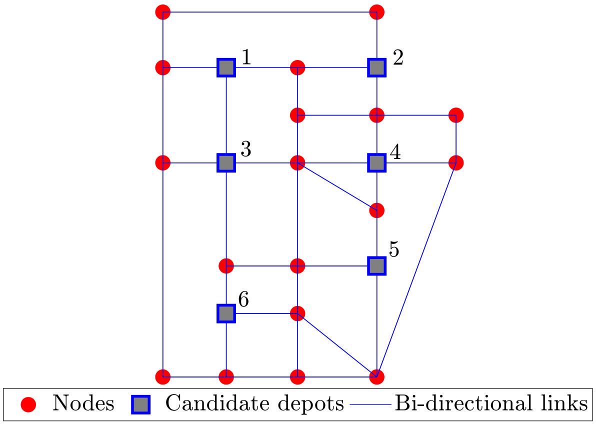


Figure 4.1: Sioux Falls modified network used.

Each scenario was executed ten times to account for some variation in AD_{ij}^{t}. Tables 4.5 and 4.6 summarize the results for weekdays and weekends respectively. Notably, the choice of applying a penalty at the same value as the revenue rates had a significant effect on the resulting low number of unmet demand for both scenarios. Due to this and the cost of relocation compared with the cost of additional vehicles, the model produces a large fleet with few relocations instead of a smaller fleet with frequent idle vehicle movement. Furthermore, as the served requests always need to more than the expected demand, this also contributes to a larger fleet with less relocations. Nonetheless, the depot establishment count appears to be consistent for almost all the executions for both weekdays and weekends. Figures 4.3a and 4.3b outline the number of vehicles in each state for typical runs of weekdays and weekends respectively, confirming the low value of relocations.

[1] 288 five-minute intervals throughout the day divided by 2.4% mode share (288/2.4%=12000)

Table 4.4: Parameter values used.

Parameter	Value
DC (per day)	£100.00
PC (per day)	£40.00
AVC (per day)	£60.00
CMC (per day)	£60.00
B_f	£5.00
T_f (per minute)	£1.15
D_f (per km)	£3.50
$TFUD$ (per minute)	£1.15
$DFUD$ (per km)	£3.50
P	100
N	6
AV	10000
MC (per step t)	£20

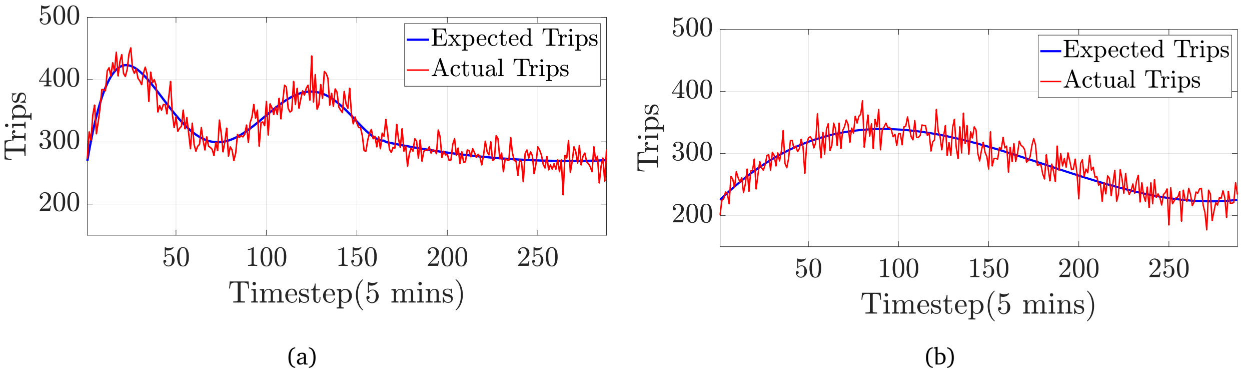


Figure 4.2: Resulting typical trip variations over weekdays (a) and weekends (b).

4.2.3 Practical Implementation Considerations

The model presented in Section 4.2 outlines how the various costs involved in a ride-sourcing operation vary. The notion of idle vehicle redistribution was preliminary investigated, which provided useful information on the trade-offs of empty mileage versus greater fleet size. Arguably, the benefits of either of these decisions differ depending on the implementation. For example, additional vehicles in a fleet would require additional maintenance and insurance costs, as well as the initial capital for their purchase. On the other hand, a small fleet size with anticipatory empty vehicle redistribution might imply higher spending on fuel and more congested streets.

The limitations of the model presented above hinder its application to lower level operations other than for deriving strategic insights. Specifically, smartphone technologies nowadays allow for instant access to information by users. This increases the elasticity between different options in the transport market. Accounting for the demand as an exogenous parameter is not in accordance with the competitiveness presented in the market. As a consequence, the fluctuation of service quality (i.e. wait time) could

Table 4.5: Results for weekday scenario.

Run	c	$\sum x_q$	$\bar{uv}_t$	$\sum_t \sum_i \sum_j m_{ij}^t$	Runtime [sec]
1	1392	2	667	54	186.5
2	1392	2	667	54	177.4
3	1401	2	673	38	272.4
4	1392	2	667	54	171.5
5	1401	2	673	38	254.5
6	1419	2	687	59	266.0
7	1413	2	684	53	179.2
8	1407	2	675	57	273.0
9	1398	2	675	43	287.9
10	1428	2	700	97	429.5

Table 4.6: Results for weekend scenario.

Run	c	$\sum x_q$	$\bar{uv}_t$	$\sum_t \sum_i \sum_j m_{ij}^t$	Runtime [sec]
1	926	2	348	1	167.9
2	925	3	345	3	231.7
3	926	2	348	1	167.3
4	922	2	338	2	284.4
5	912	2	330	1	188.3
6	906	2	323	5	162.8
7	929	2	351	6	981.4
8	927	2	344	4	259.2
9	959	2	377	6	183.8
10	909	2	328	4	210.4

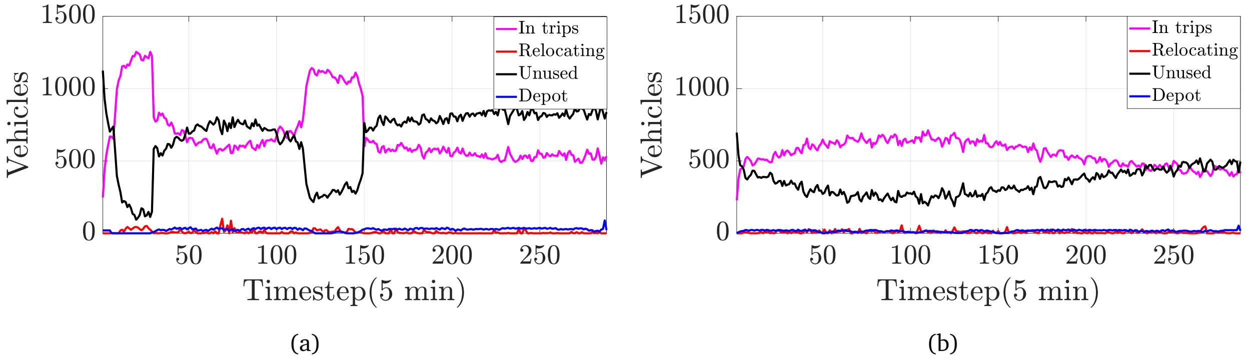


Figure 4.3: Resulting typical vehicle state variations over weekdays (a) and weekends (b).

influence traveller choices. Furthermore, as service quality is directly related with the ability of an operator to meet incoming requests, more realistic models would imitate the interplay between supply of vehicles and demand for rides.

Additionally, the model offers strategic insight on how costs and revenues might progress throughout a daily operation if specific parameters are provided but offers no framework for practical online implementation. Daily volumes of requests are not known in advance so the time horizon for decision making on assignment, idle redistribution and/or refueling must be more refined. In doing so, realistic road-network implementations would be utilised, which will inevitably increase the computational run-time of the model observed in Table 4.6, rendering it infeasible. In that respect, a practical implementation of an online fleet management optimization model, should fit into a known computational complexity framework, so as to make use of available efficient algorithms to find a solution.

4.3 Fleet Allocation using Convex Network Flows

This section, takes into account the practical implementation considerations outlined in Section 4.2.3 with regards to designing online fleet management tools and presents a more sophisticated fleet management model compared to the example presented in Section 4.2.

4.3.1 The Vehicle Redistribution Problem

The model considers an autonomous vehicle ride-sourcing fleet operator, opposed with the problem of identifying an allocation of vehicles to various operations to minimise the fleet's operational cost. The fleet operator identifies allocations of the vehicles at regular decision periods and operates in an urban road network split into different clusters.

Immediately before the allocation decision, the operator identifies the vehicle counts in each cluster, including the vehicles soon to be located in each area. At each decision epoch, the fleet operator needs to allocate the vehicles in each cluster into three possible operational states; available for trip allocation, empty redistribution, or idle.

Vehicles assigned for trip allocation are immediately available for trip requests originating from their existing cluster, and their number depends on demand estimates for the commencing period. Empty redistribution refers to vehicles allocated for empty travel to other clusters to satisfy demand estimates for the commencing and subsequent periods. Finally, idle vehicles remain inactive in their initial cluster for the commencing period and act as reserve capacity for the fleet if required.

Consequently, vehicles are allocated from their initial state and clusters into the various operations at the beginning of the decision period and end up in updated states and clusters for the subsequent period. For convenience, the updated vehicle states and the operational states are referred to as resulting states and decision states respectively.

The set of road network clusters J, with the assumption the fleet operator has estimates of the total demand Z^t_{ij} from cluster i to cluster j for each $i, j \in J$ and for every time epoch $t \in T$. The model assumes the mean utility of travel in time epoch t for autonomous ride-sourcing trips from cluster i to cluster j for each $i, j \in J$ is realised using the following generalised cost function:

$$g^t_{ij}(x) = -\bar{v}(w^t_{ij}(x) + r^t_{ij}) - pr^t_{ij} \quad \forall i, j \in J, \forall t \in T \tag{4.2}$$

Where $\bar{v}$ in (4.2) is the mean value of time of the ride-sourcing travellers, $w^t_{ij}(x)$ is the average wait time from request to pick up for a trip originating in cluster i and terminating in cluster j at period t for a supply of vehicles x, r^t_{ij} is the travel time from cluster i to cluster j during period t and p is the price per time for the service.

As shown in Section 3.4 and in the studies in [84] and [117], the pickup wait time which is inversely proportional to the square root of idle vehicles in an area over a time interval. This relationship, as shown earlier, also takes inputs such as the area and average velocity in the network. Nonetheless, the relationship between idle vehicles over a period in a cluster and the total supply of vehicles x is non-trivial and of dynamic nature, as it depends upon the rate of requests and idle vehicle arrivals. Therefore, a function of wait time $w^t_{ij}(x)$ is defined, which has the following properties:

$$w^t_{ij}(x) = \frac{\beta^t_i}{\sqrt{I^t_{ij}(x)}} \quad \forall i, j \in J, t \in T, x \in \mathbb{R}^+ \tag{4.3}$$

Where $I^t_{ij}(x)$ in equation (4.3) is a function of the average number of idle vehicles in cluster i available for travel to cluster j with respect to supply x on period t. Similar to Section 3.4, β^t_i is proportional to the square root of the cluster area and inversely proportional to the average velocity in the network. β^t_i can be determined via calibration with the use of ABM. By utilising a discrete choice model and assuming constant values for r^t_{ij} and p^t_{ij} in each period, the proportion of travellers $q^t_{ij}(x)$ choosing the ride-sourcing fleet as an option to travel from cluster i to cluster j for a period t can be calculated.

$$q^t_{ij}(x) = \frac{e^{g^t_{ij}(x)}}{e^{g^t_{ij}(x)} + \sum_{w \in W} e^{U^{tw}_{ij}}} \quad \forall i, j \in J, \forall t \in T \tag{4.4}$$

Where W in (4.4) refers to the set of alternative ride-sourcing options and U^{tw}_{ij} is the mean utility of option $w \in W$ for travelling from cluster i to cluster j in period t. As such, the number of travellers $N^t_{ij}(x)$ choosing the ride-sourcing service at period t to travel from cluster i to cluster j for a supply of vehicles x is found using the following equation:

$$N_{ij}^{t}(x) = q_{ij}^{t}(x) Z_{ij}^{t} \quad \forall i, j \in J, \forall t \in T \tag{4.5}$$

Equation (4.4) is a sigmoid function, as it can be represented in the form of the logistic function. Equation (4.5) is a scaled version of the sigmoid function in (4.4). Consequently, due to its non-linearity and monotonically increasing nature, the function for the number of travellers choosing the service $N_{ij}^{t}(x)$ does not necessarily match the supply of vehicles x.

4.3.2 Non-Linear Minimum Cost Flow Formulation

The logic defined in equations (4.2)-(4.5), represents the aggregated model which the fleet operator uses to estimate trips from and to each cluster. The number of travellers choosing the ride-sourcing service, however, is contingent on the redistribution strategy and the vehicle supply, which shapes the service quality defined in equation (4.3).

Therefore, a minimum cost flow formulation is proposed, in the form of resource allocation to solve the vehicle redistribution problem described in section 4.3.1. Consider a directed acyclic graph $G = (V, E)$, with V and E representing the sets of graph vertices and edges respectively. The set of vertices V consists of three subsets A, B, C, representing the initial states, decision states and resulting states respectively, such that $V = A \cup B \cup C$.

For initial state vertices, the model considers the numbers of available vehicles at the beginning of epoch t at each cluster. The decision state vertices are also subdivided into the subsets K, L, M of trip, redistribution and idle states respectively, such that $B = K \cup L \cup M$. Finally, for resulting states, the model considers the numbers of available vehicles at the beginning of epoch $t+1$ at each cluster. Vertices in each set are associated with the set of road network clusters J such that $|A| = |K| = |L| = |M| = |C| = |J|$. Consequently, the cardinality n of the set V of vertices is $n = |V| = 5|J|$.

A graph edge $(i, j) \in E$ between two vertices $i, j \in V$, represents the change from state i to state j due to allocation decisions. Figure 4.4 outlines an example of our proposed resource allocation graph with two clusters. Vertices in A have directed edges which connect to the vertices in B only in their respective cluster, with a direction from A to B. Consequently, there are 3 edges from vertices in A to vertices in B for each cluster.

The model further assumes edges between vertices in B and C. Each vertex in K connects to all vertices in C. For redistribution edges starting from vertices in L and terminating to vertices in C, edges which start and terminate in the same cluster are excluded. Finally, for each cluster, the model defines an edge from the corresponding cluster vertex in M to the corresponding cluster vertex in C. As such the cardinality m of the set of edges E is $m = |E| = 3|J| + |J|^2 + |J|(|J| - 1) + |J| = |J|(2|J| + 3)$.

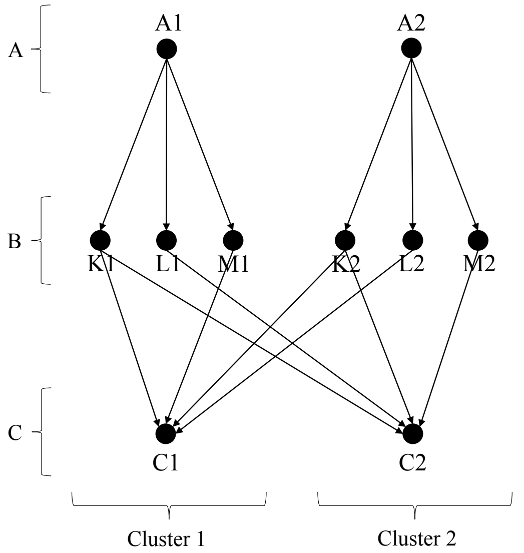


Figure 4.4: Example of the proposed resource allocation graph with 2 clusters.

To assist the minimum cost flow formulation, the following functions are introduced on E: a lower bound $l_{ij} \geq 0$, a capacity $u_{ij} \geq l_{ij}$ and a cost c_{ij} for each $(i, j) \in E$. Furthermore, the balance vector function $b : V \rightarrow \mathbb{Z}$ is also define which associates integer numbers with each vertex in V. The model assumes the following holds:

$$\sum_{v \in V} b(v) = 0 \tag{4.6}$$

The flow in graph G is denoted as a function $x : E \rightarrow Z^{\geq 0}$ on the edge set of G, such that the value of the flow on edge (i, j) is x_{ij}. The balance vector function $b(v)$ for each $v \in V$ is the difference between the flow in edges of out-degree of v and the flow in edges of in-degree of v. As such, the balance vector b is the following function on the vertices:

$$b(j) = \sum_{i:ji \in E} x_{ji} - \sum_{i:ij \in E} x_{ij} \quad \forall j \in V \tag{4.7}$$

The vertices with $b(v) > 0$ are classified as source vertices, whereas vertices with $b(v) < 0$ are sink vertices. Otherwise, if a vertex has $b(v) = 0$ that vertex is classified as balanced. Consequently, a flow x in G is feasible if $l_{ij} \leq x_{ij} \leq u_{ij}$ for all $(i, j) \in E$ and equations (4.6) and (4.7) hold for all vertices $v \in V$. Considering the resource allocation problem introduced in Section 4.3.1, the vertices $v \in A$ can be regarded as source vertices since these constitute the initial states of all vehicles in the fleet. In a similar fashion, the vertices in C can be identified as sinks, however; their balance vectors cannot be defined in advance, as certain demand requirements might lead to violation of (4.6).

The values of the balance vector function for sink nodes are related to the demand for trips towards each cluster in the road network. As the minimum cost formulation for solving the resource allocation problem would adhere that a feasible flow satisfies equations (4.6) and (4.7), the graph is transformed from a multi-source, multi-sink to a multi-source, single-sink one. To do so, the set of vertices D is introduced, for which $|D| = |J|$ such that there is one vertex from set D in each cluster. The sink vertex t is also introduced for which $t \cap J = \emptyset$. Vertices in D denote demand satisfiability for the subsequent period. Consequently, $V \leftarrow V \cup (D \cup t)$.

Vertices from C are connected with edges to vertices in D in each cluster, to denote that vehicles which terminate their tasks or are idle in a cluster during a time epoch, could be available if needed in the same cluster for the subsequent period. Furthermore, to account for excess vehicles for subsequent demand, the model also considers directed edges from all vertices in C to the sink vertex t. The flow is transferred from vertices in D to the sink vertex t by including additional directed edges between them.

Edges starting from vertex sets K to C in each cluster as shown in Figure 4.4 represent trip edges. If one focuses on an individual cluster, the vehicle flow through these edges originates from the same cluster and aims to satisfy the demand for the commencing period. However, depending on geographical proximity, redistributing vehicles from other areas might arrive in the cluster before the end of the commencing period. As a consequence, redistributing vehicles could be exposed to a portion of the demand originating from the cluster during the commencing period.

The above description implies that within a period, in each cluster, there can be variable supply levels exposed to variable demand portions due to the mixing of redistributing vehicles from different clusters. This behaviour is captured in the formulation of the rebalancing problem in [60]. To account for the intra-period redistribution mixing, additional sets of vertices and edges are introduced to the network described in Figure 4.4 between the vertex sets of B and C. Specifically, in each cluster, vertices are added, representing the arrival of vehicle flow from redistributing vertices L from other clusters. As a consequence, $|J| - 1$ vertices are added in each cluster. These additional vertices are extensions to the vertex subset K since they are trip vertices.

Each additional edge in K is then connected with a redistribution vertex in L from other clusters, resulting in $|J|(|J| - 1)$ additional edges. Also, each additional edge in K is connected with all vertices in C, resulting in $J^2(|J| - 1)$ additional edges. Furthermore, to model the vehicle mixing with vehicles already in each cluster, new edges need to be added from the original vertices in K, to the additional vertices in K. To do so, the sequence of vehicle mixing is identified using a sorted list of the arrival times in each cluster.

For convenience, in each cluster i in J, original vertices in K are denoted as K_i. As the arrival of vehicles from other clusters has a cumulative effect on the supply in each cluster, the additional vertices in K are denoted using the sequence of arrivals. For example, if vehicles from L_j arrive in cluster i before vehicles from L_k for i, j, k in J, the vertices corresponding to the mixing of vehicles are denoted as K_{ij} and K_{ijk}, for redistribution occurring from clusters j and k respectively. Finally, to complete the mixing, in each cluster, $|J| - 1$ edges are introduced between the intra-cluster vertices in K according to the sequence of arrivals. Revisiting the above example, in cluster i, directed edges are introduced from K_i to K_{ij} and from K_{ij} to K_{ijk}. An outline of the transformed graph is shown in Figure 4.5. As such, the following equations hold:

$$b(v) > 0 \quad \forall v \in A \tag{4.8}$$

$$b(v) = 0 \quad \forall v \in B \cup C \cup D \tag{4.9}$$

$$b(t) < 0 \tag{4.10}$$

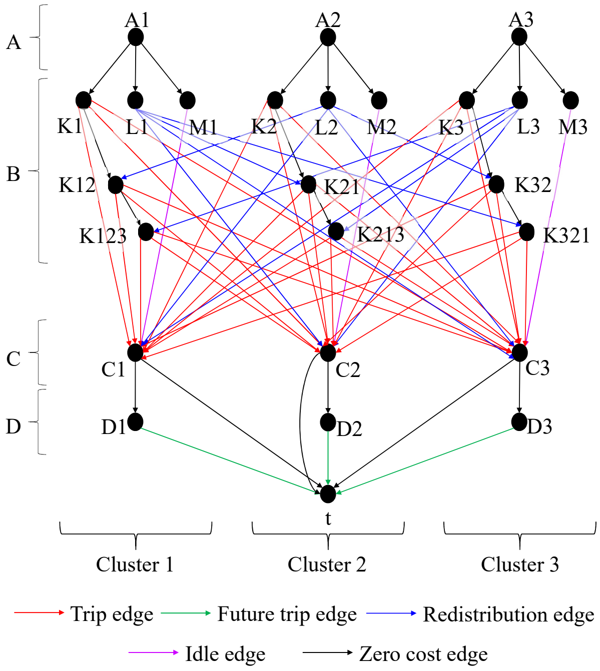


Figure 4.5: Example of the transformed multi-source single-sink graph with 3 clusters.

To assist the notation, a function $n : V \to J$ is defined, which maps vertices of the resource allocation network to clusters in J. Furthermore, to simplify set notation for edges, the edge sets $\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D}, \mathcal{E} \in E$ are defined. These edge sets represent the trip, future trip, redistribution, idle and zero cost edges respectively, as described in Figure 4.5. As such, the cost functions for the edges in the graph are defined as follows:

$$c_{ij}(x_{ij}) = h_{ij}(x_{ij}) \quad \forall (i,j) \in \mathcal{A} \tag{4.11}$$

$$c_{ij}(x_{ij}) = r^1_{n(i)n(j)} C_M x_{ij} \quad \forall (i,j) \in \mathcal{C} \tag{4.12}$$

$$c_{ij}(x_{ij}) = C_I x_{ij} \quad \forall (i,j) \in \mathcal{D} \tag{4.13}$$

$$c_{ij}(x_{ij}) = f_{ij}(x_{ij}) \quad \forall (i,j) \in \mathcal{B} \tag{4.14}$$

$$c_{ij}(x_{ij}) = 0 \quad \forall (i,j) \in \mathcal{E} \tag{4.15}$$

Equations (4.11)-(4.13) define the cost functions for edges directed from decision state vertices B to vertices in C. Equation (4.11) can have a negative sign, as the revenues from trip allocations with a supply of x_{ij} in the next epoch are subtracted from the cost using function $h_{ij}(x_{ij})$. Parameter $r^1_{n(i)n(j)}$ is the average travel time between the clusters $n(i)$ and $n(j)$ of vertices i and j at the initial epoch as introduced in equation (4.2), and C_M is the cost of a moving vehicle per time. Consequently, equation (4.12) defines the cost of redistribution for vehicles. Equation (4.13) defines the cost of idle vehicles, with C_I to denote the cost of an idle vehicle per period.

Equation (4.14) defines the potential profit for vehicles available in a cluster in the subsequent period using function $f_{ij}(x_{ij})$. As such, it guides vehicle redistribution and idle vehicle strategies. Finally, the cost is set to zero for the remaining edges.

Equation (4.14), is similar to equation (4.11) but subtracts the revenues from potential trips in the subsequent period from the costs, depending on vehicle supply, using function $f_{ij}(x_{ij})$. Finally, the cost is set to zero for the remaining edges. Functions $h_{ij}(x_{ij})$ and $f_{ij}(x_{ij})$ in equations (4.11) and (4.14) are outlined in the following equations:

$$h_{ij}(x_{ij}) = -\phi_{ij} N^1_{n(i)n(j)}(x_{ij}) p r^1_{n(i)n(j)} + C_M r^1_{n(i)n(j)} x_{ij} \quad \forall (i,j) \in \mathcal{A} \tag{4.16}$$

$$f_{ij}(x_{ij}) = \sum_{m \in J} \Big(- N^2_{n(i)m}\Big(\frac{x_{ij}}{|J|}\Big) p r^2_{n(i)m} + C_M r^2_{n(i)m} x_{ij} \Big) \forall (i,j) \in \mathcal{B} \tag{4.17}$$

As observed in equations (4.16) and (4.17), both functions utilise $N^t_{ij}(x)$, which refers to the number of travellers choosing the service given a supply of vehicles x introduced in equation (4.5). Parameter p is the revenue per time for each vehicle as in equation (4.2), r^t_{ij} is the travel time between clusters i and j during epoch t as in equations (4.2) and (4.12). To accommodate vehicle mixing from redistribution, the number of travellers $N^t_{ij}(x)$ is factored in equation (4.16) by $\phi_{ij} \in [0, 1]$ according to the arrival sequence of redistributing vehicles in each cluster. A visual demonstration of how the demand factor ϕ_{ij} is calculated for each edge in $\mathcal{A}$ in a cluster is highlighted in Figure 4.6.

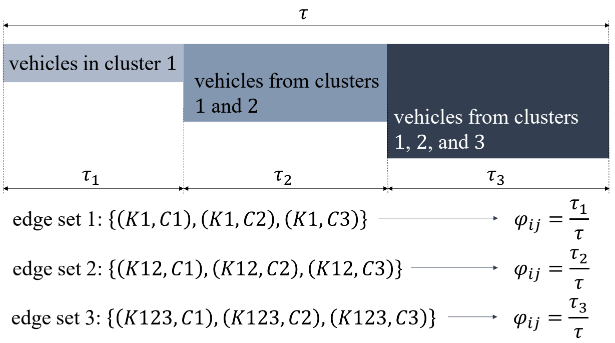

τ
vehicles in cluster 1
vehicles from clusters 1 and 2
vehicles from clusters 1, 2, and 3
τ_1
τ_2
τ_3
edge set 1: $\{(K1, C1), (K1, C2), (K1, C3)\}$
$\varphi_{ij} = \frac{\tau_1}{\tau}$
edge set 2: $\{(K12, C1), (K12, C2), (K12, C3)\}$
$\varphi_{ij} = \frac{\tau_2}{\tau}$
edge set 3: $\{(K123, C1), (K123, C2), (K123, C3)\}$
$\varphi_{ij} = \frac{\tau_3}{\tau}$

Figure 4.6: Example of how the demand factors ϕ are calculated in cluster 1 of an instance with 3 clusters and decision epoch of length τ.

The model further defines the lower bounds of all vertices to zero and unbounded edge capacities as follows:

$$l_{ij} = 0 \quad \forall (i,j) \in E \tag{4.18}$$

$$u_{ij} = \infty \quad \forall (i,j) \in E \tag{4.19}$$

The balance vectors for the source and sink vertices s and t are also defined as follows:

$$b(i) = S_i \quad \forall i \in A \tag{4.20}$$

$$b(t) = -\sum_{v \in A} b(v) = -\sum_{i \in A} S_i \tag{4.21}$$

Parameter S_i in (4.20) and (4.21) denotes the available vehicles S_i in cluster $n(i)$ at the start of the current period.

