New Formulations and Solution Methods for the Dial-a-ride Problem

Xiaotong Dong

A thesis in fulfilment of the requirements for the degree of

Doctor of Philosophy

Faculty: Engineering

School of Civil and Environmental Engineering

University of New South Wales

March 2022

🎓 Welcome to the Research Alumni Portal, Sharon Dong!

You will be able to download the finalised version of all thesis submissions that were processed in GRIS here.

Please ensure to include the **completed declaration** (from the Declarations tab), your **completed Inclusion of Publications Statement** (from the Inclusion of Publications Statement tab) in the final version of your thesis that you submit to the Library.

Information on how to submit the final copies of your thesis to the Library is available in the completion email sent to you by the GRS.

Thesis submission for the degree of Doctor of Philosophy

Thesis Title and Abstract	Declarations	Inclusion of Publications Statement	Corrected Thesis and Responses

ORIGINALITY STATEMENT

☑ I hereby declare that this submission is my own work and to the best of my knowledge it contains no materials previously published or written by another person, or substantial proportions of material which have been accepted for the award of any other degree or diploma at UNSW or any other educational institution, except where due acknowledgement is made in the thesis. Any contribution made