As such, the resource allocation problem introduced in section 4.3.1 can be solved using the following nonlinear minimum cost flow optimization problem:

Model 1:

$$\begin{aligned}
&\text{minimize} \quad \sum_{(ij) \in E} c_{ij}(x_{ij}) && && \text{(4.22a)}\\
&\text{subject to} && &&\\
&x_{ij} \geq l_{ij} && \forall (i,j) \in E, && \text{(4.22b)}\\
&x_{ij} \leq u_{ij} && \forall (i,j) \in E, && \text{(4.22c)}\\
&b(j) = \sum_{i:ji \in E} x_{ji} - \sum_{i:ij \in E} x_{ij} && \forall j \in V, && \text{(4.22d)}\\
&x_{ij} \in \mathbb{R} && \forall (i,j) \in E && \text{(4.22e)}
\end{aligned}$$

Equations (4.22b)-(4.22c) ensure the flow of vehicles through each edge (i,j) is within the specified lower and upper bounds respectively, as specified in equations (4.18) and (4.19). (4.22d) is the flow continuity constraint as specified in equation (4.7).

4.3.3 Convex Minimum Cost Flow Transformation

The objective function in (4.22a) is nonlinear, as a result of the costs for edges in $\mathcal{A}$ and $\mathcal{B}$. Such cost functions include the term $N_{ij}^t(x)$ introduced in (4.5), of which its component $q_{ij}^t(x)$ (eq. (4.4)) and sub-component $w_{ij}^t(x)$ (eq. (4.3)) are nonlinear. Identifying the nature of equation (4.16) is paramount for the choice of a solution method for *Model 1*. Classifying equation (4.16) would suffice as (4.17) is a linear sum of equation (4.16) instances.

According to section 3.4, equation (4.3) will result in maximum wait time up to the point of critical fleet size, when the number of idle vehicles $I_{ij}^t(x)$ will be sufficient to sustain the rate of incoming requests. Including this notion in the cost equations will result in increasing costs for trip edges up to the point of critical fleet size. Instead, to assist the solution process and reduce concavity, the following relaxed version of equation (4.3) is selected which can result in similar wait times for fleet sizes above the critical fleet size:

$$w_{ij}^t(x) = \frac{\alpha_i^t Z_{ij}^t}{x+1} \quad \forall i,j \in J, t \in T, x \in \mathbb{R}^+ \tag{4.23}$$

The parameter α_i^t can be determined by using linear least squares regression for specific values of Z_{ij}^t and pairs of x and w_{ij}^t identified via ABM. To aid the analysis, only one alternative ride-sourcing option is assumed, which offers identical pricing rates and travel times, with a fixed wait time $\bar{w}$. It is expected that the ride-sourcing platform prices its rides at a rate higher than its cost per time, such that $p > C_M$. Finally, the model assumes Z_{ij}^t is large enough to justify the cost of redistribution. For notation convenience, any index notation i, j, t is omitted in the proof of the following theorem.

Theorem 1. *$h_{ij}(x_{ij})$ has an absolute minimum point in $x_{ij} \in [0, \infty]$.*

Proof. First consider the limit of $h(x)$ (eq. (4.16)) as $x \to \infty$. Note that $\lim_{x\to\infty} w(x) = 0$, therefore, $\lim_{x\to\infty} g(x)$ is equal to some constant value $-r(\bar{v} + p)$. Consequently for $x \to \infty$, $q(x)$ converges to some finite maximum probability p_{max}. Since the upper bound of $q(x)$ is 1, $\lim_{x\to\infty} N(x) = Z$. It is therefore straightforward to deduce the following limit:

$$\lim_{x\to\infty} h(x) = \infty \tag{4.24}$$

Now consider the limit of $h(x)$ for $x \to 0^+$. For a large Z, $\lim_{x\to 0^+} w(x) = \alpha Z$, $\lim_{x\to 0^+} g(x) = -\bar{u}(\alpha Z + r) - pr$. Due to the exponential nature of $q(x)$, for small values of x (i.e. $x = 1$ for large Z), $\lim_{x\to 0^+} q(x) = 0$ and consequently $\lim_{x\to 0^+} N(x) = 0$. Therefore the following result is reached:

$$\lim_{x\to 0^+} h(x) = 0^+ \tag{4.25}$$

Let now the case of $q(x) = 0.5$ to be explored. For $q(x) = 0.5$, $w(x) = \bar{w}$, therefore by rearranging the terms of $w(x)$, for $w(x) = \bar{w}$, then $x = \frac{\alpha}{\bar{w}} Z - 1$. Since $q(x) = 0.5$, equation (4.16) results to

$h(x) = -0.5Zpr + C_M rx$. Substituting x with $\frac{\alpha}{\bar{w}}Z - 1$, and simplifying the following equation is identified:

$$h(x) = r\Big(-0.5Zp + C_M(\frac{\alpha}{\bar{w}}Z - 1)\Big) \tag{4.26}$$

The model assumes that the fleet operator chooses p such that $p > C_M$ and α can be scaled, such that $\alpha < \bar{w}$ and $-0.5Z > \frac{C_M}{p}(\frac{\alpha}{\bar{w}}Z - 1)$. Thus, by implementing the above inequalities, for $q(x) = 0.5$, the following relationship holds:

$$h(x) < 0 \quad \forall x \in \mathbb{R}^+ | q(x) = 0.5 \wedge \frac{C_M \alpha}{p\bar{w}} \leq 0.5 \tag{4.27}$$

Using equations (4.24) and (4.25), and by showing that $h(x)$ is negative for some $x \in \mathbb{R}^+$, the analysis concludes that $h(x)$ has an absolute minimum point in $x \in [0, \infty]$.

□

Corollary 1. *$h_{ij}(x_{ij})$ is convex for some domain $x_{ij} \in [x'_{ij}, x^*_{ij}]$. Where $h_{ij}(x^*_{ij})$ is the absolute minimum value of $h_{ij}(x_{ij})$ for $x_{ij} \in \mathbb{R}^+$ and x'_{ij} is the largest value of x_{ij} such that $h(x'_{ij})$ is a non-stationary inflection point and $x'_{ij} < x^*_{ij}$.*

The aim of this analysis is to identify and utilize the convexity of the domain $[x'_{ij}, x^*_{ij}]$ of each non-linear edge cost function to solve *Model 1* as a convex minimum cost flow problem. In line with convexity, the upper bounds u_{ij} of each non-linear edge (i, j) are replaced to the absolute minimum value x^*_{ij}. Therefore additional to equation (4.19), the model introduces the following:

$$u_{ij} = x^*_{ij} \quad \forall (i, j) \in \mathcal{A} \cup \mathcal{B} \tag{4.28}$$

In a similar fashion, setting the lower bound l_{ij} of any non-linear edge (i, j) to the inflection point x'_{ij}, would restrain the non-linear cost functions to the convex domain. Nonetheless, the model refrains setting lower bounds to the problem to avoid potential infeasibility of *Model 1*. Instead, each non-linear cost function is split to a piece-wise one, with a linear part between $[0, x'_{ij}]$, and non-linear convex part between $[x'_{ij}, x^*_{ij}]$. $h_{ij}(x_{ij})$ is linearised between $[0, x'_{ij}]$ to avoid any concave parts of the cost misguiding the solution algorithm (Section 4.3.4) towards non-optimal solutions.

As it can be observed in the next section (Section 4.3.4), the proposed solution algorithm identifies optimal solutions of convex functions by incrementally moving from higher absolute values of the cost derivative $\frac{dh_{ij}(x_{ij})}{dx_{ij}}$ towards values where the derivative approaches zero $\big(\frac{dh_{ij}(x^*_{ij})}{dx_{ij}} = 0\big)$. Consequently, the equation of the linearised part $h^L_{ij}(x_{ij})$ between $[0, x'_{ij}]$ for $h_{ij}(x_{ij})$ is set to the equation of the tangent of $h_{ij}(x_{ij})$ at x'_{ij} follows:

$$h^L_{ij}(x_{ij}) = \frac{dh_{ij}(x'_{ij})}{dx_{ij}}x_{ij} - \frac{dh_{ij}(x'_{ij})}{dx_{ij}}x'_{ij} + h_{ij}(x'_{ij}) \tag{4.29}$$

Similarly, by performing the same procedure for $f_{ij}(x_{ij})$ in equation (4.17), the linearised part of the cost has the following form:

$$f^L_{ij}(x_{ij}) = \frac{df_{ij}(x'_{ij})}{dx_{ij}}x_{ij} - \frac{df_{ij}(x'_{ij})}{dx_{ij}}x'_{ij} + f_{ij}(x'_{ij}) \tag{4.30}$$

The model thereby introduces the following convex cost functions to replace equations (4.11) and (4.14) with equations (4.31) and (4.32) respectively:

$$c_{ij}(x_{ij}) = h^C_{ij}(x_{ij}) \quad \forall (i,j) \in \mathcal{A} \tag{4.31}$$

$$c_{ij}(x_{ij}) = f^C_{ij}(x_{ij}) \quad \forall (i,j) \in \mathcal{B} \tag{4.32}$$

The functions $h^C_{ij}(x_{ij})$ and $f^C_{ij}(x_{ij})$ in equations (4.31) and (4.32) respectively have the following form:

$$h^C_{ij}(x_{ij}) = \begin{cases} h^L_{ij}(x_{ij}) & x_{ij} \leq x'_{ij} \\ h_{ij}(x_{ij}) & x_{ij} > x'_{ij} \end{cases} \tag{4.33}$$

$$f^C_{ij}(x_{ij}) = \begin{cases} f^L_{ij}(x_{ij}) & x_{ij} \leq x'_{ij} \\ f_{ij}(x_{ij}) & x_{ij} > x'_{ij} \end{cases} \tag{4.34}$$

As such, by replacing equations (4.11) and (4.14) with equations (4.31) and (4.32) respectively and adapting the upper bounds of equation (4.28) for non-linear edges, *Model 1* becomes a Convex Minimum Cost Flow (CMCF) optimization problem.

4.3.4 Edge-Splitting Pseudo-Polynomial Algorithm

It was previously mentioned that it is possible to transform the vehicle redistribution problem into a CMCF problem. The flow x_{ij} is a discrete quantity in the model as it considers the count of vehicles in each link (i,j). Linear minimum cost flow models adhere to the following theorem, as stated in [124]:

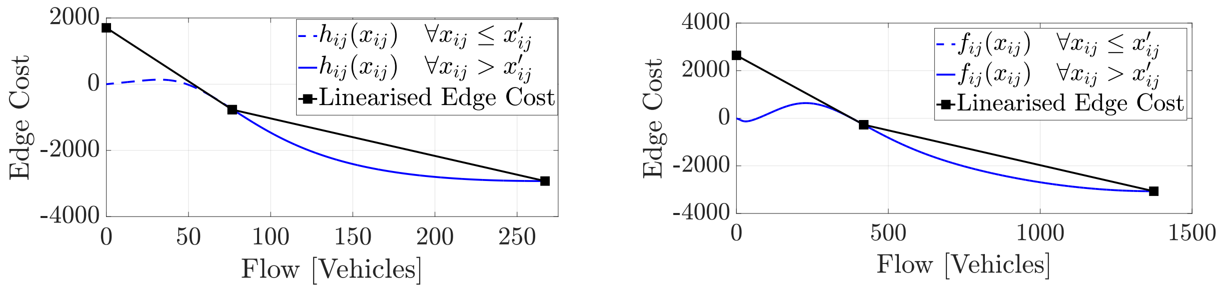
Theorem 2. *(Integrality Theorem) If the capacities of all edges and the balance values of all the nodes are integer, the linear minimum cost flow problem always has an integer optimal flow.*

For proof of the above theorem, readers should refer to [124]. It is thus deduced by restricting the parameters of the problem to integers (capacities and balance vectors), one can solve the linear minimum cost flow problem in polynomial time. The aim of this section is to exploit the integrality theorem, using an appropriate linearisation technique, to solve the CMCF problem transformation of *Model 1* in polynomial time.

The CMCF problem has been previously tackled efficiently in the literature. [125] initially proposed an extension of the scaling method for linear minimum cost flows presented in [126], for convex cost flows with quadratic functions. At a subsequent stage, [127] and [128] separately conducted studies on solving minimum cost flows with general convex objectives. A variant of the algorithm proposed by [127], and along the lines of [128], is featured in [124]. [129] proposed polynomial algorithms for solving the CMCF problem in circles, lines or trees. The problem of quadratic CMCF was also tackled more recently in [130], using an enhanced version of [127], utilizing the technique for linear minimum cost flows proposed in [131].

A consistent assumption of the studies which efficiently address the CMCF problem is that the edge costs are non-negative. Nonetheless, as it has been observed in Sections 4.3.2 and 4.3.3, the non-linear cost functions of *Model 1* can have negative values. The notion of negative costs (i.e. profit), implies that in an optimal solution of *Model 1*, there would be edges with negative costs. As such, this section refrains from using the above algorithms, and instead, incorporates a modified version of the pseudo-polynomial algorithm for CMCF presented in [124].

The algorithm applies piece-wise linearisation by introducing parallel linear edges for each non-linear edge in the network. To limit the amount of additional parallel edges in the network, the algorithm starts with only two parallel edges for each non-linear edge (i, j). As observed in equations (4.33) and (4.34), $h_{ij}^{C}(x_{ij})$ and $f_{ij}^{C}(x_{ij})$ are partly linearized (i.e. for $x \leq x'_{ij}$). Consequently, by replacing $h_{ij}(x_{ij})$ and $f_{ij}(x_{ij})$ with their linearised versions between x'_{ij} and x^{*}_{ij}, the algorithm is initiated with two parallel linear edges for each non-linear edge of *Model 1*. Figures[2] 4.7a and 4.7b outline the initial linearisation of costs for non-linear edges $(i, j) \in \mathcal{A}$ and $(i, j) \in \mathcal{B}$ respectively.



(a) (b)

Figure 4.7: Cost function variation for edges $(i, j) \in \mathcal{A}$ (a) and $(i, j) \in \mathcal{B}$ (b).

As observed in figures 4.7a and 4.7b, due to convexity, the slope of each parallel linear edge from left to right gradually increases, from a minimum negative value to zero, while moving from $x_{ij} = 0$ to $x_{ij} = x^{*}_{ij}$. Consequently, per unit flow is more expensive through the parallel linear edge corresponding

[2] For figures 4.7a and 4.7b the following parameters were used: $\bar{v} = 0.3$, $\alpha = 2$, $\bar{w} = 5$, $p = 1$, $C_M = 0.3$.

to the non-linear section for $x_{ij} > x'_{ij}$. As such, assuming both parallel edges have residual capacity, flow through parallel edges with smaller slope is always prioritised over edges with larger slope value (i.e. closer to zero). This observation is described as the property of contiguity in [124].

Utilizing contiguity, for each parallel linear edge, the upper bound (capacity) is set to the difference between the right and left flow boundaries of the linearised edge, while maintaining zero lower bounds. Consequently, for the initial linearisation configuration, the upper bounds will be $u^1_{ij} = x'_{ij} - 0$ and $u^2_{ij} = x^*_{ij} - x'_{ij}$ from left to right respectively, as observed in figures 4.7a and 4.7b. Since the algorithm is initiated with two parallel linear edges per non-linear edge, their linear cost functions are denoted as c^1_{ij} and c^2_{ij} for edges corresponding to $x \leq x'_{ij}$ and $x_{ij} > x'_{ij}$ respectively. The transformation of each non-linear edge to a pair of linearised ones is shown in figure 4.8.

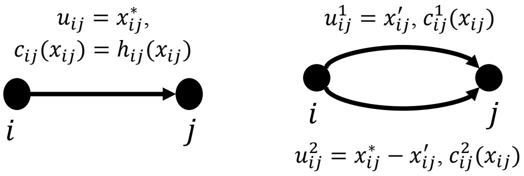

$u_{ij} = x^*_{ij}$,
$c_{ij}(x_{ij}) = h_{ij}(x_{ij})$
i
j
$u^1_{ij} = x'_{ij}, c^1_{ij}(x_{ij})$
i
j
$u^2_{ij} = x^*_{ij} - x'_{ij}, c^2_{ij}(x_{ij})$

Figure 4.8: Linear transformation of a non-linear edge (i, j) to linear edges $(i, j)^1$ and $(i, j)^2$ with resulting linear costs c^1_{ij} and c^2_{ij} respectively.

To maintain the validity of Theorem 2, the integer versions of x'_{ij} and x^*_{ij} are considered, such that the resulting capacities u^1_{ij} and u^2_{ij} are also integers. Furthermore, to identify point x'_{ij} for each non-linear edge in *Model 1*, the first and second derivatives of the non-linear cost are approximated using the forward and central difference formulas respectively.

The function $P(i, j)$ is introduced which identifies the set of parallel edges between any pair of vertices $i, j \in V$. The linearised version of $G = (V, E)$ is referred to as $G_L = (V, E_L)$. The non-linear edge linearisation of *Model 1* up to this point can be described via the following formulation:

Model 2:

$$\text{minimize} \quad \sum_{(ij)\in E} \sum_{k=1}^{P(i,j)} c^k_{ij}(x^k_{ij}) \tag{4.35a}$$

$$\text{subject to}$$

$$x^k_{ij} \geq l^k_{ij} \qquad \forall (i, j) \in E, \forall k \in P(i, j), \tag{4.35b}$$

$$x^k_{ij} \leq u^k_{ij} \qquad \forall (i, j) \in E, \forall k \in P(i, j), \tag{4.35c}$$

$$b(j) = \sum_{i:ji\in E} \sum_{k=1}^{P(j,i)} x^k_{ji} - \sum_{i:ij\in E} \sum_{k=1}^{P(i,j)} x^k_{ij} \quad \forall j \in V, \tag{4.35d}$$

$$x^k_{ij} \in \mathbb{R} \qquad \forall (i, j) \in E, \forall k \in P(i, j) \tag{4.35e}$$

A high level structure of the CMCF algorithm is outlined in Algorithm 3. Algorithm 3 utilizes graphs G

and G_L, and the structure of *Model 2* to identify the set of minimum cost flows F, such that $x_{ij}^k \in F$, with $(i,j)^k \in E_L$. Algorithm 3 incorporates an iterative procedure to arrive at the optimal solution for the CMCF of graph G.

Algorithm 3 CMCF Edge-Splitting Algorithm

```
Inputs: Model 2, graph $G_L$ and graph $G$
$OPT =$ false
while $OPT =$ false do
    $F \leftarrow \emptyset$
    $F = NetworkSimplex(F, G_L, \textit{Model 2})$
    $U = \textit{SplittableEdges}(F, G_L)$
    if $U = \emptyset$ then
        $OPT =$ true
    else
        for $(i,j)^k \in U$ do
            $G_L = \textit{Split}(G, G_L, (i,j)^k)$
        end for
    end if
end while
Output: Flow $F$
```

To facilitate the edge-splitting algorithm, the boolean variable OPT is defined with the default value set to false, which signals the algorithm to stop if an optimal solution is found during an iteration (i.e. if OPT is true). At the beginning of each iteration, the network simplex algorithm is used [124] to solve the minimum cost flow problem and obtain a set F of flows x_{ij}^k. The algorithm thereby screens through flows $x_{ij}^k \in F$ using the function $\textit{SplittableEdges}(F, G_L)$, to identify the set U of linearised parallel edges which are subject to further splitting. If U is an empty set, the set F of flows x_{ij}^k is an optimal solution to the CMCF version of *Model 1*. Otherwise, the algorithm proceeds with splitting each edge $(i,j)^k \in U$ using the routine $\textit{Split}(G, G_L, (i,j)^k)$, which updates the linearised graph G_L and move to the next iteration.

The function $NetworkSimplex(F, G_L, \textit{Model 2})$ is used to denote the procedure of solving *Model 2* using network simplex and populating set F. Presentation of the network simplex algorithm is omitted as it is well known in the literature. The routine followed for the $\textit{SplittableEdges}(F, G_L)$ function is outlined in Algorithm 4. As observed in Algorithm 4, the routine investigates the flow of each parallel edge $(i,j)^k$ of graph G_L which exists in the convex domain of edge (i,j) of graph G (i.e. $k > 1$). If the edge $(i,j)^k$ is the first edge in (i,j) which is not at capacity and can be divided, the process adds edge $(i,j)^k$ to the set U.

The $\textit{Split}(G, G_L, (i,j)^k)$ function used in Algorithm 3 is outlined in Algorithm 5. Initially, the algorithm identifies the corresponding upper (x_U) and lower (x_L) flow values of edge $(i,j)^k$ in the convex domain of edge (i,j). Since the flow F is contiguous, the algorithm can find x_U and x_L by finding the total flow in (i,j), and the total flow in (i,j) excluding x_{ij}^k respectively. The algorithm then identifies the split point x_S as the midpoint[3] of x_U and x_L. The algorithm thus removes the edge $(i,j)^k$ and append

[3] $\lceil X \rceil$ denotes the ceiling function.

$P(i, j)$ with the indices of the two new parallel edges to be added. For each new parallel edge, the process finds its upper and lower bound, as well as its cost function in a similar fashion as described earlier in the construction of G_L before initialising Algorithm 3. The split function is concluded by adding the two new parallel edges in E_L and returning the updated graph G_L.

Algorithm 4 SplittableEdges Function

```
Inputs: Set $F$ of flows $x_{ij}^k$, graph $G_L$
$U \leftarrow \emptyset$
for $(i, j) \in ((K \times C) \cup (D \times t))$ do
    for $k \in P(i, j) \setminus k = 1$ do
        if $(l_{ij}^k \leq x_{ij}^k < u_{ij}^k) \cup (x_{ij}^{k-1} = u_{ij}^{k-1})$ then
            if $(u_{ij}^k > 1)$ then
                $U \leftarrow U \cup (i, k)^k$
            end if
        end if
    end for
end for
Output: Set $U$ of splittable edges
```

Algorithm 5 Split Function

```
Inputs: Graph $G$, graph $G_L$ and edge $(i, j)^k$
$x_U = \sum_{n \in P(i,j)} x_{ij}^n$
$x_L = \sum_{n \in P(i,j) \setminus n = k} x_{ij}^n$
$x_S = \lceil \frac{x_{ij}^U + x_{ij}^L}{2} \rceil$
$E_L \leftarrow E_L \setminus (i, j)^k$
$n = max(P(i, j))$
$P(i, j) \leftarrow P(i, j) \cup (n + 1) \cup (n + 2)$
$u_{ij}^{n+1} = x_S - x_L$
$u_{ij}^{n+2} = x_U - x_S$
$l_{ij}^{n+1} = 0$
$l_{ij}^{n+2} = 0$
Define $c_{ij}^{n+1}(x)$ by finding the linear equation between points $[x_L, c_{ij}(x_L)]$ and $[x_S, c_{ij}(x_S)]$.
Define $c_{ij}^{n+2}(x)$ by finding the linear equation between points $[x_S, c_{ij}(x_S)]$ and $[x_U, c_{ij}(x_U)]$.
$E_L \leftarrow E_L \cup (i, j)^{n+1} \cup (i, j)^{n+2}$
Output: Updated linearised graph $G_L$
```

The rationale behind the solution method, as described in Algorithms 3-5, is that the algorithm keeps splitting linearised edges in the convex domain until it satisfies some optimality conditions. Specifically, the process obtains the optimal solution when all the flows in each of the linearised edges in this domain are either zero or equal to the upper bound, and no further splitting can induce incremental cost savings.

If the total input flow (i.e. $\sum_{i \in A} S_i$) is large enough, Algorithm 3 allocates the upper bound flow in each of the non-linear edges of G due to their negative costs (profitable edges). The case described above would terminate after the first iteration with the optimal solution of the CMCF version of *Model 1*. Otherwise, for each non-linear edge, the first parallel edge which is not at capacity is split into two parts. For any split edge, due to convexity, the flow in the next iteration would always be confined within the resulting pair of parallel edges.

As a result of the above description, Algorithm 3 terminates when for each non-linear edge, the splitting procedure produces parallel edges of unit capacity (i.e. $u_{ij}^k = 1$). Consequently, the number of iterations is logarithmic and relates to the maximum interval $x_{ij}^* - x_{ij}'$ out of all the non-linear edges. If this maximum interval is denoted as Δ, the procedure needs to solve the minimum cost flow problem $\log_2(\Delta)$ times, adding at most $|J|^3 + |J|$ parallel edges to G_L at each iteration.

The cardinality m of the set of edges E can be expressed in terms of $|J|$; hence each network simplex run is polynomially bounded by the number of variables of the original problem. However, Δ cannot be expressed by the number of variables in the network G. Consequently, Algorithm 3 runs in pseudo-polynomial time. Nonetheless, even in extreme practical cases $\log_2(\Delta)$ is a small number (i.e. for $\Delta = 10000$, $\log_2(\Delta) \approx 13$), hence the algorithm can be applied in practical implementations.

4.4 Discussion

The effectiveness of the redistribution algorithm proposed in Section 4.3 was tested in a simulated ride-sourcing fleet operator using the agent-based modelling framework with a FIFO customer assignment policy proposed in Chapter 3. Customer choices in the simulator were decided on an individual level using equations (4.2) and (4.4) with individual wait and travel times identified during the simulation. Additional to the framework in 3, an empty relocation state was included for vehicles, during which vehicles can still be assigned to customers.

The algorithmic methodology was implemented in Python, which served as the fleet management logic in the agent-based model and tested on a workstation with an Intel i7-4790 CPU (3.6GHz) and 8GB RAM. The IBM Cplex solver was used to obtain network simplex solutions for *Model 2*.

4.4.1 Model Instance

The area of Manhattan, NYC was selected, to apply a case study of the algorithm due to the comprehensive trip data-set available in [119] which served as the demand input. Travel times in the network were calculated using the OSMnx library [118]. By assuming a small proportion of traffic attributes to ride-sourcing, endogenous congestion was omitted in the agent-based model. Nonetheless, exogenous congestion was accounted by applying a 20% penalty to the free-flow speeds in residential and motorway link segments, and 40% elsewhere during peak hours.

Using the data-set in [119], typical demand profiles were created for weekdays in Manhattan, NYC from 05:00 am to 12:00 am. K-means clustering was used to split the road-network into twenty clusters, with a universal value of α (equation (4.23)) for convenience. To benchmark the performance of

Table 4.7: Instances tested in the simulation.

ID	Method	Clusters	τ [min]	α
a_0	Model 2	20	30	0.75
a_1	Model 2	20	15	0.75
a_2	Model 2	20	30	1.5
a_3	Model 2	20	30	0.75
b_1	LP	20	30	
b_2	LP	20	15	
b_3	LP	20	5	

the proposed algorithm in Section 4.3, the study tested it against the case of no redistribution and also against a linear programming (LP) method from the state-of-the-art which does not assume any supply-demand elasticity, namely the rebalancing model in [60]. The study in [60] was selected, as it has been tested against other central rebalancing methods such as [107]. All algorithms were tested using different fleet sizes from 2500 to 15000 vehicles.

The study used UK estimates of the value of time $\bar{v}$ and vehicle moving costs C_M from [132] as information on values of time for New York was not available. Consequently, the average value of time $\bar{v}$ was set to 17.69 GBP/hour. The vehicle moving cost C_M was set to the conservative estimate of 12.96 GBP/hour to reflect current driver valuations. The idle vehicle cost C_I was set to 1 GBP/hour to account for parking costs, whereas the price per minute for a ride p was set to 1.00 GBP/min to reflect previous research on AV pricing [122]. The value of $\bar{w}$ was set to 5 (minutes).

The proposed algorithm performs allocations based on demand expectations for two subsequent periods (Z^1_{ij} and Z^2_{ij}). Since the application of predictive algorithms is beyond the scope of this study, it is assumed that the platform has complete knowledge of the demand in the two subsequent periods for each cluster when applying the proposed algorithm and the model in [60]. Nonetheless, to scrutinise how demand prediction accuracy affects the efficacy of the algorithm, the study also considered an instance with a normally distributed error with mean 20% and standard deviation 10% on the total demand Z^t_{ij}.

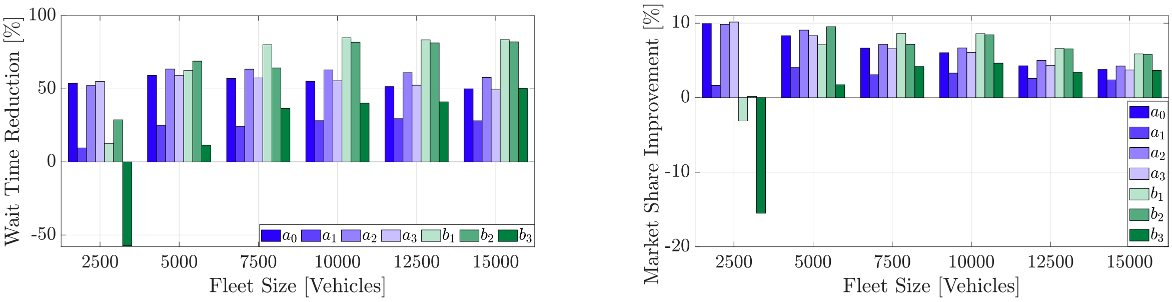
To investigate the robustness of the proposed method and the effectiveness of the window length τ, instances with different values of α and τ, respectively were created. The tested instances and parameters are outlined in table 4.7. Where LP in table 4.7, refers to the model of [60] and a_3 represents the version of *Model 2* with a random error on total demand Z^t_{ij}.

4.4.2 Analysis

The analysis used four different Key Performance Indicators (KPIs) to compare the effectiveness of each set-up outlined in table 4.7 with the case of no redistribution. Specifically, the study considered wait time reduction, market share improvement, profit improvement and additional mileage, expressed as percentage shifts from the no redistribution benchmark.

As observed in figures 4.9-4.10, frequent redistribution degrades the efficacy of all tested methods. *Model 2* was found to be ineffective for $\tau = 15$ minutes, whereas the LP benchmark algorithm presented noticeable degradation for a window length of 5 minutes. This ineffectiveness is expected, as both algorithms use central optimization and a refined window length deteriorates the quality of aggregation. It is, therefore, deduced that *Model 2* is best fitted for low frequency in redistribution (i.e. every 30 minutes).

As observed in figure 4.9a, vehicle redistribution via the proposed method can reduce the average customer wait time up to more than 50% when compared to no redistribution. Furthermore, when considering market share, *Model 2* contributes to an increase of up to 10%, as shown in figure 4.9b. However, as the fleet size increases to larger values, the contribution of redistribution towards higher market share and reduced wait times becomes marginal due to the abundance of available vehicles across the network.



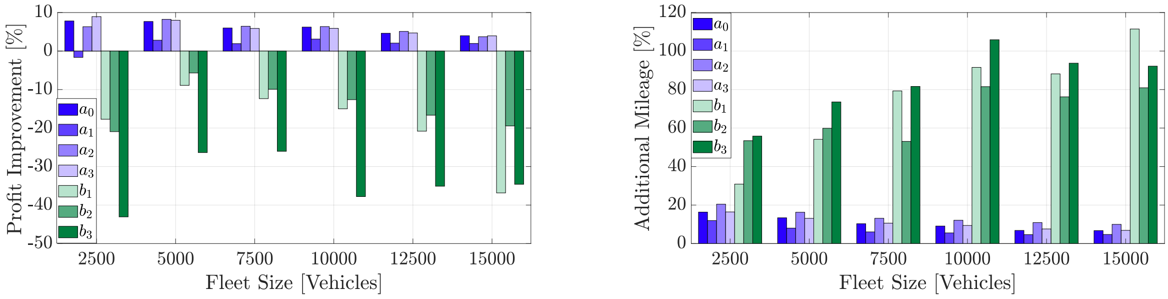
(a)

(b)

Figure 4.9: Average wait time reduction of customers (a) and market share improvement (b) for different fleet sizes and redistribution strategies.

Comparing the results of the proposed method with the algorithm proposed in [60] and ignoring smaller window lengths (a_1 and b_3), it is observed that in many cases (above 2500 vehicles) the LP method outperforms *Model 2* in reducing wait times and improving market share (figures 4.9a and 4.9b). Nonetheless, as the *Model 2* is modelled to minimise cost based on expected customer choices, the effectiveness of the proposed algorithm over LP is clearly stated in profit improvement and additional mileage as observed in figures 4.10a and 4.10b respectively.

Observing figures 4.10a and 4.10b, it can be deduced that *Model 2* can achieve a sizeable profit increase of up to 10% with an additional mileage of 20% across the network. On the contrary, the LP algorithm achieves much lower wait times and higher market share in the expense of considerable additional mileage and a significant profit reduction (more than 10% in most cases). This result is anticipated, as there is no notion of diminishing returns for additional redistribution in the LP method, which is the reason for non-linearity in *Model 2*.



(a)

(b)

Figure 4.10: Profit improvement (a) and additional mileage (b) for different fleet sizes and redistribution strategies.

4.4.3 Practical Implementation

To assess the usefulness of *Model 2* in practical implementations, the study tests its robustness with regards to input functions such as equation (4.23) and uncertainty in demand estimation. It is observed that varying α in equation (4.23) from 0.75 to 1.50 does not have a significant effect on the resulting redistribution. As (4.23) is only used by the operator to estimate the effect of vehicle supply on the average wait time, variations of the wait time function can change the capacity of trip links in *Model 2*. Nonetheless, as the wait time change for either value is marginal when considering the large sizes of incoming demand and available vehicles in (4.23), so is the change across all KPIs when comparing instance a_0 to instance a_2.

Furthermore, when comparing the effect of a randomly distributed error on demand estimation (instance a_3), the performance of a_3 is consistently similar to both a_0 and a_2 across all KPIs. This similarity is most likely present due to setting an upper bound on the capacity of trip links in *Model 2* and allowing for idle vehicles in each cluster to be available for assignment in-between redistribution decisions.

Finally, to confirm the efficiency of Algorithm 3, which solves *Model 2*, its runtime was measured for each fleet size in scenario a_0. The results outline that in the worst case, the proposed algorithm requires an average of 74.7 seconds to identify the optimal redistribution, which is considered acceptable if a central fleet redistribution decision is made in windows of 30 minutes. The average runtimes for all tested fleet sizes are outlined in table 4.8.

Table 4.8: Algorithm 3 average runtimes for a_0 scenario and various fleet sizes.

Fleet Size	2500	5000	7500	10000	12500	15000
Runtime [sec]	74.7	23.8	23.8	24.2	24.1	23.8

4.5 Summary

This chapter explored how to efficiently optimize fleet management operations in ride-sourcing, focusing on the tactical time-horizon. Initially, a straw-man linear programming model for fleet allocation was presented to provoke discussion on the impact of non-linearity and problem structure. The model was tested in the simplified Sioux Falls network, and results showed that two main caveats. First, the acumen in customer behaviour needs to be accounted for to produce more realistic demand-supply relations. Second, a more sophisticated programming model structure is required to facilitate efficient implementations in realistic, large scale networks.

Consequently, the study then used network theory to transform the vehicle redistribution problem into a CMCF problem with negative costs, accounting for customer behaviour under an assumed market structure. The inclusion of customer acumen in the model and the transformation of the redistribution problem to the problem of minimum cost flow addressed the straw-man model's caveats. The proposed edge-splitting algorithm solves the CMCF problem exactly in pseudo-polynomial time by allocating vehicles to spatio-temporal tasks.

The work in this chapter demonstrated the practicability of the redistribution algorithm in an agent-based model simulating ride-sourcing in a large urban setting, such as Manhattan, NYC. The fleet management algorithm proposed in the form of the CMCF problem provides efficient large scale solutions to supply-demand imbalance. Nonetheless, operational time-horizon decision problems in ride-sourcing, such as pricing and assignment, if employed effectively, can provide further relief in the system when supply-demand asymmetries prevail. Therefore, in the next chapter, the study will investigate the interplay between the pricing and assignment problems in ride-sourcing, and how the decision-making concerning these problems can be designed efficiently.

Chapter 5

Pricing and Assignment Algorithms in Peak Demand Periods for Ride-Sourcing Fleets

5.1 Introduction

The previous chapter highlighted that effective allocation of a fleet across activities in the network using fleet management approaches could better utilise the fleet and result in higher profit margins. Such approaches are heavily reliant on assignment methodologies and pricing strategies which the operator follows. Inevitably, assignment decisions are identified in a more refined time-horizon than the fleet management strategies presented in Chapter 4. As an example, the model proposed in Section 4.3 in the previous chapter was calculated in an online fashion every 30 minutes of the fleet's operation; however, the operator would assign requests to vehicles at an almost continuous rate within the 30 minute time horizon for fleet management.

Furthermore, considering the pricing of rides, in Chapter 4, the fleet management strategies presented assume a static pricing strategy for a proof of concept. However, as discussed in section 2.3, a common practise of ride-sourcing firms is to apply dynamic pricing at periods of peak travel to balance the supply of vehicles to the demand for rides. Additionally, as noted in section 2.3 CDAs can be used as a method to achieve both assignment and pricing of multiple rides to vehicles simultaneously and efficiently as an alternative to dynamic pricing methods during periods of peak travel. Such pricing and assignment methodologies, by design, vary with fluctuations in the volume of incoming requests across a road-network, typically, in a more refined time-horizon than the one used for the fleet management models in Chapter 4.

As such, this chapter focuses on the lowest level of operations for a ride-sourcing fleet, proposing methodologies for both pricing and assignment of rides. So far, the straightforward version of FIFO assignment for private trips was used in both Chapters 3 and 4. Nonetheless, in practice, operators utilise more sophisticated versions which can perform better, either by implementing quasi-online assignments and/or offering a more comprehensive array of services such as shared trips, which can positively impact traffic.

By using a similar structure as in Chapter 4, this chapter first presents some rudimentary examples in Sections 5.2 and 5.3, to outline the potential impact of trip sharing and dynamic pricing, and to indicate how such methods can behave in competitive markets respectively. Both examples use the ABM framework introduced in Chapter 3 to identify impact.

Building on the practical implementations of both examples presented in Sections 5.2 and 5.3, Section 5.4 then presents an efficient method for assignment and pricing of shared rides using combinatorial double auctions. The model incorporates the operator's service time as a trading asset and fits a computational complexity framework that enables fast and efficient calculation, a requirement when considering the refined time horizon of the operation. Section 5.5 concludes this chapter with a practical implementation of the auction model presented in Section 5.4.

5.2 Dynamic Pricing Example in One-Sided Autonomous Ride-Sourcing Markets

The review of the literature in section 2.3 reveals that while there has been extensive analysis of dynamic pricing on two-sided ride sourcing platforms, there is limited understanding of its implications when TNCs stop being two-sided markets (when AVs enter the market). To address this issue, this section, tests a utility-based dynamic pricing model in an Autonomous TNC (ATNC) market, with private and shared ride services, in competition and monopoly, with an alternative public transport mode using an ABM.

5.2.1 Model Description

The actors in the model are ATNCs that offer private and shared ride services of identical quality. They receive ride requests from the public and respond with bids that reflect the current state of their fleets, and the expected wait and travel times. Travellers decide whether to accept a bid or choose public transport using a generalised costing mechanism.

In the monopolistic scenario, a single ATNC applies a dynamic or a static pricing model for its bids, whereas, in the competitive scenario, we consider a duopoly with a dynamic pricing ATNC and a static pricing ATNC. The static pricing model sets the static ride price p_s as the sum of linear terms consisting of a base fare f_b, a time proportional fare to ride time t with a rate per time f_t, and a distance proportional fare to ride distance d with a rate per distance f_d:

$$p_s = f_b + f_t t + f_d d \tag{5.1}$$

The dynamic pricing model determines the price p_d using the static price p_s and a dynamic multiplier m.

$$p_d = m p_s \tag{5.2}$$

ATNCs that utilise dynamic pricing models are expected to set the value of m with the objective to maximise the expected revenue for each traveller using the utilities of the available travel options as inputs. Each traveller in the model is therefore assumed to evaluate the utility of each option using a nested logit model. Traveller utilities for each option are calculated using (5.3) for private ATNC rides U_{P_i} of each firm i, (5.4) for shared ATNC rides U_{S_i} of each firm i and (5.5) for public transport U_{PT}.

$$U_{P_i} = V_{P_i} + \varepsilon_{P_i} = \alpha - \nu(w_{P_i} + t_P) - p_{P_i} + \varepsilon_{P_i} \quad \forall i \in I \tag{5.3}$$

$$U_{S_i} = V_{S_i} + \varepsilon_{S_i} = \alpha - \nu(w_{S_i} + t_S) - p_{S_i} + \varepsilon_{S_i} \quad \forall i \in I \tag{5.4}$$

$$U_{PT} = V_{PT} + \varepsilon_{PT} = -\nu(w_{PT} + t_{PT}) - p_{PT} + \varepsilon_{PT} \tag{5.5}$$

Where ν is the value of time for each traveller which is assumed to vary across the traveller population following a gamma distribution with shape factor k and scale $\bar{\nu}/k$. The parameter α is used to represent the inherent preferences for the ATNCs due to unobserved factors such as comfort, brand image and trust and is assumed to vary across the traveller population following a normal distribution with standard error σ_α. I is the set of ATNCs in the model, which is $\{1\}$ in monopoly and $\{1, 2\}$ in duopoly. Parameters w_{P_i}, w_{S_i} and w_{PT} represent the waiting times for trips using each travel option. In turn, t_P, t_S and t_{PT} are the travel times for each mode. Parameters p_{P_i} and p_{S_i} represent the price for private and shared rides of each ATNC i and p_{PT} is the price of travelling via public transport.

The stochastic error terms ε_{P_i}, ε_{S_i} and ε_{PT} are randomly distributed variables following a type 1 extreme value distribution, which is the assumption underlying the nested logit model structure. The latter assumes a heightened correlation between the stochastic error terms for the ATNCs thus allowing for a higher elasticity between the ATNC alternatives. The travel and waiting times for each option vary between travellers and depend on network characteristics and vehicle distributions across the network at the time of the request.

This model adopts a three-level nested logit model [133], illustrated by the tree diagram in Figure 5.1. The levels used in the model are defined as follows:

- Level 1: Choice between ATNC and Public transport
- Level 2: Choice between private or shared ATNC rides
- Level 3: Choice of ATNC

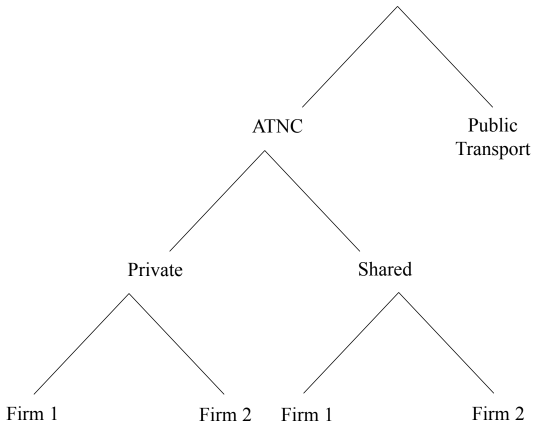


Figure 5.1: Three-level nested logit diagram.

The choice probabilities between the modes of ATNC T and public transport PT are computed as follows:

$$P(b) = \frac{e^{\mu\ V_b}}{e^{\mu\ V_T} + e^{\mu V_{PT}}} \qquad \forall\, b = \{T,\ PT\} \tag{5.6}$$

Where μ is the scale of the stochastic error terms, assumed to be 1 between the first-level options (ATNCs versus public transport). The probability of choosing between private P or shared ride S services is given by:

$$P(T_j) = P(T_j|T)P(T) \qquad \forall\, j = \{P,\ S\} \tag{5.7}$$

$$P(T) = \frac{e^{\mu\ V_T}}{e^{\mu\ V_T} + e^{\mu V_{PT}}} \tag{5.8}$$

$V_T = IV_T$ is the inclusive value of the ATNC nest and is calculated using equation (5.9):

$$IV_T = \frac{1}{\mu_T} \ln \left[\sum_j e^{\mu_T \times V_{T_j}} \right] \tag{5.9}$$

Where μ_T is the scale of the error terms for the ATNC options, which captures the heightened correlation between the different ATNC services as shown in equation (5.10):

$$c_T = 1 - \left(\frac{1}{\mu_T} \right)^2 \tag{5.10}$$

The value of $P(T_i|T)$ is calculated using equation (5.11):

$$P(T_j|T) = \frac{e^{\mu_T\ V_{T_j}}}{e^{\mu_T\ V_P} + e^{\mu_T V_S}} \qquad \forall\, j = \{P,\ S\} \tag{5.11}$$

$V_P = IV_P$ and $V_S = IV_S$ are the inclusive values of the private and the shared rides service nests respectively and are calculated using equations (5.12) and (5.13):

$$IV_P = \frac{1}{\mu_P} \ln \left[\sum_i e^{\mu_P \times V_{P_i}} \right] \tag{5.12}$$

$$IV_S = \frac{1}{\mu_S} \ln \left[\sum_i e^{\mu_S \times V_{S_i}} \right] \tag{5.13}$$

Where μ_P and μ_S are the scales of the error terms for different ATNCs, capturing the heightened correlations between different options each service type nest as shown in equations (5.14) and (5.15):

$$c_P = 1 - \left(\frac{1}{\mu_P} \right)^2 \tag{5.14}$$

$$c_S = 1 - \left(\frac{1}{\mu_S} \right)^2 \tag{5.15}$$

The probabilities of choosing a private or a shared ride with firm i are obtained using equations (5.16) and (5.17) respectively:

$$P(P_i) = P(P_i|P)P(P) \qquad \forall\, i \in I \tag{5.16}$$

$$P(S_i) = P(S_i|S)P(S) \qquad \forall\, i \in I \tag{5.17}$$

The values of $P(P_i|P)$ and $P(S_i|S)$ are calculated using equations (5.18) and (5.19) respectively and the values of $P(P)$ and $P(S)$ are found using equation (5.7):

$$P(P_i|P) = \frac{e^{\mu_P \; V_{P_i}}}{\sum_{i=1}^{n} e^{\mu_P \; V_{P_i}}} \qquad \forall\, i \in I \tag{5.18}$$

$$P(S_i|S) = \frac{e^{\mu_S \; V_{S_i}}}{\sum_{i=1}^{n} e^{\mu_S \; V_{S_i}}} \qquad \forall\, i \in I \tag{5.19}$$

Using the probabilities as estimated by the three-level nested logit model, the expected revenue for the dynamic pricing firm i from each traveller is defined in equation (5.20):

$$\mathbb{E}(R_i) = P(P_i)p_d + P(S_i)\beta p_d \qquad \forall\, i \in I_d \tag{5.20}$$

The value of β corresponds to a reduction factor, assuming shared ride services are priced at a predetermined discounted rate for the ATNCs. Where $I_d \subseteq I$ represents the dynamic pricing ATNC in the model. If M is the set of all possible values of m defined between $[1, m_{max}]$, the dynamic pricing firm i chooses the value m^* for m which maximises equation (5.20), as shown in (5.21):

$$m^* = \arg\max_m \{P(P_i)mp_s + P(S_i)m\beta p_s | m \in M\} \tag{5.21}$$

5.2.2 Baseline Scenario

The dynamic pricing method presented in Section 5.2.1 was tested in a city-wide scenario using an ABM built using the framework in chapter 3. The agents in the ABM are the AVs of each operator and travellers. AVs are assumed to exist in the system throughout the whole simulation period, while travellers appear once at the time of their travel request and exit when they are served. Each operator in the ABM has a specified fleet size and applies either a dynamic or a static pricing model.

Once a traveller appears is unassigned and sends a travel request to all the ATNCs in the ABM. The operators assign AVs based on their vehicle availability for their private and shared ride services and place their bids for each service as defined by the pricing model in Section 5.2.1. The traveller, in turn, evaluates the utility of each option using the nested logit model described in Section 5.2.1 and makes a choice. If an ATNC choice is made, the traveller waits for pick-up; otherwise, it exits the system with the choice of public transport.

For the purpose of this section, the proposed dynamic pricing model was tested on the Greater London road network as obtained from [134]. Using vehicle velocity data from [135] and testing for a weekday between 08:00 to 00:00, the average vehicle velocity was set to 17.6 km/h between 08:00 to 19:00 and to 32.2 km/h from 19:00 to 00:00. The values of t and d in equation (5.1) are estimated using the average velocity of the system at the time of request and the shortest path via the A^* algorithm.

In the absence of a generic trip database for Greater London, the input trip data was created from public transport data (Rolling OD Survey) available from [136]. The database in [136] provides trip counts from and to each London Tube Station at different times of the day and represents a typical weekday in Autumn 2017. Using the trip counts as means of Poisson Distribution for each OD pair and each period in the day, a trip database of 94816 individual public transport trips was created for a typical weekday which represents a 2% sample of the daily London Tube trip counts.

Each trip in the database was given random origin and destination coordinates within a 20 minute walking distance at 5 km/h from the origin and destination stations of the public transport trip. Consequently, t_{PT} for each traveller is the sum of the walking time to and from the origin and destination stations respectively and the travel time in the public transport system, allowing for a 2 minute interchange/waiting time which is represented by w_{PT} in equation (5.5). The values of p_{PT} for each traveller were defined using the pay-as-you-go Tube and rail fares for 2017 [137] and depend on the start and finish zones of the public transport network, as well as the time of travel .

The mean value of time $\bar{\nu}$ was chosen to be 12.85 GBP/h, which corresponds to the value of time for underground passengers for the year 2035 (assuming ATNC services by 2035) as provided in [138]. The shape factor k for the gamma distribution of the values of time was chosen to be 3.00 after calibration. To achieve a considerable mode share for ATNCs (since the original database considers public transport trips), the value of the inherent preference for ATNCs was set to follow a normal distribution $N(2, 0.5)$.

The correlations of c_P and c_S in equations (5.14) and (5.15) respectively, were assumed to be equal and set to 0.9. The equality assumption was based on the initial assumption made in Section 5.2.1 that any ATNCs in the simulation offer services of identical quality. However, the correlation between different ATNC services c_T, was set to be 0.6, so as to suggest that price has less weight on the choice between a private and a shared ride than the choice between different firms for the same product. The set M of the possible dynamic multipliers was set in the interval $[1.0, 3.0]$ in 0.1 increments.

The values of f_b, f_t and f_d were chosen so as to reflect the AV rates predicted in the literature. Specifically, the authors in [14], assume SAEV prices between $0.75-$1.00 per mile, which would generate significant revenues to the operators. Similar studies such as [139], estimate that SAEV services could be offered at approximately $0.66-$0.74 per occupied mile of travel, by performing financial analysis on the anticipated SAEV market.

In the pricing model, trips are priced on a per time and per distance basis on top of a fixed base fare, therefore, by setting $f_b = 0.5$, $f_t = 0.03$ and $f_d = 0.15$, the price per mile depending on the average velocity is as shown in Figure 5.2. These values converged to a rate of 0.41 GBP/mi for the low average velocity scenario (17.6 km/h) and 0.34 GBP/mi for the high average velocity scenario (32.2 km/h). A minimum price of 1.00 GBP is set for each trip to avoid low prices for shorter trips. Furthermore, the reduction factor β for shared rides is set to 0.75 between 08:00-19:00 and increased to 0.9 afterwards.

The model was tested with both monopolistic and competitive market structures. In the monopoly market structure, different instances of dynamic and static pricing ATNCs were considered, whereas, in the competitive market structure, a dynamic pricing ATNC was operating against a static pricing ATNC at all instances. The different fleet size and market structure scenarios tested with the model are shown in Table 5.1.

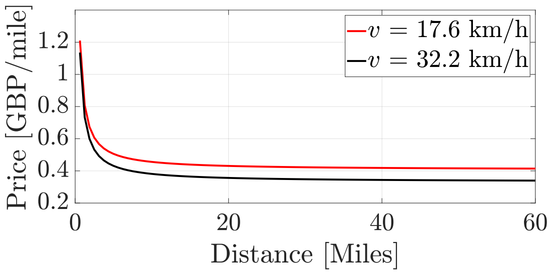


Figure 5.2: Price per Mile for Static Pricing Model.

5.2.3 Analysis

The results indicate that the effects of the proposed pricing model vary significantly depending on the market structure. Consequently, the analysis is split into two parts, focusing on monopoly and competition, respectively.

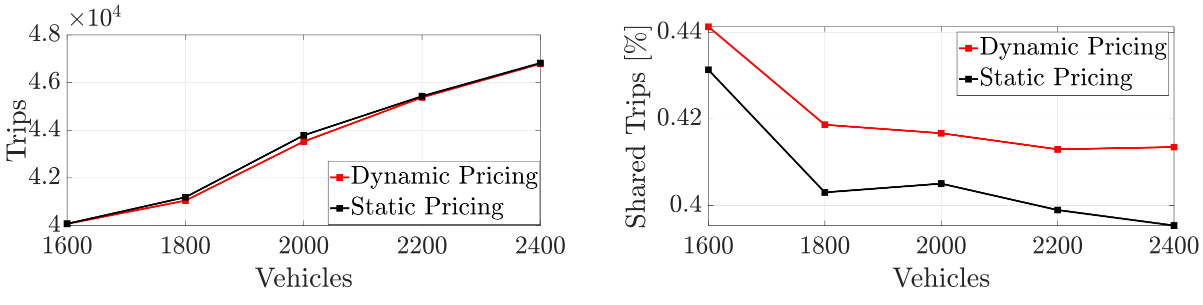
Table 5.1: Fleet Size and Market Structure Scenarios

Monopoly	Competition	
Dynamic & Static	Dynamic	Static
2400	800	1600
2200	1200	1200
2000	1300	700
1800	700	1300
1600	1000	1000
	1000	600
	600	1000
	800	800

Monopoly

It is observed, that in comparison with the static pricing model, dynamic pricing attracts a similar amount of trips in monopoly. This observation can be seen in Figure 5.3a, where although static pricing produces slightly more trips in some scenarios, there are no significant differences in the total trip numbers. This remark appears to be consistent with an increasing number of vehicles.

Vehicle sharing appears to be more attractive in monopoly with a decreasing fleet size and seems to converge to a constant value as the fleet size increases. This behaviour could be explained by the expectation that less unoccupied vehicles are available as the fleet size decreases, which results in more cases where travellers are only offered shared trip bids, thus increasing the proportion of shared trips. Furthermore, dynamic pricing appears to produce a higher percentage of shared trips compared to static pricing in monopoly as seen in Figure 5.3b, suggesting that some dynamically priced travellers chose to share but would have preferred private rides in the static pricing case.

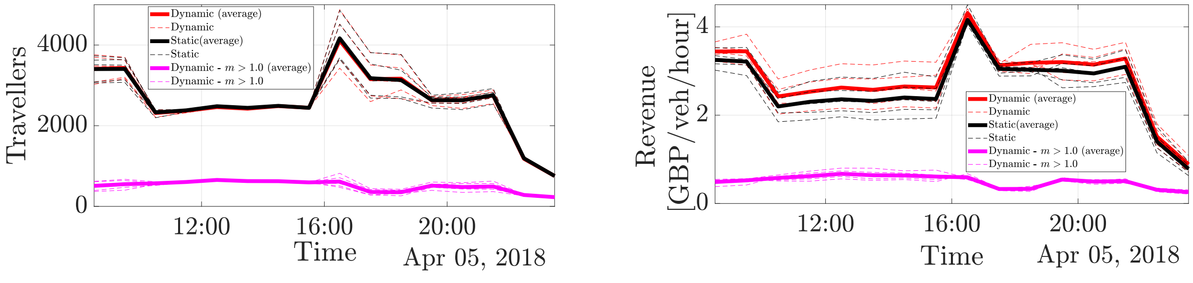


(a)

(b)

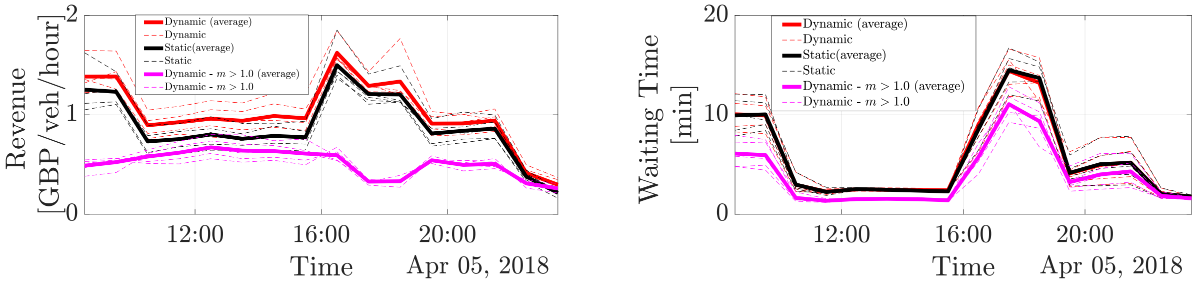
Figure 5.3: Total TNC trips by pricing model (a) and proportion of shared trips (b) in monopoly.

When investigating the behaviour of the two models in time, it is observed that most of the dynamically priced trips ($m > 1.0$) occur in non-peak times as seen in Figure 5.4a. It is also evident that dynamic pricing revenue is superior to static pricing overall in non-peak times and especially from shared trips as seen in Figures 5.4a and 5.5a. The surge of dynamically priced trips in non-peak times can be explained by the fact that, at these periods, ATNCs have an extra margin to price dynamically and still be competitive with public transport due to the low average waiting times. The observation that dynamically priced trips have a lower than average waiting time as seen in Figure 5.5b also justifies this argument.



(a) (b)

Figure 5.4: Total TNC trips (a) and TNC revenue (b) in monopoly throughout the simulation.



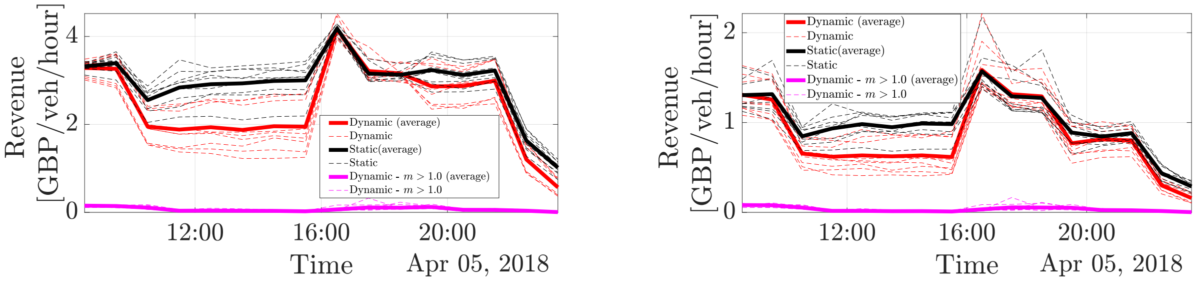
(a) (b)

Figure 5.5: TNC shared trip revenue (a) and average waiting time (b) in monopoly throughout the simulation.

Competition

The characteristics of the outputs for dynamic pricing in the competitive market are significantly different than the monopolistic scenarios. Although the total number of ATNC trips appears to be similar to the monopolistic scenarios for the same total vehicle numbers, the static pricing model seems to be vastly superior to the dynamic pricing model in non-peak times, with no significant differences when considering shared rides, as observed in Figures 5.6a and 5.6b.

Travellers strongly prefer the static alternative when average waiting times are low, suggesting that differences in waiting time at these periods are not significant to justify choosing a more expensive option for a shorter wait. On the contrary, at peak times, when average waiting times are high, dynamically priced trips are attractive if the difference in the wait is significant. These outcomes can be observed in Figure 5.7, where it can be clearly seen that the percentage difference between the average waiting time of all ATNC trips and the average waiting time of dynamically priced trips is much higher in peak hours. This difference results in competitive revenues for the dynamic pricing ATNC, when



(a)

(b)

Figure 5.6: TNC total (a) and shared (b) trip revenue in competition with static pricing throughout the simulation.

compared to the static pricing ATNC.

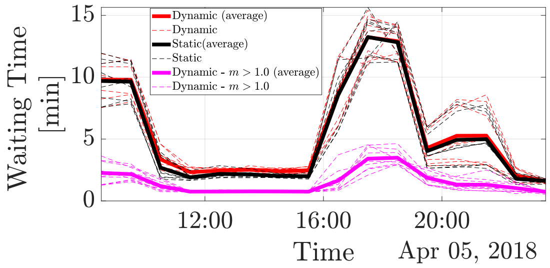


Figure 5.7: Average Waiting Time in Competition.

To further investigate this observation, the analysis scrutinised the relationship between the average waiting time of the system at all different times for all the competitive scenarios with the difference in revenue per vehicle per hour between the dynamic pricing firm and the static pricing firm. As shown in Figure 5.8, and by calculating the Pearson correlation coefficient R between the two variables, the value of $R = 0.705$, highlights a moderate to strong association between the average waiting time and the effectiveness of dynamic pricing revenues, in competition with static pricing.

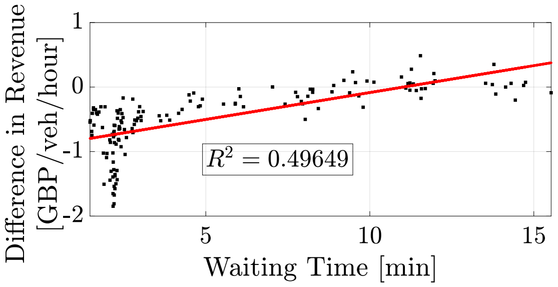


Figure 5.8: Linear Regression Analysis of Average Waiting Time and Difference of Dynamic and Static Pricing Revenues in Competition.

5.2.4 Practical Implementation Considerations

The model presented in section 5.2 can offer strategic insight on the implications of dynamic pricing in transportation mode assignment. The main result is that when used during periods of peak travel, dynamic pricing can incentivize a shift towards ride-sharing, thereby relieving the system from excess demand and potentially mitigating congestion. Furthermore, the results insinuate that operators might be able to apply such pricing strategies without compromising revenue segments.

The potential impact of the proposed dynamic pricing model presented in this section is bounded by limitations which impede its practical implementation. Firstly, nested-logit models have been extensively applied in transportation, but they constitute a simplified approach to choice modelling that cannot adequately capture the effects of dynamic pricing methods. Furthermore, trip datasets in the case study were derived from travel records that pertain to a single mode of public transportation, assuming the same distribution for the entirety of the travel demand. Nonetheless, derivation of parametric valuation distributions for travellers using data from ride-sharing companies would create travel valuation datasets which accurately resemble realistic travel behavior.

Finally, assumptions regarding the market structure in the simulated scenarios do not appropriately capture the complexity of a real transportation market. The urban traveller benefits from various transportation mode choices, other than only ride-sourcing (i.e. cycling). Additionally, due to the ease of entry in the ride-sourcing market, currently multiple TNCs offer services in large cities. Therefore, a model which emulates competitive ride-sourcing settings with multiple firms implementing dynamic pricing, will be a more realistic representation of current urban ride-sourcing markets.

5.3 A Game Theoretical Example for Dynamic Pricing Competition

As highlighted previously, the dynamic model proposed in section 5.2 can offer strategic insight to policy-makers, but one of its main drawbacks is lack of competition with multiple dynamically pricing operators. As such, this section presents an extension of the model rationale presented in section 5.2, by considering an example of two ATNCs offering services using dynamic pricing strategies. The model has game-theoretical aspects by implementing a two-player incomplete information game, and is intended for use by policy-makers.

5.3.1 Model Description

The key actors in the model are two symmetric TNC firms offering an identical product (rides) of equivalent quality. A centralised platform receives ride requests from the public. Both firms are expected to respond to these requests, with quotes for service and estimates of anticipated wait and travel times.

Travellers can then decide (using a generalised costing mechanism) whether to accept an offer or to revert to public transport.

TNCs are expected to operate with a profit maximization objective and are allowed to introduce surcharges upon the base prices for each trip. These are based upon current demand levels, and a TNC's perception on the ability of its competitors to serve a specific trip request. Using dynamic pricing, for a set I of TNCs, the final bid price is the sum of a static base price r_i per time of travel, and an extra variable tariff f_i per time of travel both set by each TNC. The price p_i for operator i is calculated using (5.22) and (5.23):

$$p_i = f_i + r_i \qquad \forall\, i \tag{5.22}$$

$$f_i \in \mathbb{R}^+ \qquad \forall\, i \tag{5.23}$$

The base price level r_i is assumed to equal the marginal cost per travel time per vehicle for each TNC i. As such, following from Bertrand competition [140], firms can only control bid values through f_i. In turn, travellers in the model evaluate the utility of each bid using a similar nested-logit[1] model structure as in section 5.2. Customer utilities for each modal options are calculated using (5.24) for TNC firms U_{T_i} and (5.25) for public transport U_P:

$$U_{T_i} = V_{T_i} + \varepsilon_{T_i} = \alpha_i - \nu(w_i + t_T) - p_i t_T + \varepsilon_{T_i} \qquad \forall i \in I \tag{5.24}$$

$$U_P = V_P + \varepsilon_P = -\nu(w_P + t_P) - p_P + \varepsilon_P \tag{5.25}$$

Where p_P and w_P are the price and waiting times for trips using public transport, w_i is the waiting time for firm i, t_P and t_T are the in-vehicle travel times via public transport and TNC respectively. Similarly to section 5.2.1, the parameter α_i is used to represent the inherent preferences for the ATNCs due to unobserved factors such as comfort, brand image and trust and is assumed to vary across the traveller population following a normal distribution with standard error σ_{α_i}

Similar to the model in section 5.2, the stochastic error terms ε_{T_i} and ε_P are randomly distributed variables following a type 1 extreme value distribution, which is the assumption underlying the nested logit model structure. The latter assumes a heightened correlation between the stochastic error terms for the TNCs (i.e. ε_{T_i}) thus allowing for a greater elasticity between the TNC alternatives. The values of p_P, w_P, w_i, t_P and t_T vary between clients and depend on network characteristics and vehicle distributions across the network at the time of the request.

For the nested logit model choice probabilities are computed as follows for the high level choice of TNC T or public transport P [133]:

[1] A shared trip service in this section is omitted to assist convenience for the proof of concept but it can be implemented in a similar way.

$$P(b) = \frac{e^{\mu\ V_b}}{e^{\mu\ V_T} + e^{\mu V_P}} \qquad \forall\ b = \{T,\ P\} \tag{5.26}$$

Where μ is the scale of the stochastic error terms, assumed to be 1 between the high-level options (TNCs versus public transport). The probability of choosing one of the TNC firms is given by:

$$P(T_i) = P(T_i|T)P(T) \qquad \forall\ i \tag{5.27}$$

$$P(T) = \frac{e^{\mu\ V_T}}{e^{\mu\ V_T} + e^{\mu V_P}} \tag{5.28}$$

$V_T = IV_T$ is the inclusive value of the TNC nest and is calculated using equation (5.29):

$$IV_T = \frac{1}{\mu_T} \ln \left[\sum_i e^{\mu_T \times V_{T_i}} \right] \tag{5.29}$$

Where μ_T is the scale of the error terms for the TNC options, which captures the heightened correlation between the different TNC choices as shown in equation (5.30):

$$corr = 1 - \left(\frac{\mu}{\mu_T} \right)^2 \tag{5.30}$$

The value of $P(T_i|T)$ is calculated using equation (5.31):

$$P(T_i|T) = \frac{e^{\mu_T\ V_{T_i}}}{\sum_{j=1}^{n} e^{\mu_T\ V_{T_j}}} \tag{5.31}$$

For each trip request, the operators evaluate the probability of winning the bid based on the estimated traveller utilities for each option. Estimates are required as operators are not privy to the real value of client-specific parameters α_i and ν, nor the real values of their competitors' w_i. To estimate the traveller utilities, each operator i uses random variables from the distributions of the parameters calibrated from observed behaviour, denoted as $\hat{\alpha}_i, \hat{\nu}_1$ and $\hat{\mu}_T$. The model assumes each operator i also uses an estimate[2] $\bar{w}_{j,i}$ for the waiting time of operator j. Hence the estimate by firm i of the utility $\bar{V}_{j,i}$ for when the customer in question chooses firm j is given by equations (5.32), (5.33) and (5.34):

$$\bar{V}_{j,i} = \hat{\alpha}_i - \hat{\nu}(\bar{w}_{j,i} + t_T) - \bar{p}_{j,i} t_T \qquad \forall\ i, j \tag{5.32}$$

$$\bar{p}_{j,i} = \bar{f}_{j,i} + r_j \qquad \forall\ i, j \tag{5.33}$$

$$\bar{f}_{j,i} \in \mathbb{R}^+ \qquad \forall\ i, j \tag{5.34}$$

[2] Estimating wait times is beyond the scope of this section.

Similarly, the estimate by firm i of the utility $\bar{V}_i$ and the estimate of the utility $\bar{V}_P$ for when the same customer chooses operator i or public transport respectively, is given by equations (5.35) and (5.36) respectively:

$$\bar{V}_i = \hat{\alpha}_i - \hat{\nu}(w_i + t_T) - p_i t_T \qquad \forall\, i \tag{5.35}$$

$$\bar{V}_P = -\hat{\nu}(\; w_P + t_P) - p_P \tag{5.36}$$

Operators choose the price of their bids to maximise their expected profit. Since the utility values used in the probability calculations are estimates, so is the expected profit. For simplicity, a scenario where only two TNCs operate is assumed. Hence, the expected profits $\mathbb{E}(\bar{Pr}_{T_i})$ and $\mathbb{E}(\bar{Pr}_{T_{j,i}})$, given the marginal cost r_i per time of travel for each operator are:

$$\mathbb{E}(\bar{Pr}_{T_i}) = \bar{P}(T_i \mid \bar{f}_{j,i}) \times ((f_i + r_i) - r_i) = \bar{P}(T_i \mid \bar{f}_{j,i}) \times f_i \qquad \forall\, i \tag{5.37}$$

$$\mathbb{E}(\bar{Pr}_{T_{j,i}}) = \bar{P}(T_{j,i} \mid f_i) \times ((\bar{f}_{j,i} + r_j) - r_j) = \bar{P}(T_{j,i} \mid f_i) \times \bar{f}_{j,i} \qquad \forall\, i, j \tag{5.38}$$

If a pair of values f_i^* and $\bar{f}_{j,i}^*$ exists for which both $\mathbb{E}(\bar{Pr}_{T_i})$ and $\mathbb{E}(\bar{Pr}_{T_{j,i}})$ are maximised, this constitutes a Nash Equilibrium. Hence, the algebraical solution for the Nash Equilibria, if they exist, could be found by solving the system of non-linear equations for f_i and $\bar{f}_{j,i}$ as defined in equations (5.39) and (5.40). Fixed costs are not considered in equations (5.37) and (5.38) since they do not influence the choice of f_i which is defined by (5.39) and (5.40):

$$\frac{\partial(\mathbb{E}(\bar{Pr}_{T_i}))}{\partial f_i} = g(f_i,\; \bar{f}_{j,i}) = 0 \qquad \forall\, i \tag{5.39}$$

$$\frac{\partial(\mathbb{E}(\bar{Pr}_{T_{j,i}}))}{\partial \bar{f}_{j,i}} = h(f_i,\; \bar{f}_{j,i}) = 0 \qquad \forall\, i, j \tag{5.40}$$

The mixed strategy Nash equilibrium in the proposed model is expected to be different than the Bertrand model [140], where the equilibrium price is the marginal cost. This is due to allowing for variation in the inherent preference and waiting times between TNCs. This variation gives the competitive advantage to the operator with the highest sum of the inherent preference and waiting time terms in equations (5.32) and (5.35) to set the equilibrium value of extra variable tariff above 0.

To find a solution, the mixed strategy profile $f^* = (f_1^*, f_2^*)$ of the game needs to be calculated. The mixed strategy profile considers the set of responses for each operator, based on the possible variations of the competitor's price. Then, for each operator i, the best response to strategy f_{-i}^* is added to the set of best responses $BR_i(f_{-i}^*)$. Overlapping values in the sets $BR_1(f_{-1}^*)$ and $BR_2(f_{-2}^*)$ constitute Nash Equilibrium points of the model.

To demonstrate a solution approach for the model, the range of f_i values is discretized into a finite set $f_i : F \longrightarrow [0, ..., f_{max}]$. The reaction for each TNC i based on the price set by TNC j is outlined in equation (5.41):

$$f_{i|f_j} = \arg\max_{f_i} \mathbb{E}(\bar{Pr}_{T_i}(f_i, f_j)) \tag{5.41}$$

Figure 5.9, outlines how the best responses of each operator and the Nash equilibrium can be identified graphically. For reference, the responses on figure 5.9 were obtained by assuming a discretized range of $f_i \in [0, 1]$ in 0.1 increments. The value of ν was sampled from a lognormal distribution with mean 17.69 GBP/hour and standard deviation of 0.02 [132]. The values of r_i were set to 0.5 for both operators and the cost of a public transport trip set to 2.50 GBP. The travel time t_T for a TNC trip was set to 20 minutes whereas for public transport, t_P was set to 40 minutes. To highlight that the model can be useful when there are sizeable differences in wait times, the w_1 and w_P were set to 5 minutes and w_2 was set to 15 minutes. Finally, the value of μ_T was set to 0.9 as in section 5.2.1.

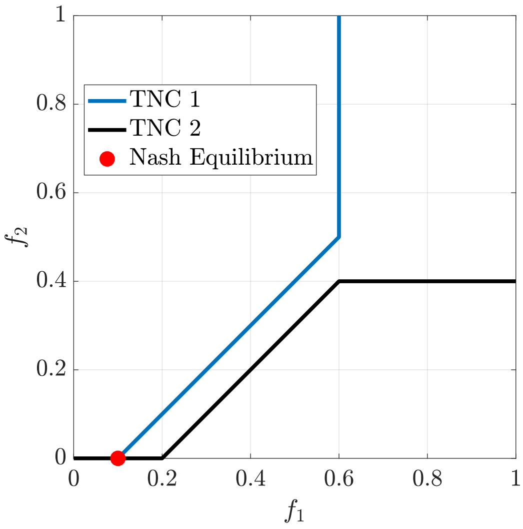


Figure 5.9: Best response graph for a model of two TNCs.

5.3.2 Practical Implementation Considerations

The example presented in this section offers a framework for extension of the model presented in section 5.2.1 in competitive ride-sourcing market structures. The model assumes incomplete information between the TNC participants, as the state of each operator (service quality and fare setting) are confidential. This uncertainty in the model, in most cases, precludes TNCs from setting dynamic fares, unless they offer a sizeable competitive advantage, such as a superior service quality. Nonetheless, the model can serve as a preliminary exercise by policy makers.

Reckoning the practical implementation considerations of both the models presented in sections 5.1 and

5.3.1, section 5.4 proposes an assignment and pricing methodology, which can be useful to operators as an operationally capable approach. The practical capability of the model is tested in section 5.5 where the chapter 5 is concluded.

5.4 Assignment and Pricing of Shared Rides using Combinatorial Double Auctions

The model in this section assumes that travellers request shared rides through a central TNC platform that operates its own vehicle fleet. Alongside origin/destination coordinates, travellers also submit their trip valuations. Vehicles have a fixed per-minute cost rate that is known in advance by the platform. The objective of the model is to maximise the trade surplus, defined as the sum of differences between traveller valuations and vehicle costs.

Assignments are performed in intervals with duration Δ - given the larger pool of possible matches; this quasi-online approach is expected to outperform a possible FIFO alternative ([17, 2]). Two assignment types are considered: the first is between riders willing to share a trip (i.e. rider-rider), and the latter pertains to vehicles that would like to offer trips (vehicle-riders). In both cases, the algorithm seeks to identify potentially combinable requests, therefore establishing shareability networks [17] that serve as inputs to the CDA model alongside rider trip valuations. Any vehicles or travellers that are not matched by the CDA are deferred to later model executions alongside any requests that might have emerged in the meantime.

5.4.1 Pre-matching

The pre-matching stage is used to filter incompatible[3] vehicle-rider and rider-rider combinations before the execution of the CDA, therefore reducing instance sizes without penalising solution quality. Quality indices δ_w and δ_d are used to reflect the maximum allowable rider wait time, and detour[4] respectively. Let R represent a set of ride requests and K a set of vehicles operated by the platform.

For each vehicle $k \in K$ pre-matching seeks to obtain a subset $N_k \subseteq R$ that the vehicle can access within a period with approximate duration δ_w. Conversely, for each ride request $r \in R$, the process seeks to identify a subset $A_r \subseteq K$ that can be picked up within δ_w.

A ride request r is placed in N_k and a vehicle k is placed in A_r according to Algorithm 6 if condition C_0 (eq. (5.42)) is met, where $T(\langle k, r \rangle)$ is the travel time from the current location of vehicle k to the origin of request r, and $T(c)$ is the execution time of a stop sequence c.

[3] Incompatible combinations produce large wait and/or detour times for riders in the combination.

[4] Detour is defined as the additional in-vehicle time of a shared trip from a private trip that a rider might experience.

$$C_0: \quad T(\langle k, r \rangle) \leq \delta_w \tag{5.42}$$

Algorithm 6 Prematching check: Vehicle-Rider

```
for $k \in K$ do
    for $r \in R$ do
        if $C_0$ then
            $N_k \leftarrow N_k \cup r$
            $A_r \leftarrow A_r \cup k$
        end if
    end for
end for
```

In the case of rider-rider matching, the algorithm obtains the subset of second requests $I_r \subseteq R \setminus r$ that can be matched with a request $r \in R$ and executed with a detour lasting δ_d or less. The algorithm also obtains a subset of requests $J_r \subseteq R \setminus r$ that can be matched with r as the second rider in the vehicle, also with a detour of δ_d or less. As such, for every request pair $i, j \in R, i \neq j$ where i and j are the first and second rider, respectively, there exists a set of origin-destination combinations $\langle o_i, o_j, d_i, d_j \rangle$ and $\langle o_i, o_j, d_j, d_i \rangle$. The following conditions apply:

$$C_1: \quad T(\langle o_i, o_j, d_i \rangle) \leq P_i + \delta_d \tag{5.43}$$

$$C_2: \quad T(\langle o_i, o_j, d_i, d_j \rangle) \leq P_j + \delta_d \tag{5.44}$$

$$C_3: \quad T(\langle o_i, o_j, d_j, d_i \rangle) \leq P_i + \delta_d \tag{5.45}$$

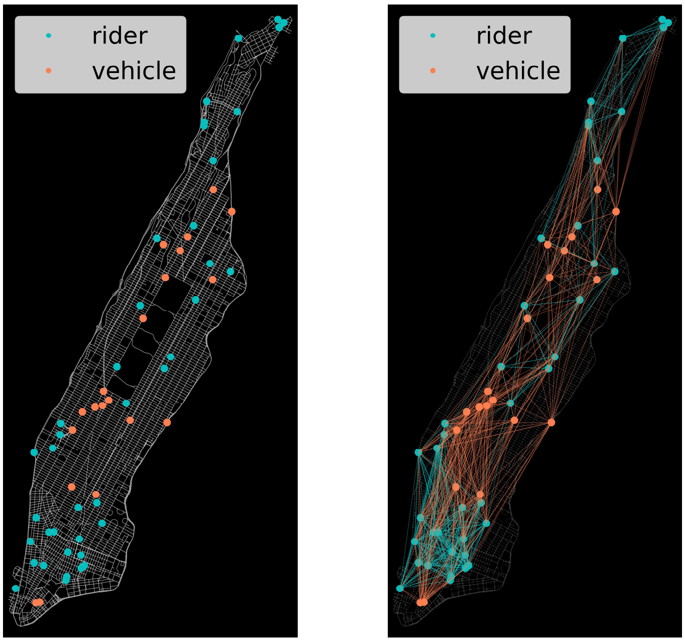
$$C_4: \quad T(\langle o_i, o_j, d_j \rangle) \leq P_j + \delta_d \tag{5.46}$$

In eq. (5.43)-(5.46), P_r represents the travel time for a private trip $r \in R$. Algorithm 7 is used to prematch rider pairs - since these are obtained alongside vehicle-rider pairs the complexity of these operations relates to the cardinality[5] of set R and is $O(|R|^2)$ [17]. The maximum possible total detour and waiting time for any rider $r \in R$ once the assignment is confirmed is $\delta_w + \delta_d$ due to pre-matching.

Algorithm 7 Prematching check: Rider-Rider

```
for $i \in R$ do
    for $j \in R \setminus i$ do
        if $(C_1 \wedge C_2) \vee (C_3 \wedge C_4)$ then
            $I_i \leftarrow I_i \cup j$
            $J_j \leftarrow J_j \cup i$
        end if
    end for
end for
```

[5] $|S|$ denotes the cardinality of any set S in this section.



(a) (b)

Figure 5.10: Problem instance before (a) and after (b) pre-matching.

The resulting adjacency subsets N_k, A_r, I_r and J_r can be visualised using a network where nodes represent vehicles or ride requests. A link from a vehicle k to rider r exists if $r \in N_k$ (and consequently $k \in A_r$), whereas a link between riders i and j exists if $j \in I_i$ (and consequently $i \in J_j$) or vice-versa. Figures 5.10a and 5.10b illustrate the auction participants' initial locations and the result of pre-matching respectively, in a randomly generated problem instance of 20 vehicles and 40 riders in Manhattan, NYC.

5.4.2 Combinatorial Double Auction Model

The auction model builds upon [2] by introducing a trading good and applying a shareability network to reduce search space. Furthermore, it takes into account the quality of shared trips and the proximity of vehicles to achieve higher time savings. As a result, riders would obtain different overall trip valuations when matched to different passengers or vehicles, while the pool of potential assignments would be further honed due to the use of a trip compatibility network. Without loss of generality, it is assumed that individual trip requests only consist of single riders. This assumption can be relaxed to extend the model to cater for larger passenger groups.

A set of riders R and a set of vehicles K are considered. Each rider $r \in R$ is identified as a 6-element tuple $\langle F_r, C_r, P_r, I_r, J_r, A_r \rangle$, where F_r is the maximum reservation price, C_r is the time valuation, P_r is an 1D array of vehicle travel times required for a private trip (pick-up to drop-off), while I_r, J_r and A_r are the adjacency subsets obtained through pre-matching (Section 5.4.1).

The 3D array $S_{i,j,n}$ represents the remaining vehicle travel time for matched riders i and j, once the final passenger is picked up, with the pick-up sequence in the order $\langle o_i, o_j \rangle$. The use of 3 dimensions for $S_{i,j,n}$ is chosen, to account for i and j having different remaining travel times once j is picked up. For example, if i is dropped off first, the remaining time for i might be $T(\langle o_j, d_i \rangle)$, whereas the remaining time for j could be $T(\langle o_j, d_i, d_j \rangle)$. At the same time, the remaining travel time for the vehicle would be $T(\langle o_j, d_i, d_j \rangle)$. Using the procedure described in Algorithm 7, the assignment with the shortest total vehicle time is obtained. Finally, the index n can take values between $[1, 2, 3]$, denoting whether $S_{i,j,n}$ refers to the first or final passenger to be picked up, or the vehicle itself, respectively.

The array $W_{i,j}$ is used to represent the vehicle travel time from initial vehicle locations or rider origins i to rider origins j, for $i \in K \cup R$, $j \in R$. The binary decision variable $x_{i,j} \in \{0, 1\}$ is used to indicate if a vehicle or request i is assigned by the action to request j, such that $i \in K \cup R$, $j \in R$. Then let:

$$T_{r,1} = \sum_{i \in I_r} \left[x_{r+|K|,i} \big(W_{r+|K|,i} + S_{r,i,1} \big) \right] \tag{5.47}$$

$$T_{r,2} = \sum_{i \in J_r} \left[x_{i+|K|,r} \big(W_{i+|K|,r} + S_{i,r,2} \big) \right] \tag{5.48}$$

denote the driving times from the pick-up location of the first passenger to drop-off location of the first and second passenger respectively. Similarly, let:

$$T_{r,3} = \sum_{i \in I_r} \left[x_{r+|K|,i} \big(W_{r+|K|,i} + S_{r,i,3} \big) \right] \tag{5.49}$$

be the driving time from the pick-up location of the first passenger to the drop-off location of the last passenger. The total service time t_r of each request r is therefore defined as follows[6]:

$$t_r = \sum_{k \in A_r} \Big[x_{k,r} W_{k,r} + x_{k,r} T_{r,1} \Big] + T_{r,2} \tag{5.50}$$

Using the waiting and travel time from (5.50) the model defines the reservation price $f(r)$ for rider r as follows:

$$f(r) = F_r - C_r t_r \tag{5.51}$$

The utility u_r of a rider with respect to request r is:

[6] The wait time from the initial vehicle location to first passenger pickup which the second passenger experiences is omitted for complexity reasons.

$$u_r = \begin{cases} f(r) - \sum_{k \in K} p_{k,r}(t_r) & \text{if } r \text{ can be served} \\ 0 & \text{otherwise} \end{cases} \tag{5.52}$$

where $p_{k,r}(t_r)$ in (5.52) is the corresponding service charge for rider r when is assigned to vehicle k, as a function of the travel time t_r. Its value is determined by the platform and is equal to zero if vehicle k is not assigned to request r. Each available vehicle $k \in K$, is described as a 3-tuple $\langle B_k, Q_k, N_k \rangle$; where B_k is its marginal operational cost, Q_k is its capacity before assignment and N_k is a subset defining riders in its vicinity (calculated as per Section 5.4.1). The travel time d_k to serve a particular set of riders for vehicle k, from starting to travel to the first rider until the delivery of the last rider is defined as follows:

$$d_k = \sum_{r \in N_k} \Big[x_{k,r} W_{k,r} + x_{k,r} T_{r,3} \Big] \tag{5.53}$$

Using eq. (5.53), the cost of serving the riders assigned to each vehicle k is defined as:

$$b(k) = B_k d_k \tag{5.54}$$

As such, the total utility for vehicle k when included in the auction process is defined by:

$$\mu_k = \begin{cases} \sum_{r \in R} p_{k,r}(t_r) - b(k), & \text{if } k \text{ serves any ride} \\ 0, & \text{otherwise.} \end{cases} \tag{5.55}$$

To identify the winners of the auction and the assignment of vehicles to riders, a WDP methodology that simultaneously considers all rider bids and vehicle costs is adopted. To achieve this, the structure of the existing formulation is modified to ensure that utilities equal to zero if rider r cannot be served or vehicle k is not assigned, for rider and vehicle utilities respectively. Since t_r and d_k both equal to zero if rider r or vehicle k are not included in any assignments, the versions of the rider utility u_r and vehicle utility μ_k are transformed as follows:

$$u_r = X_r F_r - C_r t_r - \sum_{k \in K} p_{k,r}(t_r) \tag{5.56}$$

$$\mu_k = \sum_{r \in R} p_{k,r}(t_r) - b(k) \tag{5.57}$$

where the term $X_r = \left(\sum_{k \in A_r} x_{k,r} + \sum_{i \in I_r} x_{i+|K|,r} \right)$ indicates whether rider r is in the auction either as a first or as a second client. The model aims to maximise the total utility of all the participants (vehicles and riders), with the objective function defined as follows:

$$SW = \sum_{r \in R} u_r + \sum_{k \in K} \mu_k = \sum_{r \in R} \Big(X_r F_r - C_r t_r \Big) - \sum_{k \in K} b(k) \tag{5.58}$$

where SW indicates the value of social welfare. Observe that the service charges cancel out in the summation of the participants' utilities. The optimisation problem is then formulated with the following set of constraints:

Model 1 (Winner Determination Problem for Ride Sharing):

$$\begin{aligned}
\text{maximize} \quad & SW && \text{(5.59a)}\\
\text{subject to} \quad & x_{k,r} + \sum_{i \in I_r} x_{r+|K|,i} \leq Q_k, \quad \forall k \in K, \forall r \in N_k, && \text{(5.59b)}\\
& \sum_{k \in A_r} x_{k,r} + \sum_{i \in J_r} x_{i+|K|,r} \leq 1, \quad \forall r \in R, && \text{(5.59c)}\\
& \sum_{k \in A_r} x_{k,r} - 1 \leq M\Big(1 - \sum_{i \in I_r} x_{r+|K|,i}\Big), \forall r \in R, && \text{(5.59d)}\\
& 1 - \sum_{k \in A_r} x_{k,r} \leq M\Big(1 - \sum_{i \in I_r} x_{r+|K|,i}\Big), \forall r \in R, && \text{(5.59e)}\\
& \sum_{r \in R} x_{i,r} \leq 1, \quad \forall i \in K \cup R, && \text{(5.59f)}\\
& \sum_{i \in K \cup R} x_{i,r} \leq 1, \quad \forall r \in R, && \text{(5.59g)}\\
& \sum_{k \in A_r} x_{k,r} - \sum_{i \in I_r} x_{r+|K|,i} = 0, \quad \forall r \in R, && \text{(5.59h)}\\
& x_{i,j} \in \{0,1\}, \quad \forall i,j \in K \cup R && \text{(5.59i)}
\end{aligned}$$

Eq. (5.59b) ensures that the number of assigned riders to each vehicle k is at most equal to the vehicle capacity Q_k if assigned with a rider r. (5.59c) guarantees that if rider r is assigned, it is either the first rider or the second passenger to board. Eqs. (5.59d) and (5.59e) utilize the Big M method [141] to ensure that if any two riders are matched, the first rider r in the matching has to be picked up by a vehicle k. M is defined as a sufficiently large positive number.

Eq. (5.59f) ensures that each vehicle or rider is assigned as a starting point towards a rider at most once. Eq. (5.59g) ensures that each rider is assigned as a destination from a vehicle location or a rider no more than once. Finally, eq. (5.59h) ensures that if a vehicle is connected to a rider, there would be an additional rider in the trip.

Note that eqs. (5.50) and (5.53) feeding into the objective function, include non-linear terms. Therefore, variables $y_{k,r} \in \mathbb{R}^+$ and $z_{k,r} \in \mathbb{R}^+$ are introduced, to replace the non-linear terms in equations (5.50) and (5.53) respectively as shown in equations (5.60) and (5.61).

$$t_r = \sum_{k \in A_r} \Big(x_{k,r} W_{k,r} + y_{k,r} \Big) + T_{r,2} \tag{5.60}$$

$$d_k = \sum_{r \in N_k} \big(x_{k,r} W_{k,r} + z_{k,r} \big) \tag{5.61}$$

Consequently the objective function in equation (5.58) transforms into the following:

$$\begin{aligned} SW_L &= \sum_{r \in R} u_r + \sum_{k \in K} \mu_k \\ &= \sum_{r \in R} \Big(X_r F_r - C_r t_r \Big) - \sum_{k \in K} B_k d_k \\ &= \sum_{r \in R} \Bigg[X_r F_r - C_r \Bigg[\sum_{k \in A_r} \Big(x_{k,r} W_{k,r} + y_{k,r} \Big) + T_{r,2} \Bigg] \Bigg] \\ &- \sum_{k \in K} B_k \Bigg[\sum_{r \in N_k} \Big(x_{k,r} W_{k,r} + z_{k,r} \Big) \Bigg] \end{aligned} \tag{5.62}$$

where SW_L denotes the value of the objective after linearization. To ensure that the variable $y_{k,r}$ equals its desired value, the following linearization constraints are introduced:

$$y_{k,r} \leq M x_{k,r} \tag{5.63}$$

$$y_{k,r} \leq T_{r,1} \tag{5.64}$$

$$y_{k,r} \geq T_{r,1} - M(1 - x_{k,r}) \tag{5.65}$$

$$y_{k,r} \in \mathbb{R}^+ \tag{5.66}$$

for every $r \in R$ and every $k \in A_r$. In a similar fashion, the following linearization constraints for variable $z_{k,r}$ are introduced:

$$z_{k,r} \leq M x_{k,r} \tag{5.67}$$

$$z_{k,r} \leq T_{r,3} \tag{5.68}$$

$$z_{k,r} \geq T_{r,3} - M(1 - x_{k,r}) \tag{5.69}$$

$$z_{k,r} \in \mathbb{R}^+ \tag{5.70}$$

for every $k \in K$ and every $r \in N_k$.

By incorporating the additional variables and constraints in equations (5.60)-(5.70), the optimisation methodology for *Model 1* transforms to the following MILP:

Model 2 (Transformed WDP for Ride Sharing)

$$\begin{aligned} \text{maximize} \quad & SW_L \\ \text{subject to} \quad & \text{(5.59b) - (5.59i)}, \\ & \text{(5.63) - (5.70)} \end{aligned}$$

5.4.3 Reduction to Maximum Weighted Independent Set

To assess the complexity of *Model 2*, a reduction to the Maximum Weighted Independent Set (MWIS) problem is presented. It is assumed that in the largest instance, all vehicles can be matched to all requests, and all requests are sharing-compatible. In that scenario, with $\mathcal{K}$ and $\mathcal{R}$ being the sets of vehicles and requests, respectively, let C denote the set of all possible combinations, where $|\mathcal{C}| = |\mathcal{K}||\mathcal{R}|^2 - |\mathcal{K}||\mathcal{R}|$.

Assuming that all vehicles will be assigned, the set of all path-vehicle allocations is $\binom{|\mathcal{K}||\mathcal{R}|^2 - |\mathcal{K}||\mathcal{R}|}{|\mathcal{K}|}$. To prove the APX-hardness of *Model 2*, this section uses an approximation-preserving reduction from MWIS.

Theorem 3. *Model 2 is NP-Hard*

Proof. An instance of MWIS, a known APX-hard[7] problem [142], is reduced to an instance of *Model 2*. Given a weighted graph $G = (V, E, w)$, the MWIS objective is to find a set of pairwise disjoint nodes $S \subseteq V$ with maximum total weight.

Let the tuple (k, i, j) denote the ride-sharing trip of *Model 2* with vehicle k in which the first passenger is i and the second is j, $\forall k \in \mathcal{K}, i, j \in \mathcal{R}$ and $i \neq j$. Also let $u_i(k, i, j)$ and $u_j(k, i, j)$ denote the utilities of riders i and j respectively, for the trip (k, i, j) and $u_k(k, i, j)$ denote the utility of the vehicle.

Consider now the following representation; let $G = (V, E, w)$ be a graph where each vertex represents a combination $c = (k, i, j)$. An edge exists between vertices c_n and c_m if and only if the trip combinations c_n and c_m have a common element, i.e. a common vehicle or rider. Let:

$$w_c = u_k(k, i, j) + u_i(k, i, j) + u_j(k, i, j) \tag{5.71}$$

[7] APX is the complexity class of optimization problems that cannot be approximated within some constant factor unless $P \neq NP$

denote the weight of vertex $c = (k, i, j)$. It is noted that changing the order of two riders in a combination can result in a different weight for the corresponding vertex. That is because the detour or the wait time after the reordering can exceed either of the thresholds δ_d, δ_w set during pre-matching, thus resulting in a different value of rider utilities.

The section now proves the correctness of the above transformation. Let $OPT(I')$ denote an optimal solution to a *Model 2* instance I'. For any two trip combinations c, c' that either have a common rider or vehicle, at most one of them will be in $OPT(I')$ and the vertices representing these trips will be connected by an edge in graph G. As a result $OPT(I')$ is represented by a set of independent nodes in G and since the solution is optimal with cost $\sum_{r\in R} u_r + \sum_{k\in K} \mu_k = \sum_{c\in V} w_c$ by equation (5.71) this corresponds to an independent set of maximum weight in G.

Conversely, suppose $OPT(I)$ is an optimal solution on an instance I of MWIS in G. Since $OPT(I)$ is independent, no pair of nodes will be connected, so no pair of trips from WDP will have a common element. Again according to eq. (5.71), the total weight of the selected trips is maximised. □

It is noted that the above reduction preserves the approximation [142]. Let f be the (polynomial time) transformation from an instance I' of *Model 2* to an instance I of MWIS as described above i.e. $I = f(I')$ and let g be the (polynomial time) algorithm that produces a solution to I given a solution to I'. Let also $\alpha = 1$ and $\beta = 1$. Using transformation f, the optima of I and I' satisfy the following inequality $OPT(I') \leq \alpha OPT(I)$. Furthermore, having a solution with weight w' for any instance I', one construct a solution for I with weight w such that $|w - OPT(I)| \leq \beta|w' - OPT(I')|$ using algorithm g.

Corollary 2. *Model 2 is APX-Hard.*

Many greedy approximation algorithms have been previously proposed, with their approximation ratio expressed as a polynomial in terms of the average or maximum node degree in the graph [143]. It is noted that in the fully connected scenario, the average/maximum degree of node c is $\Delta_c = |R|(|R| - 1) - 1 + (|K| - 1)(4|R| - 6)$. To demonstrate this, if a combination (k, i, j) is considered, there exist additional $|R|(|R| - 1) - 1$ trip combinations with vehicle k. For every other vehicle from the remaining $|K| - 1$, there exist $2(|R| - 1)$ trip combinations including rider i and an additional $2(|R| - 2)$ including rider j, which are not already accounted. Thus, simplifying $(|K| - 1)(2(|R| - 1) + 2(|R| - 2))$ results to $(|K| - 1)(4|R| - 6)$.

5.4.4 Local Search Algorithm Using Greedy Search Initialisers

It was established earlier that solving the MWIS problem for a fully connected CDA scenario, would involve finding a MWIS in graphs with $|\mathcal{C}| = |\mathcal{K}||\mathcal{R}|^2 - |\mathcal{K}||\mathcal{R}|$ nodes with an average/maximum node degree of $\Delta_c = |R|(|R| - 1) - 1 + (|K| - 1)(4|R| - 6)$. Considering a small localised example with 10 vehicles and 20 potential riders, that would generate a network with 3800 nodes with an average/maximum node degree of 1045.

An exact solution would, therefore, be impractical, as existing solution algorithms are slow even for a few hundreds of vertices [144]. A local search algorithm is proposed, based on simulated annealing (SA), a technique that has been shown to perform very well for the maximum clique problem (a similar premise, as it is the opposite of an independent set)[145].

Simulated Annealing (SA) was initially proposed as a probabilistic method to solve difficult optimisation problems [146]. It aims to bring a system from an arbitrary initial state to an eventual state of minimum energy. Most SAs use an energy measure that is inversely proportional to the quality of the solution and is minimised using an iterative process. Starting from a seed solution, SA iterations generate several neighbouring solutions, which are accepted in accordance with a stochastic process. The process continues until the "temperature" of the problem reaches a user-defined minimum. A high-level structure of the SA algorithm for the MWIS problem is presented in Algorithm 8.

Algorithm 8 SA for the Independent Set Problem

```
Generate initial solution $S_0$ for graph $G$
Set initial and minimum temperatures $T_0$, $T_{min}$
$S_{old} = S_0$
$E_{old} = energy(S_{old}, G)$
$S_{best} = S_{old}$
$E_{best} = E_{old}$
$T \leftarrow T_0$
while $T > T_{min}$ do
    $S_{new} = neighbour(S_{old}, G)$
    $E_{new} = energy(S_{new}, G)$
    if $E_{new} < E_{best}$ then
        $S_{best} = S_{new}$
        $E_{best} = E_{new}$
    end if
    $S_{old}, E_{old} = select(S_{old}, S_{new}, E_{old}, E_{new}, T)$
    $T = \alpha T$,    (where $\alpha$ is a constant and $\alpha < 1$)
end while
Output: $S_{best}$, $E_{best}$
```

Algorithm 8 utilizes a graph G, constructed to identify all possible vehicle-rider-rider combinations by representing them as a set of nodes. Each node in the set is a 3-tuple, $\langle c, w_c, N_c \rangle$. c refers to the combination of vehicle-rider-rider in the form of $\langle k, i, j \rangle$, w_c refers to the weight of the node as defined in Section 5.4.3 and N_c is a list of neighbouring nodes. It can be easily shown that the degree of each vertex is equal to $|N_c|$.

To construct the graph the algorithm sets $N_c = \emptyset$ and iterate through the network nodes to populate N_c for each vertex. As with Algorithm 9, this process requires $|K||R|^2$ iterations (fully connected scenario) to create the set of vertices V. Populating N_c for each vertex (and creating the edge set E), requires $|V|^2$ iterations (Algorithm 10). Since $|V|^2 = (|K||R|^2)^2$, the complexity of the worst case scenario for network generation is $O(|K|^2|R|^4)$. This process, however, can be easily parallelised.

A set of greedy heuristics with known lower bound performance [143] is used to obtain an initial

Algorithm 9 Vertex Generation Process

```
$V \leftarrow \emptyset$
for $k \in K$ do
    for $i \in N_k$ do
        for $j \in I_i$ do
            $w_c = u_i(k,i,j) + u_i(k,i,j) + u_k(k,i,j)$
            if $w_c \geq 0$ then
                $c = \langle k,i,j \rangle$
                $N_c = \emptyset$
                $V \leftarrow V \cup \langle c, w_c, N_c \rangle$
            end if
        end for
    end for
end for
Output: $V$
```

Algorithm 10 Edge Generation Process

```
Non-empty set $V$
$E \leftarrow \emptyset$
for $i \in V$ do
    for $j \in V \setminus i$ do
        if $c_i \cap c_j \neq \emptyset$ then
            $N_{c_i} \leftarrow N_{c_i} \cup j$
            $N_{c_j} \leftarrow N_{c_j} \cup i$
            $E \leftarrow \langle i,j \rangle$
        end if
    end for
end for
Output: $G = (V, E)$
```

solution S_0, consisting of an ordered set of vertices in V. These operate by sorting vertices in a descending order with respect to w_c, $1/|N_c|$, $w_c/|N_c|$ and $w_c/\sum_{i \in N_c} w_i$, respectively. The best solution among these four is identified through inspection.

To calculate the energy of a solution (Algorithm 11), the process iterates through the ordered vertex sequence S. At each step, the algorithm adds the next vertex in S to the independent set I and removes its neighbours from S. Iterations continue until S is empty. The energy of the solution is, therefore, equal to the negative sum of all values w_c, for each vertex within I.

When it comes to the generation of neighbouring solutions, the algorithm randomly selects two vertices in the independent set I of the old solution S_{old} and switches their positions in S_{old} to produce sequence S_{new}. This approach increases the chance that sequence S_{new} will produce a different independent set and energy than S_{old}. Finally, the stochastic selection method is formed on defining an acceptance probability for every new solution, which is calculated using E_{old}, E_{new} and temperature T as shown in Algorithm 12. Better solutions are always accepted, whereas worse solutions have less chance of being accepted as the iterations progress (i.e. as temperature T decreases).

Algorithm 11 Energy Calculation

```
Non-empty ordered sequence $S$
Graph $G = (V, E)$
$I \leftarrow \emptyset$
while $S \neq \emptyset$ do
    $i = S(1)$
    $I \leftarrow I \cup i$
    $S \leftarrow S \setminus (S \cap (N_{c_i} \cup i))$,    (obtain $N_{c_i}$ from $G$)
end while
$E = -\sum_{i \in I} w_{c_i}$,    (obtain $w_{c_i}$ from $G$)
Output: $E$
```

Algorithm 12 Selection Process

```
Inputs: $S_{old}$, $S_{new}$, $E_{old}$, $E_{new}$, $T$
$p = X$,    (where $X \sim U(0,1)$)
if $E_{new} < E_{old}$ then
    $p_a = 1$
else
    $p_a = e^{(E_{old} - E_{new})/T}$
end if
if $p_a > p$ then
    $S_{old} = S_{new}$
    $E_{old} = E_{new}$
end if
Outputs: $S_{old}$, $E_{old}$
```

5.4.5 Trip Price Determination

Optimal solutions of the WDP in CDAs produce efficient outcomes which are individually rational. That is, assuming participants in the auction are truthful about their valuations. There is, however, no guarantee that auction participants (bidders) will state their true valuations. [92] explains this problem with an example of three bidders. This section will extend this example to the proposed CDA, to illustrate how untruthful bids can arise. Consider a CDA scenario involving three riders (bidders) and one vehicle. Assume that from the six possible allocation combinations, the following three yield a positive value for total trade surplus:

$$f_1(\langle 1,2 \rangle) = 10, \quad f_2(\langle 1,2 \rangle) = 8, \quad b_1(\langle 1,2 \rangle) = 10 \tag{5.72}$$

$$f_1(\langle 2,1 \rangle) = 7, \quad f_2(\langle 2,1 \rangle) = 9, \quad b_1(\langle 2,1 \rangle) = 11 \tag{5.73}$$

$$f_1(\langle 1,3 \rangle) = 5, \quad f_3(\langle 1,3 \rangle) = 10, \quad b_1(\langle 1,3 \rangle) = 12 \tag{5.74}$$

In eqs. (5.72)-(5.74), $f_r(\langle S \rangle)$ and $b_k(\langle S \rangle)$ represent total valuation and cost for a rider r and a vehicle k, respectively, for a trip with a pickup sequence S. Using *Model 2*, the platform allocates the trip with the only vehicle servicing riders 1 and 2 in the sequence $\langle 1, 2 \rangle$ as it is the combination producing the highest trade surplus. Note that riders 1 and 2, assuming everyone bids truthfully, can report a lower value per time and still win the auction with the same combination.

The inclusion of additional riders will give rise to more complex bidding strategies. In the case that riders 1 and 2 reduce their bids excessively, they might lose in the auction. This characteristic CDA property is known as the threshold problem [147] and refers to the implication of valuation misreporting thresholds for individual participants, which can motivate bidders to employ perverse bidding strategies [148].

Pricing in VCG auctions, where bidders pay the difference of welfare in their absence with the welfare of others when they are included in the auction, is incentive-compatible [92]. Furthermore, incentive-compatible payments have been derived through the solution of dual relaxed linear problems of the WDP [149]. Previous studies [150, 151, 2], used relaxed dual WDP problems to identify allocation and pricing in double auctions, with Lagrangean multipliers to be considered as prices. It has been shown that optimal dual variables in LP coincide with VCG payments [152].

However, the use of near-optimal CDA solutions does not preserve incentive compatibility [153]. Negligible variations from the optimal objective can have significant consequences on the payments to be made by bidders [154]. As such, an approximate WDP solution would inhibit the use of VCG or dual LP relaxations that would guarantee incentive-compatibility. The NP-hardness of the proposed CDA prohibits the identification of exact WDP solutions in practical implementations, thereby the uses of VCG or dual LP relaxations are omitted for price determination.

Instead, this section proposed a model which resembles a Generalized First Price (GFP) auction for trip pricing. A GFP mechanism is an untruthful auction mechanism, where participants bid for the allocation of a limited amount of slots. Participants pay their bid values in case they are assigned to a slot. Previous research outlined deficiencies in the GFP mechanism by strategically employed shill bidding which destabilizes the auction [155]. Subsequent work in [156] attributes these GFP deficiencies to the auction interface and argues that GFP auctions can be robust by allowing expressiveness of the participants using multidimensional bids.

In the conventional GFP, an individual i submits a single bid f_i, which is multiplied by $s_1 \geq s_2 \geq ... \geq s_k$, k being the last available slot. The expressive version of GFP dictates that an individual i submits a different bid f_{ik} for each slot k which is multiplied by $s_1 \geq s_2 \geq ... \geq s_k$ accordingly. Our proposed CDA resembles an expressive GFP, as travellers bid for a limited number of vehicle seats (slots) and by submitting a valuation per time C_r, they might obtain a different valuation $f(r)$ for each potential vehicle-rider-rider assignment.

To limit the effect of untruthful bids on the auction outcome, the model proposes that each rider only submits the valuation per time C_r. The platform in turn identifies and privately informs the rider of its maximum reservation price F_r, so that if matched, the payment will comprise of a discounted static price for the time of the trip attributing to P_r and an additional variable rate attributing to $C_r\bar{\delta}_r$, where $\bar{\delta}_r$ is the wait and detour time saved by choosing the platform, instead of the rest of the market.

Consequently, the maximum reservation price F_r is derived by the platform using the following generalised cost equation:

$$F_r = p_b + P_r p_t + C_r(P_r + \delta_w + \delta_d) \tag{5.75}$$

where p_b is the flat fee and p_t is the discounted price per minute for a shared trip, lasting P_r minutes if private, as specified by the platform. δ_w and δ_d refer to the guaranteed maximum wait and detour times respectively, which are used in pre-matching by the platform.

By introducing this format, it is straight-forward to deduce by observing equations (5.51), (5.54) and (5.58) that in the event where bidders submit per time valuations C_r which are very close to zero, the proposed CDA converts to an optimal 3D assignment problem where the sum of detours is minimised, if the following inequality holds for any vehicle k and riders i, j prior to the auction:

$$b(k) \leq f(i) + f(j) \tag{5.76}$$

By introducing this condition with equation (5.76), the model ensures that the auction always returns an assignment if a pre-matching instance exists as any rider payments in the GFP instance will always cover the vehicle costs. It is thereby necessary to choose the appropriate value for the flat fee p_b, such that equation (5.76) holds. In doing so, it is assumed that the total rider payment per vehicle equals its cost. It is also assumed that vehicle costs B_k are uniform across the fleet (i.e. $B_k = B \quad \forall k \in K$) and extend the functions as per equations (5.51) and (5.54):

$$Bd_k = F_i - C_i t_i + F_j - C_j t_j \tag{5.77}$$

Using equation (5.75), and by substituting $\delta_w + \delta_d$ with δ, the following equation holds:

$$Bd_k = 2p_b + p_t(P_i + P_j) + C_i(P_i + \delta - t_i) + C_j(P_j + \delta - t_j) \tag{5.78}$$

In the minimal total bid scenario, both C_i and C_j in equation (5.78) would be zero. Also, it is noted that d_k is equal to $max(t_i, t_j)$. By setting the total wait and detour time experienced by each rider r as δ_r, one can replace t_r by $P_r + \delta_r$. Therefore, with C_i and C_j set to zero this results to the following equation:

$$Bmax(P_i + \delta_i, P_j + \delta_j) = 2p_b + p_t(P_i + P_j) \tag{5.79}$$

The maximum vehicle cost in equation (5.79) for any values of P_i and P_j, occurs if $max(P_i + \delta_i, P_j + \delta_j) = max(P_i, P_j) + \delta$, for δ as introduced above, being the maximum total wait and detour time guarantee by the platform for an individual rider. Assuming the value of p_b is zero and that p_t is set by the platform such that $p_t \geq B$, if $min(P_i, P_j) \geq \delta$ the condition in (5.76) always holds. If however $min(P_i, P_j) < \delta$, a flat fee p_b is required to ensure the condition in (5.76). As such, assuming both $P_i, P_j \rightarrow 0$, and $max(\delta_i, \delta_j) = \delta$, using equation (5.79), the flat fee for the proposed GFP interface should be as follows:

$$p_b = \frac{B\delta}{2} = \frac{B(\delta_w + \delta_d)}{2} \tag{5.80}$$

5.5 Discussion

The methodology presented in section 5.4 was implemented using Python and tested on a workstation with an Intel i7-4790 CPU (3.6GHz) and 8GB RAM. Exact solutions were obtained using the Branch and Cut algorithm provided by IBM ILOG Cplex Optimization Studio 12.7.1.

To test the algorithm, a case study network set in Manhattan, NYC was created. The underline road network and travel times were obtained using the OSMnx library [118]. To account for congestion, a 20% penalty was applied to the free-flow speeds in residential and motorway link segments, and 40% elsewhere. Rider origin-destination pairs, as well as vehicle locations, were sampled uniformly in space to create CDA instances. Only trips with travel time that is greater than 5 minutes were considered, while δ_w and δ_d were both set to 10 and 15 minutes respectively.

For this model, UK-based estimates of working time valuations were used [132] for the derivation of rider valuations. Vehicles were assumed to have a capacity of two customers, with their operating costs B_k uniformly set to 12.96 GBP/hour. Conversely, customer time valuations C_r were sampled from a log-normal distribution with a mean of 17.69 GBP/hour and $\sigma = 0.02$. The discounted price per minute p_t was set to 0.75 GBP/min.

Table 5.2 provides a performance comparison of the SA and the Branch and Cut (BC) algorithms for a range of instances. As can be seen in the table and in figure 5.11, the runtime for the BC approach grows exponentially as more vehicles and riders are considered in the instance, thereby increasing the node count of the MWIS instance, whereas the runtime for the SA remains relatively short.

The APX-complete nature of the problem is also signified in the solution comparison between BC and SA as observed in figure 5.12a, as the percentage error gradually increases with a larger instance size. However, as shown in figure 5.12a, the approximation error is relatively low for instances of such size.

Table 5.2: Performance comparison of SA and BC.

Total vehicles	Total riders	BC solution	BC runtime [sec]	SA solution	SA runtime [sec]	Error [%]
4	8	48.199	0	48.199	0.002	0.00
8	8	77.714	0.13	77.714	0.008	0.00
5	10	120.523	0.02	120.523	0.004	0.00
10	10	127.936	0.13	127.936	0.027	0.00
5	12	108.327	0.03	108.327	0.004	0.00
6	12	136.996	0.05	136.996	0.007	0.00
12	12	145.248	0.11	144.866	0.026	0.26
6	14	122.487	0.05	122.487	0.021	0.00
7	14	131.537	0.28	128.81	0.029	2.07
7	15	208.895	0.5	208.895	0.034	0.00
8	16	171.665	0.27	170.613	0.053	0.61
7	17	160.367	0.06	160.253	0.024	0.07
9	18	236.672	1.66	235.918	0.076	0.32
8	20	204.96	0.28	204.93	0.061	0.01
10	20	236.125	1.45	234.785	0.149	0.57
11	22	295.669	3.45	291.632	0.209	1.37
12	24	331.237	29.95	321.797	0.598	2.85
10	25	326.057	39.86	321.967	0.64	1.25
14	28	377.018	103.5	373.223	2.702	1.01
15	30	397.518	130.92	387.628	2.374	2.49
15	20	285.793	18.14	283.101	0.407	0.94
20	20	270.432	58.24	267.002	0.746	1.27
12	25	324.687	26.64	321.524	1.239	0.97
15	25	323.191	42.3	316.062	1.399	2.21
16	25	341.845	82.19	333.836	1.496	2.34
17	25	330.884	124.78	321.94	1.573	2.70
14	30	419.626	49.59	415.906	2.193	0.89
13	26	329.992	63.22	324.955	2.587	1.53
16	32	431.772	1003.55	408.41	3.423	5.41
17	34	469.94	4126.59	445.309	4.456	5.24

A visual comparison of the results obtained by the BC and SA algorithms is provided in figures 5.13a and 5.13b, respectively, for an instance involving 10 vehicles, 20 customers and two edges per match outlines the similarities between solutions obtained using the two approaches.

To strengthen the argument for the inclusion detour calculations on CDAs for ride-sharing, a comparison analysis was conducted between exact solutions of *Model 2* and the algorithm in the state-of-the-art which mostly resembles the problem statement, namely the CDA model in [2]. Instances were created from 10 to 22 requests, with the assumption of one seat per request. For each instance, it was assumed there are just enough vehicles to cover the demand (i.e. half the number of requests). To run the CDA in [2], the distance-based methodology was converted to time-based to match *Model 2* and omitted private rides. A vehicle capacity of two rides was used for all vehicles in both models.

As observed in Figure 5.14, since the CDA in [2] omits detours and wait times in their calculation, the

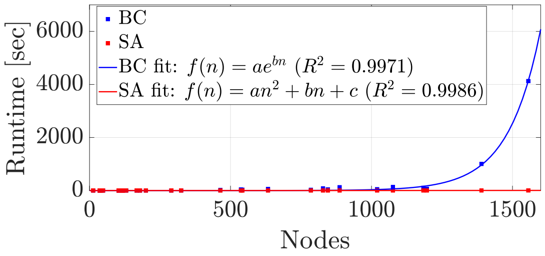


Figure 5.11: Run-time for BC and SA methods.

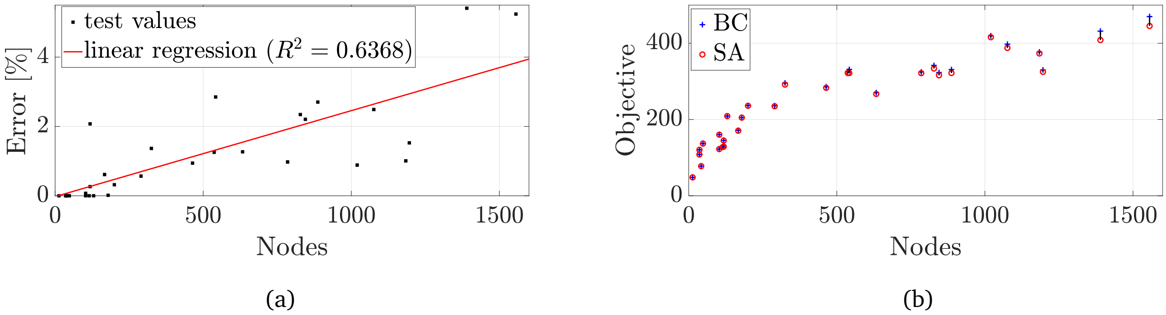


Figure 5.12: Percentage error of approximation (a) and solution values for BC and SA methods (b) against node count.

resulting assignment creates much higher detour and wait times on average for each instance. Consequently, ignoring the effect of detours and wait times in ride-sharing CDAs can produce assignments which might not be acceptable by the users of the service. Taking the time dimension into account th convenience of the service can be massively improved as observed. However, this is achieved with an increase in computational complexity, as discussed in 5.4.3. Nonetheless, a reduction of the solution space can be achieved via the pre-matching stage, as shown in 5.4.1.

5.5.1 Trade Surplus Implications

A large number of problem instances were considered, with fleet sizes ranging between 3 and 60 vehicles, and a customer base of 10 to 60 riders. From the range of greedy heuristics that were considered for SA initialisation described in section 5.4.4, weight-based approaches were found to yield the best results (Figure 5.15). Figures 5.16a and 5.16b illustrate the relationship between problem sizes and algorithm run times, which is found to be in polynomial time.

The trade surplus index (TSI) is defined as the ratio of the objective value and the number of assigned vehicles in each instance. Figures 5.17a and 5.17b illustrate its relationship with the fleet coverage index (FCI), defined as the ratio of vehicles available against the number of vehicles required to serve all requests. An interesting feature of this approach (as shown in Figures 5.17a and 5.17b) is that the TSI is inversely proportional to the FCI for values of the latter between 0 and 1, and remains constant

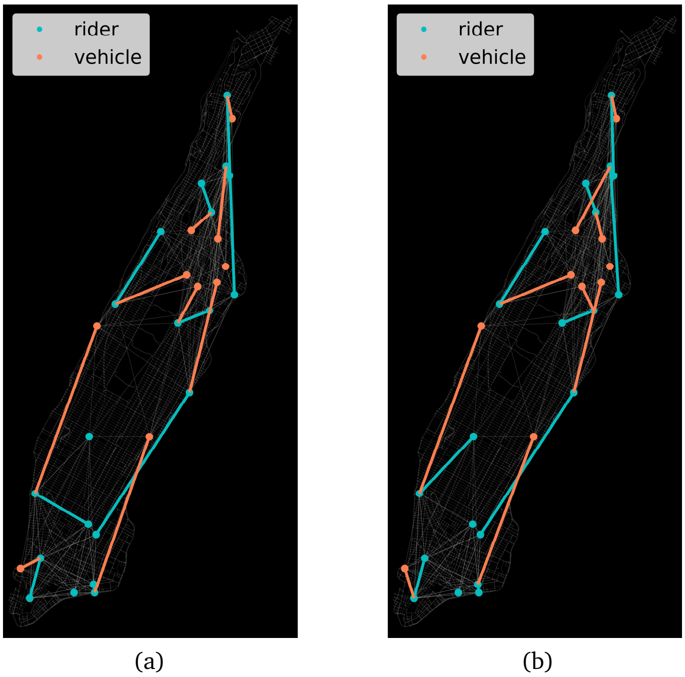


Figure 5.13: Visualisation of (a) BC and (b) SA solutions

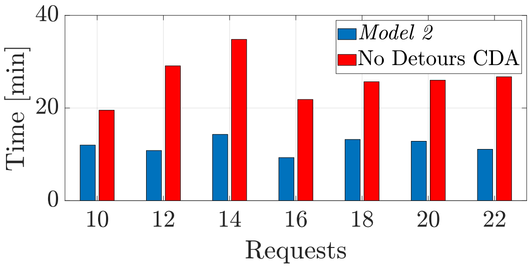


Figure 5.14: Average total wait and detour time per request for *Model 2* and CDA in [2].

beyond that point.

This pattern can be explained by considering a scenario with 1 vehicle and 10 riders. In this case, the node with the highest weight will be the solution in the MWIS problem. The addition of a new vehicle (with the same cost), assuming that it is included in the MWIS solution, will lead to a reduction in the average node weight. This trend will persist with further increases in the size of the fleet, as riders with lower valuations are accommodated and gradually reduce the overall TSI. As such, once $FCI > 1$ the TSI will on average remain constant, consistent with the notion of market equilibrium while supply increases beyond current demand levels.

5.5.2 Practical Implementation

To investigate the practical implementation of the proposed methodology, ride-sharing data provided by the Taxi and Limousine Commission (TLC) of NYC were analysed. Specifically, the high volume

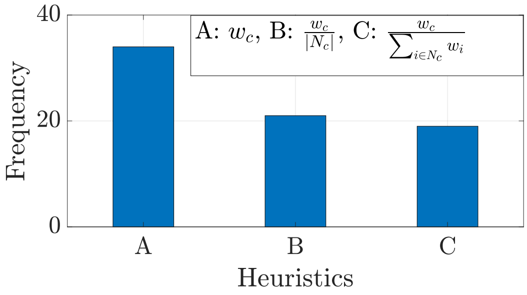
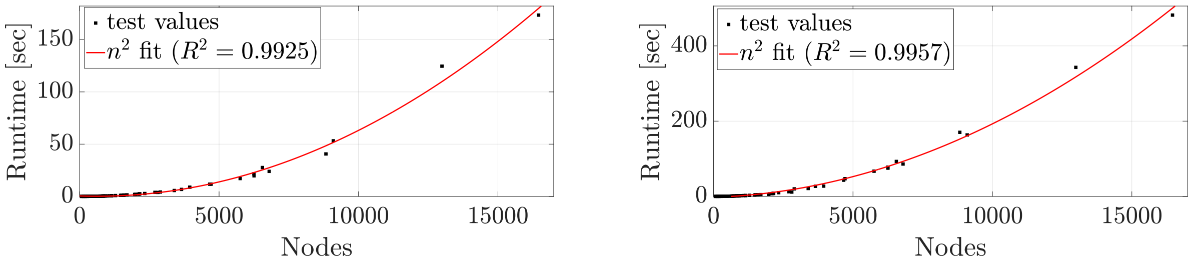


Figure 5.15: Performance comparison of greedy initialisers



(a)

(b)

Figure 5.16: Network creation time (a) and SA runtime (b) against node count.

for-hire vehicle trip records provided in [119] were exported and typical daily weekday trip count profiles which originate and terminate in the island of Manhattan NYC were identified for the entirety of the ride-sharing market.

By recording the MWIS node count for a varying request input, the effect of ride requests on the problem size for an FCI equal to one (supply=demand) was grasped, as shown in figure 5.18. Using the identified runtime trends outlined in figures 5.16a, 5.16b and 5.18, table 5.3 was compiled, which reports the time performance of the proposed methodology for varying request inputs. By assessing the request performance levels in table 5.3 fifty requests were chosen as the practical limit in Manhattan, since the proposed methodology produces ride-sharing solutions approximately within one minute, which is regarded as acceptable.

Table 5.3: MWIS instance node count and runtimes against varying request inputs.

Requests	Nodes	Network runtime [sec]	SA runtime [sec]	Total runtime [sec]
40	2500	5	10	15
45	3500	10	20	30
50	5000	15	50	65
55	6000	25	80	105
60	8000	45	120	165

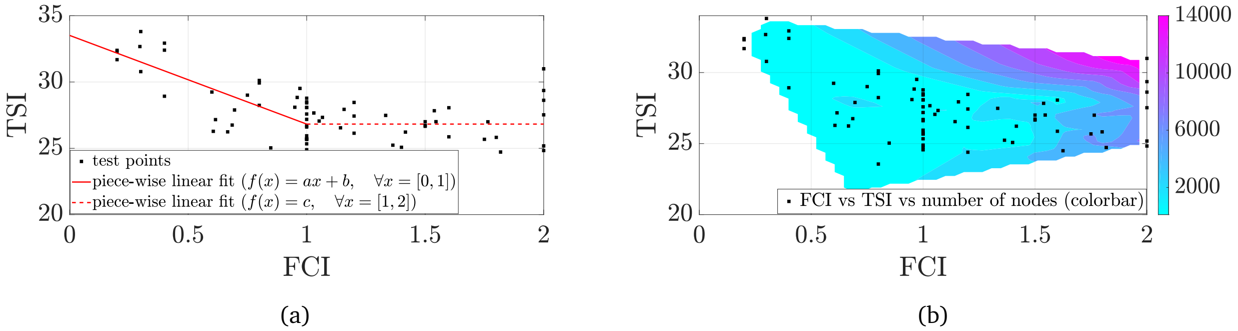


Figure 5.17: Trade surplus per serving vehicle and fleet coverage (a) and trade surplus per serving vehicle, fleet coverage and node numbers (b).

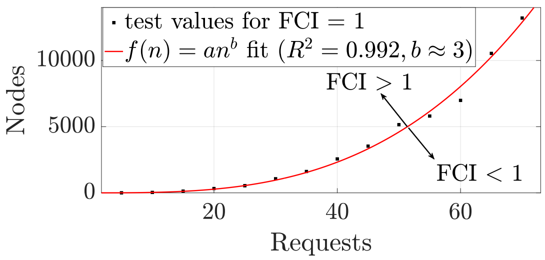


Figure 5.18: MWIS node count for a varying request input in Manhattan, NYC.

By examining the typical per-minute shared ride count in Manhattan in figure 5.19, it is observed that the demand surpasses the cutoff of fifty requests only during three distinct demand peaks, specifically during the morning, afternoon and evening. As such, since the highest peak narrowly exceeds a hundred shared trip rides, the proposed methodology can be practically implemented during peak hours when demand for rides exceeds supply (FCI $\leq$ 1) with an assignment duration interval Δ of thirty seconds for the entire Manhattan shared ride market.

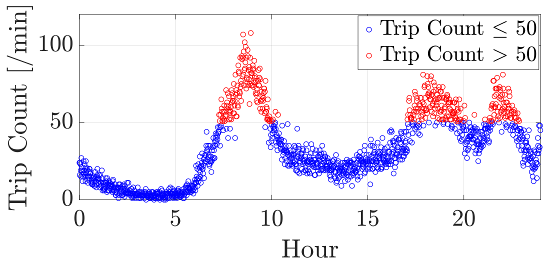


Figure 5.19: Typical Weekday Per-Minute Shared-Trip Count in Manhattan, NYC.

Nonetheless, the choice of a practical request cutoff value also depends on the error of the SA solution when compared to the exact solution. As observed in figure 5.18, the number of nodes increases almost in a cubic rate with an increasing number of requests. Also, the percentage error increases approximately in a linear fashion with an increasing number of nodes, as observed in figure 5.12a. As such, an instance of 100 requests might have a comparable SA total utility value when split in three instead of two instances of 50 requests. For reference, by running BC instances of 50 requests for FCI$= 1$, using the upper bound[8] and the best integer solution provided, the SA percentage error is pinpointed within $10\% - 20\%$ of the exact solution.

The practical implementation recommendations above assume pervasiveness of autonomous vehicles similar to the levels of current conventional ride-sharing platforms. Nonetheless, the adoption rate of autonomous vehicles in commercial ride-sharing is a conjecture. As such, plausible scenarios could involve mixed fleets and custom rider requirements. Even so, the algorithm proposed in section 5.4 is still applicable for such customisation as one could screen any preferences in the pre-matching stage (Section 5.4.1).

5.6 Summary

This chapter examined the implications of decision-making in the pricing and assignment operations in ride-sourcing systems. The chapter's objective was to identify how these two operations could be efficiently decided, to offer a high-quality service and relieve the overburdened system at periods of peak travel demand. The chapter investigated the effects of dynamic pricing via the use of ABM to achieve its objective. Also, it proposed a practically implementable local search method for assigning and pricing shared rides in tandem, based on CDAs.

Initially, in section 5.2, the study used an ABM to identify the emergent effects of dynamic pricing in urban transportation. The intelligent agents in the ABM were travellers choosing their transportation option. The traveller agents' choice process was modelled via the use of a three-level nested-logit model, deriving from the assumed market structure. The ABM was tested in the Greater London network, with scenarios including a dynamic pricing and a static pricing ride-sourcing firm, as well as public transport. The ABM study was able to identify the emergent properties of dynamic pricing during peak travel demand, such as shifting travellers to ride-sharing or public transport, whilst generating similar amounts of revenue with static pricing policies.

The dynamic pricing model in section 5.2 priced rides by maximising expected revenue per trip request. The expected revenue maximisation was achieved by leveraging knowledge about the discrete choice model used by the traveller agents. Nonetheless, the ABM implementation in section 5.2 only assumed one dynamically pricing ride-sourcing firm in each scenario, even in competition. By contrast, section 5.3 extended the model to account for multiple dynamically pricing firms operating concurrently, with the use of incomplete information games. Both models in sections 5.2 and 5.3 serve as simple bottom-up ABMs to identify emergent phenomena in autonomous ride-sourcing markets using dynamic pricing and simple traveller agent logic.

Building on the stylised fact of the demand shift towards ride-sharing when using dynamic pricing

[8] An exact solution for such an instance size was prohibitive due to combinatorial explosion. As such, a long-run upper bound and best integer solution of the BC algorithm before termination were used.

to relief an overburdened autonomous ride-sourcing platform, the chapter then presents a practically implementable pricing and assignment method for shared rides. The proposed assignment and pricing approach utilises a local search algorithm that solves a WDP MILP variant approximately in polynomial time by computing three-dimensional assignments to maximise trade surplus. By investigating the robustness of the proposed model, a GFP auction interface was derived which conveniently reduces to a stable three-dimensional assignment with minimal detours if riders report untruthful bids. The practicability of the proposed assignment and pricing method was demonstrated in a large urban setting such as Manhattan, NYC.

This chapter's focus was to investigate the interplay between decision problems in the operational time horizon, namely those of assignment and pricing. The study concludes that dynamic pricing contributes to a more sustainable ride-sourcing market during hours of peak travel. The pricing process could be more efficiently implemented in combination with the assignment of shared rides. The aforementioned operational efficiency is outlined in the proposed CDA method in section 5.4. The next chapter will summarise the contributions of the work and highlight how the methodology in this chapter fits the study's broader scope.

Chapter 6

Conclusions and Further Research

The aim of this concluding chapter is to revisit the motivations of the study outlined in Chapter 1 as depicted from the research gaps identified in Chapter 2 and review how these have been addressed throughout Chapters 3 to 5. In section 6.1, the chapter summarises the main findings and contributions of the study. The chapter also provides recommendations for further research through which the work presented in this study can be extended in section 6.2.

6.1 Main Findings and Contributions

Efficient ride-sourcing services are becoming increasingly relevant in urban transportation in recent years, in line with the rapid advances in telecommunications and smartphone technologies. These services are facilitated by TNCs, acting as digital platforms which connect available drivers to prospective travellers. The anticipated introduction of AVs in TNC services to reduce operational costs and increase profit margins will modify the existing structure and dynamics of the market. The footprint of these services on the urban transportation market is such that the implications of TNC services are of interest to operators and regulators alike. On the one hand, the TNCs are interested in improving their operations to offer a more reliable service and increase profits. On the other hand, regulators need to understand the implications of TNC decisions, to regulate accordingly in the interest of public welfare.

To facilitate the improvement of AV ride-sourcing services, and attain a better understanding of scenario-based implications, stakeholders require the aid of ABM simulations to act as digital twins of the market, as well as sophisticated mathematical models to improve operations. This study aims to provide simulation, and mathematical modelling means to ride-sourcing operators, regulators, and researchers in this domain. The main findings of this study and the contributions towards more efficient TNC operations can be summarised as follows:

Survey of Existing Ride-Sourcing Research

- Existing research in autonomous ride-sourcing lacks a modular framework describing the components and building sequence of ABMs for the different problems entangled within the domain. Such a framework is required to streamline future advancements as it will accelerate reproducibility of existing research and enable faster development and testing of sophisticated mathematical models in the field.
- Current fleet management mathematical models in the literature omit the inclusion of intelligent customer behavior in their design. This omission is consistently evident in both mathematical programming and reinforcement learning approaches, which are the two main avenues for efficient fleet management decisions and can result in over-estimating the benefits of idle vehicle redistribution processes to operators.
- Research on assignment and pricing of shared rides, lacks an efficient methodology which applies both operations in tandem at hours of peak travel. Existing methodologies either address each problem in isolation, or address both pricing and assignment as a single problem with the use of combinatorial auctions by ignoring the effects of detours due to the increasing computational complexity they entail. If sophisticated combinatorial auction models do include shared trip detours in their design, this can result in fairer dynamic pricing during to alleviate congestion and a more attractive ride-sharing service for travellers.

Methodological Contributions

- The study identifies the core components of ABMs in autonomous ride-sourcing systems and presents them in clear-cut classifications based on the fundamentals of agent-based modelling.
- The study provides a model building sequence of ABMs for tackling the different problems identified in autonomous ride-sourcing.
- An aggregated model is proposed which makes use of pickup wait times in identifying the minimum required fleet size for autonomous ride-sourcing fleets based on queuing theoretical implementations.
- The study proposes a model which incorporates customer behaviour and local competition to account for the notion of diminishing returns in the vehicle redistribution problem.
- The study models the vehicle redistribution problem as a non-linear minimum cost flow problem.
- The work presents a derivation of a convex space for the problem and its transformation into a convex minimum cost flow problem, which is solved using a pseudo-polynomial edge splitting algorithm.
- The study proposes a winner determination problem modelled for dynamic ride-sharing assignment, implementing a combinatorial double auction while considering the effect of detours on the valuations of auction participants.
- A local search algorithm is provided which produces approximate results in polynomial time using greedy heuristic solutions as initializers.
- The effects of shill bidding on the proposed combinatorial double auction are identified, followed

by a suggestion of a robust trip price determination methodology.

Model Application

- A study performing simulations is presented, which is structured using the proposed ABM framework, to identify upper bounds of the critical fleet size and the dynamics of pickup wait times for the urban areas of Manhattan, San Francisco, Paris and Barcelona. The results are compared with the expected outputs of the proposed aggregated queuing theoretical model.
- The non-linear minimum cost flow methodology to address the idle vehicle redistribution problem, is implemented within an ABM in a case study in the urban area of Manhattan, NYC with the use of local historical taxi demand data.
- A bottom-up ABM implementation is presented, to identify the emergent effects of dynamic pricing in urban transportation markets using simple operator and traveller agent logic. The model is implemented in the area of Greater London, using demand derived from a public transportation trip data-set.
- The local search algorithm proposed to address the assignment and pricing problem, is tested with historical taxi demand data from the urban area of Manhattan, NYC.

The methodology presented in this study can be used by TNCs, regulators or researchers in the field. For instance, the ABM framework and building sequence proposed in Chapter 3 can be used by TNCs to test the effectiveness of new models for their operations or the market impact of introducing new products in their services. The ABM framework is also useful for regulators that wish to gain foresight in the reaction of market participants to new regulation such as congestion charges within zones in the urban transport network. Furthermore, researchers developing sophisticated mathematical models can use the ABM framework to test their algorithms and cite it as a means for the reproducibility of their research.

The methodologies presented in Chapters 4 and 5 are capable of city-wide decision-making in ride-sourcing operations and can be useful to both operators and researchers conducting relevant studies. By testing the idle vehicle redistribution model presented in Chapter 4 with historical taxi data, the study infers the model can decrease customer wait times by more than 50%, and increase operational profit up to 10% with less than 20% increase in vehicle mileage when compared to baseline scenarios. Furthermore, the fast local search heuristic proposed to efficiently decide assignment and pricing operations concurrently in Chapter 5 can produce results that lie within 10% of the computationally expensive exact approach for practical implementations.

6.2 Further Research

This section outlines the various opportunities identified as suitable for further research from this study. These include methodologies to address the limitations of models presented in this study, as well as proposals for model extensions to tackle ride-sourcing problems with more localised implementations.

ABM Framework Extensions

The ABM framework presented in Chapter 3 can be extended further to provide readily-available solutions that cover a more comprehensive range of problems in the AV ride-sourcing spectrum. Specifically, a study listing the typical simulation set-ups that can be used to tackle the different decision problems in ride-sourcing can serve as a basis for future research development. Such a study could result in standardised ABM instances to test mathematical models across different papers and thus accelerate the production and reproducibility of research.

Points of interest for new research utilising ABM simulation are the computational trade-offs of different modules included in the ABM. As an example, simulating background traffic with background noise agents and a car following model can, in practice, offer more realism but could potentially increase the computational burden considerably. An analysis of these module inclusion trade-offs and alternative modelling remedies to limit the implications of omitting them would help future research. A complimentary research product to such analysis could be the identification of stylised facts in urban transportation markets. These stylised facts could be utilised to ascertain the level of realism of new ABMs.

Endogenous Congestion for Ride-Sourcing Simulations

The two most common assumptions in traffic modelling for ride-sourcing purposes include no endogenous implication on network velocities by ride-sourcing fleets; or using a car-following model and assuming the entire traffic is composed of discrete vehicles as agents in the model. Nonetheless, both approaches can have significant shortcomings depending on the context of implementation.

The assumption of no endogenous implication on network congestion by ride-sourcing fleets is valid when considering small fleet sizes compared to background flows. Nonetheless, as the popularity of ride-sourcing platforms increases and their transportation market share potentially attracts travellers from public transport or private vehicle modes, the assumption of no endogeneity in congestion can quickly become unfounded. Furthermore, car-following models with tens of thousands of vehicles moving through a simulation network as noise agents can overburden the computational process. Therefore, it is apparent that the development of a unified framework for discrete and continuous network flows would improve the accuracy and robustness of any ride-sourcing operation simulator.

Fleet Sizing ABM Study Extensions

The fleet sizing ABM model presented in Chapter 3 serves as a proof-of-concept; however, it could be extended to account for localised urban transportation environments, subject to data availability. Such extensions could consider a model accounting for the endogenous effects of congestion and background traffic as discussed above, the inclusion of alternative transport options, an account for intelligent customer behaviour or the use of electric charging points and an energy consumption model.

The extensions above would likely result in smaller critical fleet sizes than the experiments conducted in Chapter 3. Nonetheless, their inclusion could aid for useful cost-benefit studies of different fleet sizes if future studies also account for the operational costs and revenues of ride-sourcing services. Such research could identify the distribution of the monetised benefit for various fleet sizes by running simulations with multiple random seeds.

Reinforcement Learning for Intelligent Traveller Behaviour

The traveller behaviour used in methodological sections of this study is based on simple nested-logit models. These discrete choice models serve the proof-of-concept for the vehicle redistribution and the assignment and pricing methodologies presented in Chapters 4 and 5 respectively. Nonetheless, they do not capture the heterogeneity of the customer population, but rather serve as aggregated representations of the customer behaviour.

Future extensions on vehicle redistribution algorithms and assignment and pricing methodologies could use more sophisticated and localised traveller behaviours. Additional sophistication could be achieved by accounting for more detailed market structures and traveller-specific features, such as income level and type of work. Such features could be extracted using reinforcement learning algorithms within the ABM framework presented in Chapter 3, by setting the local market structure and deriving heterogeneous demand features based on census data. The outcome of such research could be trained traveller personas, which could better inform the ride-sourcing models presented in this study.

Multi-Agent Reinforcement Learning for Fleet Management

The methodology presented in Chapter 4 is a model-based optimization solution. Specifically, it serves as an aggregated representation of the supply-demand dynamics to estimate an efficient vehicle redistribution strategy. Nonetheless, the complexity of the supply-demand dynamics is such, that accurately modelling effects such as the heterogeneity of the traveller population and the dynamic variation of wait times through time is an almost intractable task.

The realm of multi-agent reinforcement learning might be suitable for addressing the above considerations. Research in reinforcement learning and multi-agent reinforcement learning has attempted to tackle idle vehicle redistribution; however, existing models do not capture the effects of customer intelligence and alternative transportation options. Furthermore, the use of coordinated vehicle policies, which attain to multi-agent reinforcement learning is a computationally demanding task that faces significant limitations when considering practically implementable scenarios.

Consequently, further research is required for using multi-agent reinforcement learning to identify optimal vehicle redistribution policies in competitive transport environments with intelligent travellers. A suitable candidate for training reinforcement learning agents in such research is the ABM framework presented in Chapter 3.

Assignment and Pricing CDA Model Extensions

In Chapter 5, the study proposed a local search algorithm to identify optimal assignments based on stated customer valuations for their trips. The computational efficiency and speed of the proposed local search method were sufficient for practical implementations. Nonetheless, both computational complexity and accuracy improvements are possible when exploring the breadth of meta-heuristics and machine learning algorithms in solving the proposed problem of ride-sharing auctions. Spatial clustering of requests, for example, could split much larger instances than the ones tested into parallel problems, which could be solved in a reasonable time, without compromising much of the efficiency of the algorithm.

Also, future research on ABM studies that use heterogeneous customer populations could focus on analysing bid behaviours in the proposed methodology (or a variant of it). Such studies could produce large data-sets of bidding and pricing solutions and better assess the effects of shill bidding to design more robust combinatorial auctions for ride-sourcing.

Appendix A

List of Publications

Table A.1: List of publications related to this thesis.

Study	Type	Chapter	Status
Karamanis, R., Cheong, H., Hu, S., Stettler, M. and Angeloudis, P., 2020. *Identifying Critical Fleet Sizes Using a Novel Agent-Based Modelling Framework for Autonomous Ride-Sourcing.* arXiv preprint arXiv:2011.11085.	Working paper	3	Online
Karamanis, R., Niknejad, A. and Angeloudis, P., 2017. *A fleet sizing algorithm for autonomous car sharing* (No. 17-02884). Transportation Research Board 96th Annual Meeting.	Conference paper	4	Accepted
Karamanis, R., Anastasiadis, E., Stettler, M. and Angeloudis, P., 2020. *Vehicle Redistribution in Ride-Sourcing Markets using Convex Minimum Cost Flows.* arXiv preprint arXiv:2006.07919.	Journal paper	4	In review[1]
Karamanis, R., Angeloudis, P., Sivakumar, A. and Stettler, M., 2018, November. *Dynamic Pricing in One-Sided Autonomous Ride-Sourcing Markets.* In 2018 21st International Conference on Intelligent Transportation Systems (ITSC) (pp. 3645-3650). IEEE.	Conference paper	5	Accepted
Karamanis, R., Angeloudis, P., Sivakumar, A. and Stettler, M., 2018. *Market dynamics between public transport and competitive ride-sourcing providers.* 7th Symposium of the European Association for Research in Transportation.	Conference paper	5	Accepted
Karamanis, R., Anastasiadis, E., Angeloudis, P. and Stettler, M., 2020. *Assignment and pricing of shared rides in ride-sourcing using combinatorial double auctions.* IEEE Transactions on Intelligent Transportation Systems.	Journal paper	5	Accepted

[1] In review at IEEE Transactions on Intelligent Transportation Systems.

Appendix B

Algorithms

FIFO Assignment

Algorithm 13 describes the FIFO assignment heuristic used in the methodology.

Algorithm 13 FIFO Assignment Heuristic

```
$C \neq \emptyset$                                  ▷ List of unassigned customers.
$V \neq \emptyset$                                  ▷ List of available vehicles.
$C' \leftarrow C$                                   ▷ Proxy list of unassigned customers.
for $c \in C$ do
    if $V = \emptyset$ then
        return
    end if
    if $c \in C'$ then
        $D = ShortestDistance(V, c)$    ▷ Get shortest distance from each vehicle in $V$ to customer $c$.
        $v = \arg\min_{i \in V} D(i)$   ▷ Get closest vehicle $v$ to customer $c$.
        $Match(c, v)$                   ▷ Match customer $c$ to vehicle $v$.
        $V \leftarrow V \setminus v$    ▷ Remove vehicle $v$ from available vehicles list.
        $C' \leftarrow C' \setminus c$  ▷ Remove customer $c$ from unassigned customers proxy list.
    end if
end for
Output: Vehicle-customer assignments.
```

Path-Finding Algorithms

Algorithms 14 and 15 were used to obtain shortest paths and shortest path costs in network calculations throughout the methodology in the study.

Algorithm 14 $A^{\star}$ Algorithm

```
$G = (V, E)$                              ▷ Network with node set $V$ and edge set $E$.
$s \in V$                                 ▷ Start node.
$t \in V$                                 ▷ End node.
$C \leftarrow \emptyset$                  ▷ Closed set.
$F \leftarrow \emptyset$                  ▷ Open set.
$F \leftarrow F \cup s$                   ▷ Add start node to open set.
$g(s) = 0$                                ▷ Cost to node $s$ from start node.
$h(s) = h(s, t)$                          ▷ Estimated cost$^1$ from node $i$ to end node $j$.
$f(s) = g(s) + h(s)$                      ▷ Total cost from node $s$ to end node $t$.
$P(s) \leftarrow \emptyset$               ▷ Empty map of cheapest parent node of node $s$.
while $F \neq \emptyset$ do
    $k = \arg\min_{i \in F} f(i)$         ▷ Get node with minimum total cost from open set.
    if $k = t$ then
        return $f(t) = f(k)$, $P^* = GetPath(P, k)$   ▷ Return minimum cost and shortest path.
    end if
    $F \leftarrow F \setminus k$
    $C \leftarrow C \cup k$
    for $i \in V | (k, i) \in E$ do       ▷ For each neighbouring node of $k$.
        if $i \in C$ then
            continue
        end if
        $d = g(k) + c_{k,i}$              ▷ Compute cost to node $i$ using the cost $c_{k,i}$ of edge $(k, i) \in E$.
        if $(i \in F) \wedge (d < g(i))$ then
            $F \leftarrow F \setminus i$
        end if
        if $(i \in C) \wedge (d < g(i))$ then
            $C \leftarrow C \setminus i$
        end if
        if $(i \notin F) \wedge (i \notin C)$ then
            $P(i) = k$                    ▷ Set $k$ as parent node of $i$.
            $F \leftarrow F \cup i$
            $g(i) = d$
            $h(i) = h(i, t)$
            $f(i) = g(i) + h(i)$
        end if
    end for
end while
Output: Shortest path $P^*$ with cost $f(t)$.
```

Algorithm 15 $GetPath(P, k)$ Algorithm

```
$P* \leftarrow \emptyset$
$P* \leftarrow k$
while $k \in P.keys$ do
    $k = P(k)$
    $P^* \leftarrow k \cup P^*$                ▷ Add node $k$ to the beginning of ordered set $P^*$.
end while
Output: Shortest path $P^*$.
```

Iterative Proportional Fitting Algorithm

The IPF algorithm (Algorithm 16) was used to identify OD matrices for trip counts in section 3.5.2.

Algorithm 16 Iterative Proportional Fitting (IPF) Algorithm

```
$P = (P_1, ...P_I)$                          ▷ Set $P$ of pickup counts from catchments $i \in I$.
$D = (D_1, ...D_J)$                          ▷ Set $D$ of drop-off counts in catchments $j \in J$.
$T_{i,j} = 1 \quad \forall i \in I, j \in J$         ▷ Array of origin destination trips.
for $i \in I$ do
    if $P_i = 0$ then
        $P_i = \mu$                          ▷ Set zero entries of pickup counts to a small number $\mu \leq 0.0001$.
    end if
end for
for $j \in J$ do
    if $D_j = 0$ then
        $D_j = \mu$                          ▷ Set zero entries of drop-off counts to a small number $\mu \leq 0.0001$.
    end if
end for
$\epsilon = 0.01$                            ▷ Set algorithm tolerance.
$\epsilon^P_{max} = 1$                       ▷ Set initial maximum proportional pickup error.
$\epsilon^D_{max} = 1$                       ▷ Set initial maximum proportional drop-off error.
$\hat{P}_i = \sum_{j \in J} T_{ij}$          ▷ Calculate pickup estimate.
while $(\epsilon^P_{max} > \epsilon) \vee (\epsilon^D_{max} > \epsilon)$ do
    for $i \in I$ do
        for $j \in J$ do
            $T_{ij} = T_{ij} \frac{P_i}{\hat{P}_i}$      ▷ Weight trip counts by pickup counts.
        end for
    end for
    $\hat{D}_j = \sum_{i \in I} T_{ij}$      ▷ Calculate drop-off estimate.
    for $i \in I$ do
        for $j \in J$ do
            $T_{ij} = T_{ij} \frac{D_j}{\hat{D}_j}$      ▷ Weight trip counts by drop-off counts.
        end for
    end for
    $\hat{P}_i = \sum_{j \in J} T_{ij}$
    $\hat{D}_j = \sum_{i \in I} T_{ij}$
    $\epsilon^P_{max} = max\big(\big|1 - \frac{\hat{P}_i}{P_i}\big|\big) \quad \forall i \in I$
    $\epsilon^D_{max} = max\big(\big|1 - \frac{\hat{D}_j}{D_j}\big|\big) \quad \forall j \in J$
end while
Output: Origin-Destination trip counts $T_{ij}$.
```

Network Simplex Algorithm

The network simplex algorithm shown in Algorithm 17 was used to find minimum cost flows in section 4.4.

Algorithm 17 Network Simplex Algorithm for Multigraphs

```
$G = (V, E)$    ▷ Multigraph $G$ with sets of vertices $V$ and edges $E$.
$|V| = n$    ▷ $n$ vertices in multigraph.
$s, t \in V$    ▷ Source $s$ and sink $t$ vertices in $V$
$b : V \to \mathbb{Z}$    ▷ Balance vector functions for each vertex $v \in V$
Step 1: Identify a starting Basic Feasible Solution (BFS). In a BFS, the flow $x$ satisfies the continuity and capacity constraints and is strictly greater than the lower bound and strictly lower than the upper bound in $n-1$ edges. These $n-1$ edges also form a spanning tree for $G$. $n-1$ edges are referred to as basic variables, whereas the rest are referred to as non-basic variables.
Step 2: Calculate vertex potentials $\pi_i \quad \forall i \in V$ with $\pi_s = 0$ and reduced costs $c^{\pi}_{ijk}$ for each edge $(i, j, k) \in E$.
Step 3: If the complimentary slackness optimality conditions listed in [124] are satisfied the current flow is optimal and $x = x^*$. If the complimentary slackness optimality conditions are not satisfied, choose the non-basic variable that most violates the optimality condition as a new basic variable.
Step 4: Identify the unique cycle formed by adding or subtracting flow from the new basic variable and using the continuity and capacity constraints augment the largest feasible flow through the cycle. After this a basic variable from the previous BFS becomes a non-basic variable.
Step 5: Repeat steps 2 to 4 until a flow which satisfies the complimentary slackness conditions is identified.
Output: Minimum cost flow $x^*$.
```

The following example illustrates how Algorithm 17 can be used to find the minimum cost flow on a multigraph[2]. Consider the multigraph shown in figure B.1. For each edge (i, j, k) shown in figure B.1, the notation $(u_{ijk}, c_{ijk})^k$ refers to the upper bound u_{ijk} and cost c_{ijk}. For simplicity, all the lower bounds l_{ijk} are set to zero. Initially, (Step 1), a BFS solution is identified as shown in figure B.2.

The complimentary slackness optimality conditions for an optimal flow x^* as listed in [124], are as follows:

$$c^{\pi}_{ijk} \geq 0 \quad if \quad x^*_{i,j,k} = u_{ijk} \tag{B.1}$$

$$c^{\pi}_{ijk} = 0 \quad if \quad l_{ijk} < x^*_{i,j,k} < u_{ijk} \tag{B.2}$$

$$c^{\pi}_{ijk} \leq 0 \quad if \quad x^*_{i,j,k} = l_{ijk} \tag{B.3}$$

To calculate the vertex potentials π_i for vertices $i \in V \setminus s$ ($\pi_s = 0$), the reduced cost equation is used:

$$C^{\pi}_{ij} = \begin{bmatrix} \pi_1 & \pi_2 & \pi_t \end{bmatrix} \times A_{ij} - C_{ij} \tag{B.4}$$

[2] Multigraphs are directed graphs which may include parallel edges between vertices.

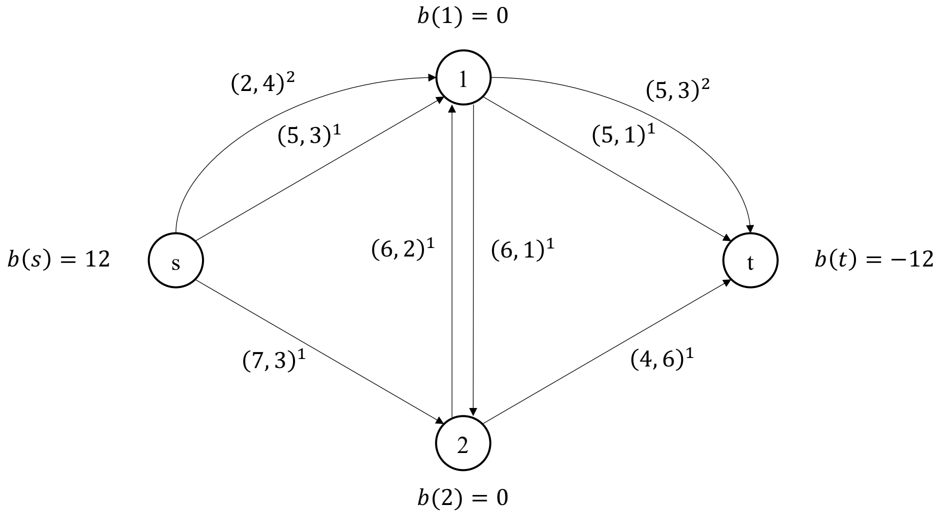


Figure B.1: Multigraph example properties.

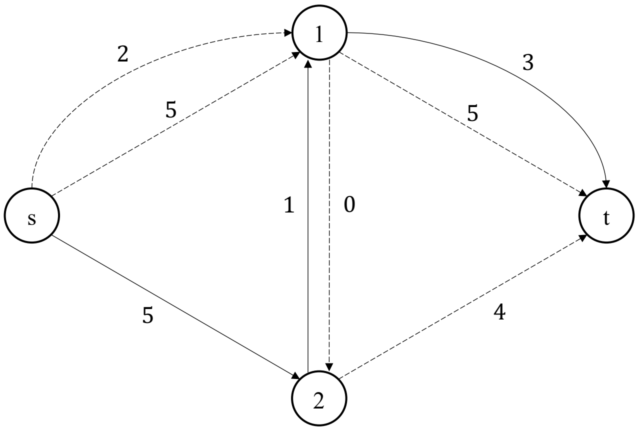


Figure B.2: Initial BFS. Dashed edges represent non-basic variables, while non-dashed edges represent the basic variables.

Where C^{π}_{ij} and C_{ij} are the matrices of reduced costs and edge costs for edges between vertices i and j. A_{ij} is the matrix of the factors of each edge between vertices i and j and the balance equations (excluding $b(s)$). For vertices 1, 2 and t in the example, the following balance equations hold:

$$x_{1t1} + x_{1t2} + x_{121} - x_{s11} - x_{s12} - x_{211} = 0 \tag{B.5}$$

$$x_{2t1} + x_{211} - x_{s21} - x_{121} = 0 \tag{B.6}$$

$$- x_{1t1} - x_{1t2} - x_{2t1} = -12 \tag{B.7}$$

For basic variables, the reduced cost c^{π}_{ijk} equals zero, whereas the reduced cost for any non-basic variable needs to be calculated using equation B.4. The reduced cost equation for edges between vertices s and 1 is as follows:

$$\begin{bmatrix} c^{\pi}_{s11} & c^{\pi}_{s12} \end{bmatrix} = \begin{bmatrix} \pi_1 & \pi_2 & \pi_t \end{bmatrix} \times \begin{bmatrix} -1 & -1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} - \begin{bmatrix} 3 & 4 \end{bmatrix} \tag{B.8}$$

Furthermore, unlike edges $(s, 1)^1$ and $(s, 1)^2$ in equation (B.8), since edge $(s, 2)^1$ is a basic variable, its reduced cost equation is as follows:

$$0 = \begin{bmatrix} \pi_1 & \pi_2 & \pi_t \end{bmatrix} \times \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix} - 3 \tag{B.9}$$

Solving equation (B.9) for the potential variables, π_2 equals to -3. Similarly, the reduced cost equation for edges between vertices 1 and 2 is as follows:

$$\begin{bmatrix} c^{\pi}_{121} & 0 \end{bmatrix} = \begin{bmatrix} \pi_1 & \pi_2 & \pi_t \end{bmatrix} \times \begin{bmatrix} 1 & -1 \\ -1 & 1 \\ 0 & 0 \end{bmatrix} - \begin{bmatrix} 1 & 2 \end{bmatrix} \tag{B.10}$$

Using $\pi_2 = -3$ in equation (B.10), the value of π_1 equals -5 and c^{π}_{121} equals -3. Revisiting equation (B.8), the values of c^{π}_{s11} and c^{π}_{s12} result to 2 and 4 respectively. Consequently, c^{π}_{s11}, c^{π}_{s12} and c^{π}_{121} satisfy the complimentary slackness optimality conditions in equations (B.1)-(B.3). The reduced cost equation for the edges between vertices 1 and t is as follows:

$$\begin{bmatrix} c^{\pi}_{1t1} & 0 \end{bmatrix} = \begin{bmatrix} \pi_1 & \pi_2 & \pi_t \end{bmatrix} \times \begin{bmatrix} 1 & 1 \\ 0 & 0 \\ -1 & -1 \end{bmatrix} - \begin{bmatrix} 1 & 3 \end{bmatrix} \tag{B.11}$$

Solving this equation for π_t it results to -8. Using the known values of π_1, π_2 and π_3, the value of c^{π}_{1t1} equals 2, which satisfies the complimentary slackness optimality condition in equation (B.1). Finally, the reduced cost equation for the non-basic edge between vertices 2 and t is as follows:

$$c^{\pi}_{2t1} = \begin{bmatrix} \pi_1 & \pi_2 & \pi_t \end{bmatrix} \times \begin{bmatrix} 0 \\ 1 \\ -1 \end{bmatrix} - 6 \tag{B.12}$$

Solving for c^{π}_{2t1}, equation (B.12) results to -1, which violates the complimentary slackness optimality condition in equation (B.1). Consequently, edge $(2, t)^1$ needs to enter the basic variable set, by subtracting θ units of flow through a cycle. Figure B.3 shows that such a cycle exists between edges $(2, t)^1$, $(2, 1)^1$ and $(1, t)^2$.

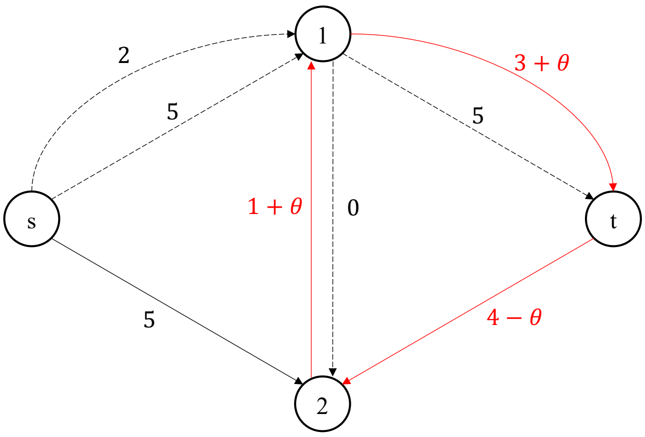


Figure B.3: Cycle for subtracting θ units of flow from edge $(2, t)^1$.

Using the edge capacities, the maximum units of flow θ to be augmented through the cycle outlined in figure B.3 is 2. This results to the following BFS outlined in figure B.4. As observed in figure B.4, since the flows between vertices s and 2 did not change and edge $(s, 2)^1$ is still a basic variable, equation (B.9) is still valid, resulting to $\pi_2 = -3$. Similarly, equation (B.10) is still valid, since edges $(1, 2)^1$ and $(2, 1)^1$ are still non-basic and basic variables respectively. Consequently, π_1 equals -5 and c^{π}_{121} equals -3 as before. As edges $(s, 1)^1$ and $(s, 1)^2$ are still non-basic variables, this also implies that equation (B.8) is still valid, with the values of c^{π}_{s11} and c^{π}_{s12} to equal 2 and 4 respectively as before. Consequently, c^{π}_{s11}, c^{π}_{s12} and c^{π}_{121} still satisfy the complimentary slackness optimality conditions in equations (B.1)-(B.3)

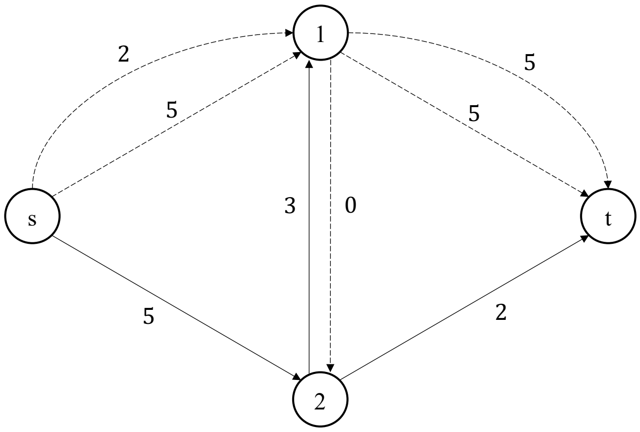


Figure B.4: Resulting BFS after subtracting 2 units of flow from edge $(2, t)^1$.

As edge $(2, t)^1$ is now a basic variable, its reduced cost equation transforms to the following:

$$0 = \begin{bmatrix} \pi_1 & \pi_2 & \pi_t \end{bmatrix} \times \begin{bmatrix} 0 \\ 1 \\ -1 \end{bmatrix} - 6 \tag{B.13}$$

Using the known value of π_2 and solving equation (B.13) for π_t, this results to $\pi_t = -9$. Finally, the reduced cost equations for edges between vertices 1 and t are as follows:

$$\begin{bmatrix} c^{\pi}_{1t1} & c^{\pi}_{1t2} \end{bmatrix} = \begin{bmatrix} \pi_1 & \pi_2 & \pi_t \end{bmatrix} \times \begin{bmatrix} 1 & 1 \\ 0 & 0 \\ -1 & -1 \end{bmatrix} - \begin{bmatrix} 1 & 3 \end{bmatrix} \tag{B.14}$$

Using the known values of π_1 and π_t in equation (B.14), c^{π}_{1t1} and c^{π}_{1t2} result to 3 and 1 respectively. Since the values of c^{π}_{1t1} and c^{π}_{1t2} also satisfy the complimentary slackness optimality conditions in equations (B.1)-(B.3), as c^{π}_{s11}, c^{π}_{s12} and c^{π}_{121} do, the BFS shown in figure B.4 constitutes the optimal solution for the minimum cost flow problem.

Bibliography

[1] S. Illgen and M. Höck, "Literature review of the vehicle relocation problem in one-way car sharing networks," *Transportation Research Part B: Methodological*, vol. 120, pp. 193–204, 2019. xv, 4, 5, 16

[2] J. J. Yu, A. Y. Lam, and Z. Lu, "Double Auction-based Pricing Mechanism for Autonomous Vehicle Public Transportation System," *IEEE Transactions on Intelligent Vehicles*, vol. 3, no. 2, pp. 151–162, 2018. xvi, 17, 19, 25, 26, 98, 100, 110, 113, 115

[3] M. Maciejewski, J. Bischoff, and K. Nagel, "An assignment-based approach to efficient real-time city-scale taxi dispatching," *IEEE Intelligent Systems*, vol. 31, no. 1, pp. 68–77, 2016. 1

[4] L. Zha, Y. Yin, and H. Yang, "Economic analysis of ride-sourcing markets," *Transportation Research Part C: Emerging Technologies*, vol. 71, pp. 249–266, 2016. 1

[5] S. Shaheen and A. Cohen, "Shared ride services in north america: definitions, impacts, and the future of pooling," *Transport Reviews*, vol. 39, no. 4, pp. 427–442, 2019. 1

[6] R. Roson, "Two-Sided Markets: A Tentative Survey," *Review of Network Economics*, vol. 4, no. 2, pp. 142–160, 2005. 2

[7] C. Shapiro, S. Carl, H. R. Varian, *et al.*, *Information rules: a strategic guide to the network economy*. Harvard Business Press, 1998. 2

[8] R. Garud, A. Kumaraswamy, A. Roberts, and L. Xu, "Liminal movement by digital platform-based sharing economy ventures: The case of uber technologies," *Strategic Management Journal*, 2020. 2

[9] Goldman Sachs, "Ride-hailing gross revenue worldwide in 2016, by key operator (in billion U.S. dollars).," 2016. 2

[10] Crunchbase, "Uber Funding Rounds," 2018. 2

[11] L. Hook, "Uber pares quarterly losses and lifts revenues," feb 2018. 2

[12] H. Qiu, R. Li, and J. Zhao, "Dynamic pricing in shared mobility on demand service," *arXiv:1802.03559*, 2018. 2, 17, 27, 53

[13] M. K. Chen, “Dynamic pricing in a labor market: Surge pricing and flexible work on the uber platform,” in *Proceedings of the 2016 ACM Conference on Economics and Computation*, pp. 455–455, 2016. 2, 53

[14] T. D. Chen and K. M. Kockelman, “Management of a Shared, Autonomous, Electric Vehicle Fleet: Implications of Pricing Schemes,” *95th Annual Meeting of the Transportation Research Board*, no. 2572, pp. 37–46, 2016. 2, 89

[15] P. M. Bösch, F. Becker, H. Becker, and K. W. Axhausen, “Cost-based analysis of autonomous mobility services,” *Transport Policy*, vol. 64, pp. 76–91, 2018. 2

[16] H. Wang and H. Yang, “Ridesourcing systems : A framework and review,” *Transportation Research Part B*, vol. 129, pp. 122–155, 2019. 3, 9, 10, 17, 20, 29

[17] P. Santi, G. Resta, M. Szell, S. Sobolevsky, S. H. Strogatz, and C. Ratti, “Quantifying the benefits of vehicle pooling with shareability networks,” *Proceedings of the National Academy of Sciences*, vol. 111, no. 37, pp. 13290–13294, 2014. 6, 22, 23, 32, 98, 99

[18] S. Narayanan, E. Chaniotakis, and C. Antoniou, “Shared autonomous vehicle services: A comprehensive review,” *Transportation Research Part C: Emerging Technologies*, vol. 111, pp. 255–293, 2020. 9, 10, 15, 27

[19] J. Von Neumann, A. W. Burks, *et al.*, “Theory of self-reproducing automata,” *IEEE Transactions on Neural Networks*, vol. 5, no. 1, pp. 3–14, 1966. 11

[20] E. R. Berlekamp, J. H. Conway, and R. K. Guy, *Winning ways for your mathematical plays*, vol. 1. CRC Press, 2018. 11

[21] E. Bonabeau, “Agent-based modeling: Methods and techniques for simulating human systems,” *Proceedings of the national academy of sciences*, vol. 99, no. suppl 3, pp. 7280–7287, 2002. 11

[22] C. M. Macal and M. J. North, “Tutorial on agent-based modelling and simulation,” *Journal of Simulation*, vol. 4, no. 3, pp. 151–162, 2010. 11, 28

[23] A. T. Crooks and A. J. Heppenstall, “Introduction to agent-based modelling,” in *Agent-based models of geographical systems*, pp. 85–105, Springer, 2012. 12

[24] M. G. McNally, “The four step model,” *Handbook of transport modelling*, vol. 1, pp. 35–41, 2000. 13

[25] J. Auld, M. Hope, H. Ley, V. Sokolov, B. Xu, and K. Zhang, “POLARIS : Agent-based modeling framework development and implementation for integrated travel demand and network and operations simulations,” *Transportation Research Part C: Emerging Technologies*, vol. 64, pp. 101–116, 2016. 13

[26] M. Fellendorf and P. Vortisch, “Microscopic traffic flow simulator vissim,” in *Fundamentals of traffic simulation*, pp. 63–93, Springer, 2010. 13

[27] K. Nagel, R. J. Beckman, and C. L. Barrett, "Transims for urban planning," in *6th international conference on computers in urban planning and urban management, Venice, Italy*, 1999. 13

[28] M. Balmer, K. W. Axhausen, and K. Nagel, "Agent-based demand-modeling framework for large-scale microsimulations," *Transportation Research Record*, vol. 1985, no. 1, pp. 125–134, 2006. 13

[29] S. Burgess, E. Fernandez-Corugedo, C. Groth, R. Harrison, F. Monti, K. Theodoridis, M. Waldron, *et al.*, "The bank of england's forecasting platform: Compass, maps, ease and the suite of models," *documento de trabajo*, no. 471, 2013. 13

[30] G. Fagiolo, M. Guerini, F. Lamperti, A. Moneta, and A. Roventini, "Validation of agent-based models in economics and finance," in *Computer Simulation Validation*, pp. 763–787, Springer, 2019. 13

[31] N. G. Mankiw, "The macroeconomist as scientist and engineer," *Journal of Economic Perspectives*, vol. 20, no. 4, pp. 29–46, 2006. 13

[32] A. Haldane, "The dappled world," *Shackle Biennial Memorial Lecture*, 2016. 13

[33] G. Fagiolo and A. Roventini, "Macroeconomic Policy in DSGE and Agent-Based Models Redux: New Developments and Challenges Ahead," *Journal of Artificial Societies and Social Simulation*, vol. 20, no. 1, p. 1, 2017. 13

[34] H. A. Simon, "A behavioral model of rational choice," *The quarterly journal of economics*, vol. 69, no. 1, pp. 99–118, 1955. 13

[35] A. Turrell, "Agent-based models: understanding the economy from the bottom up," *Bank of England Quarterly Bulletin*, p. Q4, 2016. 13

[36] K. Braun-Munzinger, Z. Liu, and A. Turrell, "An agent-based model of dynamics in corporate bond trading," 2016. 13

[37] R. Baptista, J. D. Farmer, M. Hinterschweiger, K. Low, D. Tang, and A. Uluc, "Macroprudential policy in an agent-based model of the uk housing market," 2016. 13

[38] D. Adam, "Special report: The simulations driving the world's response to covid-19.," *Nature*, 2020. 13

[39] N. Ferguson, D. Laydon, G. Nedjati Gilani, N. Imai, K. Ainslie, M. Baguelin, S. Bhatia, A. Boonyasiri, Z. Cucunuba Perez, G. Cuomo-Dannenburg, *et al.*, "Report 9: Impact of non-pharmaceutical interventions (npis) to reduce covid19 mortality and healthcare demand," 2020. 13, 14

[40] S. L. Chang, N. Harding, C. Zachreson, O. M. Cliff, and M. Prokopenko, "Modelling transmission and control of the covid-19 pandemic in australia," *arXiv preprint arXiv:2003.10218*, 2020. 14

[41] W. Shen and C. Lopes, “Managing autonomous mobility on demand systems for better passenger experience,” in *International conference on principles and practice of multi-agent systems*, pp. 20–35, Springer, 2015. 14, 24

[42] M. W. Levin, K. M. Kockelman, S. D. Boyles, and T. Li, “A general framework for modeling shared autonomous vehicles with dynamic network-loading and dynamic ride-sharing application,” *Computers, Environment and Urban Systems*, vol. 64, pp. 373–383, 2017. 14, 16, 24, 54

[43] H. Dia and F. Javanshour, “Autonomous shared mobility-on-demand: Melbourne pilot simulation study,” *Transportation Research Procedia*, vol. 22, pp. 285–296, 2017. 14, 24

[44] D. J. Fagnant and K. M. Kockelman, “The travel and environmental implications of shared autonomous vehicles, using agent-based model scenarios,” *Transportation Research Part C: Emerging Technologies*, vol. 40, pp. 1–13, 2013. 14, 24

[45] D. J. Fagnant, K. M. Kockelman, and P. Bansal, “Operations of shared autonomous vehicle fleet for austin, texas, market,” *Transportation Research Record*, vol. 2563, no. 1, pp. 98–106, 2015. 14, 24

[46] P. M. Boesch, F. Ciari, and K. W. Axhausen, “Autonomous vehicle fleet sizes required to serve different levels of demand,” *Transportation Research Record*, vol. 2542, no. 1, pp. 111–119, 2016. 14, 24

[47] L. Martinez and P. Crist, “Urban mobility system upgrade–how shared self-driving cars could change city traffic,” in *International Transport Forum, Paris*, 2015. 14, 24

[48] L. Martinez and J. Viegas, “Shared mobility: innovation for livable cities,” in *International Transport Forum*, 2016. 14, 24

[49] S. Hörl, “Agent-based simulation of autonomous taxi services with dynamic demand responses,” *Procedia Computer Science*, vol. 109, pp. 899–904, 2017. 14, 21, 24

[50] T. D. Chen and K. M. Kockelman, “Management of a shared autonomous electric vehicle fleet: Implications of pricing schemes,” *Transportation Research Record*, vol. 2572, no. 1, pp. 37–46, 2016. 14, 24, 30

[51] M. Maciejewski and J. Bischoff, “Congestion effects of autonomous taxi fleets,” 2016. 14, 24

[52] T. D. Chen, K. M. Kockelman, and J. P. Hanna, “Operations of a shared, autonomous, electric vehicle fleet: Implications of vehicle & charging infrastructure decisions,” *Transportation Research Part A: Policy and Practice*, vol. 94, pp. 243–254, 2016. 14, 21, 24

[53] M. Maciejewski, J. M. Salanova, J. Bischoff, and M. Estrada, “Large-scale microscopic simulation of taxi services. berlin and barcelona case studies,” *Journal of Ambient Intelligence and Humanized Computing*, vol. 7, no. 3, pp. 385–393, 2016. 15, 24

[54] J. Liu, K. M. Kockelman, P. M. Boesch, and F. Ciari, "Tracking a system of shared autonomous vehicles across the austin, texas network using agent-based simulation," *Transportation*, vol. 44, no. 6, pp. 1261–1278, 2017. 15, 24, 30

[55] D. J. Fagnant and K. M. Kockelman, "Dynamic ride-sharing and fleet sizing for a system of shared autonomous vehicles in austin, texas," *Transportation*, vol. 45, no. 1, pp. 143–158, 2018. 15, 21, 24

[56] R. Zhang and M. Pavone, "Control of robotic mobility-on-demand systems: a queueing-theoretical perspective," *The International Journal of Robotics Research*, vol. 35, no. 1-3, pp. 186–203, 2016. 15, 16, 24, 25, 35

[57] J. Wen, J. Zhao, and P. Jaillet, "Rebalancing shared mobility-on-demand systems: A reinforcement learning approach," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 220–225, IEEE, 2017. 15, 16, 24, 25

[58] K. Lin, R. Zhao, Z. Xu, and J. Zhou, "Efficient large-scale fleet management via multi-agent deep reinforcement learning," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1774–1783, 2018. 15, 16, 24, 25, 32

[59] J. Alonso-Mora, A. Wallar, and D. Rus, "Predictive routing for autonomous mobility-on-demand systems with ride-sharing," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3583–3590, IEEE, 2017. 15, 16, 24, 25, 32, 35

[60] A. Wallar, M. Van Der Zee, J. Alonso-Mora, and D. Rus, "Vehicle rebalancing for mobility-on-demand systems with ride-sharing," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4539–4546, IEEE, 2018. 15, 16, 24, 25, 32, 35, 67, 79, 80

[61] R. Iglesias, F. Rossi, K. Wang, D. Hallac, J. Leskovec, and M. Pavone, "Data-driven model predictive control of autonomous mobility-on-demand systems," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1–7, IEEE, 2018. 15, 16, 24, 25, 32

[62] F. Dandl, M. Hyland, K. Bogenberger, and H. S. Mahmassani, "Evaluating the impact of spatio-temporal demand forecast aggregation on the operational performance of shared autonomous mobility fleets," *Transportation*, vol. 46, no. 6, pp. 1975–1996, 2019. 15

[63] H. Yang, S. C. Wong, and K. I. Wong, "Demand-supply equilibrium of taxi services in a network under competition and regulation," *Transportation Research Part B: Methodological*, vol. 36, no. 9, pp. 799–819, 2002. 15, 17, 36

[64] M. Dell'Amico, E. Hadjicostantinou, M. Iori, and S. Novellani, "The bike sharing rebalancing problem: Mathematical formulations and benchmark instances," *Omega*, vol. 45, pp. 7–19, 2014. 16, 54

[65] M. Nourinejad, S. Zhu, S. Bahrami, and M. J. Roorda, "Vehicle relocation and staff rebalancing in one-way carsharing systems," *Transportation Research Part E: Logistics and Transportation Review*, vol. 81, pp. 98–113, 2015. 16, 54

[66] R. Nair and E. Miller-Hooks, "Fleet management for vehicle sharing operations," *Transportation Science*, vol. 45, no. 4, pp. 524–540, 2011. 16

[67] J. Pfrommer, J. Warrington, G. Schildbach, and M. Morari, "Dynamic vehicle redistribution and online price incentives in shared mobility systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 4, pp. 1567–1578, 2014. 16

[68] D. J. Fagnant and K. M. Kockelman, "The travel and environmental implications of shared autonomous vehicles, using agent-based model scenarios," *Transportation Research Part C: Emerging Technologies*, vol. 40, pp. 1–13, 2014. 16

[69] R. Iglesias, F. Rossi, R. Zhang, and M. Pavone, "A bcmp network approach to modeling and controlling autonomous mobility-on-demand systems," *The International Journal of Robotics Research*, vol. 38, no. 2-3, pp. 357–374, 2019. 16, 25, 35

[70] A. Braverman, J. G. Dai, X. Liu, and L. Ying, "Empty-car routing in ridesharing systems," *Operations Research*, vol. 67, no. 5, pp. 1437–1452, 2019. 16, 25, 35

[71] X. Yu, S. Gao, X. Hu, and H. Park, "A Markov decision process approach to vacant taxi routing with e-hailing," *Transportation Research Part B: Methodological*, vol. 121, pp. 114–134, 2019. 16, 25

[72] J.-C. Rochet and J. Tirole, "Platform competition in two-sided markets," *Journal of the european economic association*, vol. 1, no. 4, pp. 990–1029, 2003. 17

[73] G. W. Douglas, "Price regulation and optimal service standards: The taxicab industry," *Journal of Transport Economics and Policy*, pp. 116–127, 1972. 17

[74] J. C. Castillo, D. Knoepfle, and G. Weyl, "Surge pricing solves the wild goose chase," in *Proceedings of the 2017 ACM Conference on Economics and Computation*, pp. 241–242, 2017. 17

[75] J. Zhang, D. Wen, and S. Zeng, "A Discounted Trade Reduction Mechanism for Dynamic Ridesharing Pricing," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 6, pp. 1586–1595, 2016. 17, 19, 26

[76] A. Y. Lam, "Combinatorial auction-based pricing for multi-tenant autonomous vehicle public transportation system," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 3, pp. 859–869, 2016. 17, 18, 19, 25, 26

[77] M. Asghari and C. Shahabi, "An On-line Truthful and Individually Rational Pricing Mechanism for Ride-sharing," *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pp. 1–10, 2017. 17, 19, 25, 26

[78] M. Xia, J. Stallaert, and A. B. Whinston, "Solving the combinatorial double auction problem," *European Journal of Operational Research*, vol. 164, no. 1, pp. 239–251, 2005. 17

[79] D. Lehmann, R. Müller, and T. Sandholm, *The Winner Determination Problem*. 2005. 18

[80] S. Banerjee, C. Riquelme, and R. Johari, "Pricing in ride-share platforms: A queueing-theoretic approach," *Available at SSRN 2568258*, 2015. 18, 53

[81] Z. Fang, L. Huang, and A. Wierman, "Prices and subsidies in the sharing economy," *Performance Evaluation*, vol. 136, p. 102037, 2019. 18

[82] C. Yan, H. Zhu, N. Korolko, and D. Woodard, "Dynamic pricing and matching in ride-hailing platforms," *Naval Research Logistics (NRL)*, 2019. 18, 23

[83] G. P. Cachon, K. M. Daniels, and R. Lobel, "The role of surge pricing on a service platform with self-scheduling capacity," *Manufacturing & Service Operations Management*, vol. 19, no. 3, pp. 368–384, 2017. 18

[84] L. Zha, Y. Yin, and Z. Xu, "Geometric matching and spatial pricing in ride-sourcing markets," *Transportation Research Part C: Emerging Technologies*, vol. 92, pp. 58–75, 2018. 18, 40, 64

[85] J. Bai, K. C. So, C. S. Tang, X. Chen, and H. Wang, "Coordinating supply and demand on an on-demand service platform with impatient customers," *Manufacturing & Service Operations Management*, vol. 21, no. 3, pp. 556–570, 2019. 18

[86] H. Ma, F. Fang, and D. C. Parkes, "Spatio-temporal pricing for ridesharing platforms," *arXiv preprint arXiv:1801.04015*, 2018. 18

[87] O. Besbes, F. Castro, and I. Lobel, "Surge pricing and its spatial supply response," *Columbia Business School Research Paper*, no. 18-25, 2019. 18

[88] H. Guda and U. Subramanian, "Your uber is arriving: Managing on-demand workers through surge pricing, forecast communication, and worker incentives," *Management Science*, vol. 65, no. 5, pp. 1995–2014, 2019. 18

[89] K. Bimpikis, O. Candogan, and D. Saban, "Spatial pricing in ride-sharing networks," *Operations Research*, vol. 67, no. 3, pp. 744–769, 2019. 18

[90] A. Kleiner, B. Nebel, and V. A. Ziparo, "A Mechanism for Dynamic Ride Sharing based on Parallel Auctions," in *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*, pp. 266–272, 2011. 19, 26

[91] D. Zhao, D. Zhang, and E. H. Gerding, "Incentives in Ridesharing with Deficit Control," in *Proceedings of the 2014 International conference on Autonomous agents and multi-agent systems*, 2014. 19, 26

[92] S. de Vries and R. Vohra, “Combinatorial auctions: A survey,” *INFORMS Journal on Computing*, vol. 15, pp. 284–309, 08 2003. 19, 109, 110

[93] M. Asghari, D. Deng, C. Shahabi, U. Demiryurek, and Y. Li, “Price-aware real-time ride-sharing at scale: an auction-based approach,” in *Proceedings of the 24th ACM SIGSPATIAL international conference on advances in geographic information systems*, pp. 1–10, 2016. 19, 26

[94] L. Ramshaw and R. E. Tarjan, “On minimum-cost assignments in unbalanced bipartite graphs,” *HP Labs, Palo Alto, CA, USA, Tech. Rep. HPL-2012-40R1*, 2012. 20

[95] H. W. Kuhn, “The hungarian method for the assignment problem,” *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955. 20

[96] K. M. Gurumurthy and K. M. Kockelman, “Analyzing the dynamic ride-sharing potential for shared autonomous vehicle fleets using cellphone data from orlando, florida,” *Computers, Environment and Urban Systems*, vol. 71, pp. 177–185, 2018. 21

[97] L. M. Mendes, M. R. Bennàssar, and J. Y. Chow, “Comparison of light rail streetcar against shared autonomous vehicle fleet for brooklyn–queens connector in new york city,” *Transportation Research Record*, vol. 2650, no. 1, pp. 142–151, 2017. 21

[98] J. P. Dickerson, K. A. Sankararaman, A. Srinivasan, and P. Xu, “Allocation problems in ride-sharing platforms: Online matching with offline reusable resources,” in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 21

[99] D. Bertsimas, P. Jaillet, and S. Martin, “Online vehicle routing: The edge of optimization in large-scale applications,” *Operations Research*, vol. 67, no. 1, pp. 143–162, 2019. 21

[100] Z. Xu, Z. Li, Q. Guan, D. Zhang, Q. Li, J. Nan, C. Liu, W. Bian, and J. Ye, “Large-scale order dispatch in on-demand ride-hailing platforms: A learning and planning approach,” in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 905–913, 2018. 21, 22

[101] G. Lyu, W. C. Cheung, C.-P. Teo, and H. Wang, “Multi-objective online ride-matching,” *Available at SSRN 3356823*, 2019. 21, 22

[102] H. Hosni, J. Naoum-Sawaya, and H. Artail, “The shared-taxi problem: Formulation and solution methods,” *Transportation Research Part B: Methodological*, vol. 70, pp. 303–318, 2014. 22

[103] S. Ma, Y. Zheng, and O. Wolfson, “Real-time city-scale taxi ridesharing,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 7, pp. 1782–1795, 2014. 22, 23

[104] D. Pelzer, J. Xiao, D. Zehe, M. H. Lees, A. C. Knoll, and H. Aydt, “A partition-based match making algorithm for dynamic ridesharing,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 5, pp. 2587–2598, 2015. 22, 23

[105] A. Simonetto, J. Monteil, and C. Gambella, “Real-time city-scale ridesharing via linear assignment problems,” *Transportation Research Part C: Emerging Technologies*, vol. 101, pp. 208–232, 2019. 22, 23

[106] J. Jung, R. Jayakrishnan, and J. Y. Park, “Dynamic shared-taxi dispatch algorithm with hybrid-simulated annealing,” *Computer-Aided Civil and Infrastructure Engineering*, vol. 31, no. 4, pp. 275–291, 2016. 23

[107] J. Alonso-Mora, S. Samaranayake, A. Wallar, E. Frazzoli, and D. Rus, “On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment,” *Proceedings of the National Academy of Sciences*, vol. 114, no. 3, pp. 462–467, 2017. 23, 79

[108] X. Qian, W. Zhang, S. V. Ukkusuri, and C. Yang, “Optimal assignment and incentive design in the taxi group ride problem,” *Transportation Research Part B: Methodological*, vol. 103, pp. 208–226, 2017. 23

[109] J. Wang, P. Cheng, L. Zheng, C. Feng, L. Chen, X. Lin, and Z. Wang, “Demand-aware route planning for shared mobility services,” *Proceedings of the VLDB Endowment*, vol. 13, no. 7, pp. 979–991, 2020. 23

[110] R. Kucharski and O. Cats, “Exact matching of attractive shared rides (exmas) for system-wide strategic evaluations,” *Transportation Research Part B: Methodological*, vol. 139, pp. 285–310, 2020. 23

[111] H. Yang, C. Shao, H. Wang, and J. Ye, “Integrated reward scheme and surge pricing in a ridesourcing market,” *Transportation Research Part B: Methodological*, vol. 134, pp. 126–142, 2020. 24

[112] J. James, A. Y. Lam, and Z. Lu, “Double auction-based pricing mechanism for autonomous vehicle public transportation system,” *IEEE Transactions on Intelligent Vehicles*, vol. 3, no. 2, pp. 151–162, 2018. 32

[113] R. Karamanis, E. Anastasiadis, P. Angeloudis, and M. Stettler, “Assignment and pricing of shared rides in ride-sourcing using combinatorial double auctions,” *IEEE Transactions on Intelligent Transportation Systems*, 2020. 32, 35

[114] S. M. Ross, *Introduction to Probability Models, ISE.* Academic press, 2006. 37, 38

[115] L. Kleinrock, “Queueing systems. volume i: theory,” 1975. 38

[116] B. Burgstaller and F. Pillichshammer, “The average distance between two points,” *Bulletin of the Australian Mathematical Society*, vol. 80, no. 3, pp. 353–359, 2009. 39

[117] N. Korolko, D. Woodard, C. Yan, and H. Zhu, “Dynamic pricing and matching in ride-hailing platforms,” *Available at SSRN*, 2018. 40, 64

[118] G. Boeing, “Osmnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks,” *Computers, Environment and Urban Systems*, vol. 65, pp. 126 – 139, 2017. 43, 78, 112

[119] TLC, “New York City Taxi Trip Data,” 2019. 43, 44, 78, 116

[120] San Francisco County Transportation Authority, “TNCs Today,” 2020. 43, 44

[121] N. Cleave, P. J. Brown, and C. D. Payne, “Evaluation of methods for ecological inference,” *Journal of the Royal Statistical Society. Series A (Statistics in Society)*, vol. 158, no. 1, pp. 55–72, 1995. 43

[122] R. Karamanis, P. Angeloudis, A. Sivakumar, and M. Stettler, “Dynamic Pricing in One-Sided Autonomous Ride-Sourcing Markets,” in *21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 3645–3650, 2018. 53, 79

[123] L. J. LeBlanc, E. K. Morlok, and W. P. Pierskalla, “An efficient approach to solving the road network equilibrium traffic assignment problem,” *Transportation research*, vol. 9, no. 5, pp. 309–318, 1975. 60

[124] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows Theory, Algorithms and Applications*, vol. 1. New Jersey: Prentice-Hall, 1993. 73, 74, 75, 76, 133

[125] M. Minoux, “A polynomial algorithm for minimum quadratic cost flow problems,” *European Journal of Operational Research*, vol. 18, no. 3, pp. 377–387, 1984. 74

[126] J. Edmonds and R. M. Karp, “Theoretical improvements in algorithmic efficiency for network flow problems,” *Journal of the ACM (JACM)*, vol. 19, no. 2, pp. 248–264, 1972. 74

[127] M. Minoux, “Solving integer minimum cost flows with separable convex cost objective polynomially,” in *Netflow at Pisa*, pp. 237–239, Springer, 1986. 74

[128] D. S. Hochbaum and J. G. Shanthikumar, “Convex separable optimization is not much harder than linear optimization,” *Journal of the ACM (JACM)*, vol. 37, no. 4, pp. 843–862, 1990. 74

[129] J. B. Orlin and B. Vaidyanathan, “Fast algorithms for convex cost flow problems on circles, lines, and trees,” *Networks*, vol. 62, no. 4, pp. 288–296, 2013. 74

[130] L. A. Vegh, “A Strongly Polynomial Algorithm for a Class of Minimum-Cost Flow Problems with Separable Convex Objectives,” *SIAM Journal on Computing*, vol. 45, no. 5, pp. 1729–1761, 2016. 74

[131] J. B. Orlin, “A faster strongly polynomial minimum cost flow algorithm,” *Operations research*, vol. 41, no. 2, pp. 338–350, 1993. 74

[132] DfT, “TAG Data Book,” tech. rep., DfT, London, UK, 2018. 79, 97, 112

[133] M. Ben-Akiva and M. Bierlaire, “Discrete Choice Methods and their Applications to Short Term Travel Decisions,” pp. 5–33, 1999. 85, 94

[134] Ordnance Survey, "OS Open Roads," 2017. 88

[135] Inrix, "London Congestion Trends," Tech. Rep. March, INRIX, 2016. 88

[136] TfL, "Rolling Origin & Destination Survey (RODS)," 2017. 88

[137] TfL, "Tube and rail fares," 2017. 88

[138] DfT, "WebTAG: TAG data book, March 2017," 2017. 88

[139] M. K. Chen and M. Sheldon, "Dynamic Pricing in a Labor Market: Surge Pricing and Flexible Work on the Uber Platform," *Proceedings of the 2016 ACM Conference on Economics and Computation*, p. 455, 2016. 89

[140] J. Bertrand, "Theorie Mathematique de la Richesse Sociale," *Journal des Savants*, pp. 499–508, 1883. 94, 96

[141] I. Griva, S. G. Nash, and A. Sofer, *Linear and nonlinear optimization*, vol. 108. Siam, 2009. 103

[142] C. H. Papadimitriou and M. Yannakakis, "Optimization, approximation, and complexity classes," *Journal of computer and system sciences*, vol. 43, no. 3, pp. 425–440, 1991. 105, 106

[143] A. Kako, T. Ono, T. Hirata, and M. M. Halldórsson, "Approximation algorithms for the weighted independent set problem," in *International Workshop on Graph-Theoretic Concepts in Computer Science*, pp. 341–350, Springer, 2005. 106, 107

[144] S. Butenko, *Maximum Independent Set and Related Problems, with Applications*. Phd diss., University of Florida, 2003. 107

[145] S. Homer and M. Peinado, "Experiments with polynomial-time clique approximation algorithms on very large graphs," *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, vol. 26, pp. 147–168, 1996. 107

[146] C.-R. Hwang, "Simulated annealing: theory and applications," *Acta Applicandae Mathematicae*, vol. 12, no. 1, pp. 108–111, 1988. 107

[147] M. M. Bykowsky, R. J. Cull, and J. O. Ledyard, "Mutually destructive bidding: The fcc auction design problem," *Journal of Regulatory Economics*, vol. 17, no. 3, pp. 205–228, 2000. 110

[148] D. Porter, S. Rassenti, A. Roopnarine, and V. Smith, "Combinatorial auction design," *Proceedings of the National Academy of Sciences*, vol. 100, no. 19, pp. 11153–11157, 2003. 110

[149] S. Ba, J. Stallaert, and A. B. Whinston, "Optimal investment in knowledge within a firm using a market mechanism," *Management Science*, vol. 47, no. 9, pp. 1203–1219, 2001. 110

[150] G. Iosifidis and I. Koutsopoulos, "Double auction mechanisms for resource allocation in autonomous networks," *IEEE Journal on Selected Areas in Communications*, vol. 28, no. 1, pp. 95–102, 2010. 110

[151] K. Xu, Y. Zhang, X. Shi, and H. Wang, "Online combinatorial double auction for mobile cloud computing markets," in *Performance Computing and Communications Conference (IPCCC)*, pp. 1–8, 2014. 110

[152] S. Bikhchandani, S. de Vries, J. Schummer, and R. V. Vohra, "Linear programming and vickrey auctions," *IMA Volumes in Mathematics and its Applications*, vol. 127, pp. 75–116, 2001. 110

[153] N. Nisan and A. Ronen, "Computationally feasible vcg mechanisms," *Journal of Artificial Intelligence Research*, vol. 29, pp. 19–47, 2007. 110

[154] R. B. Johnson, S. S. Oren, and A. J. Svoboda, "Equity and efficiency of unit commitment in competitive electricity markets," *Utilities Policy*, vol. 6, no. 1, pp. 9–19, 1997. 110

[155] B. Edelman and M. Ostrovsky, "Strategic bidder behavior in sponsored search auctions," *Decision Support Systems*, vol. 43, no. 1, pp. 192 – 198, 2007. Mobile Commerce: Strategies, Technologies, and Applications. 110

[156] P. Dütting, F. Fischer, and D. C. Parkes, "Expressiveness and robustness of first-price position auctions," *Mathematics of Operations Research*, vol. 44, no. 1, pp. 196–211, 2019. 110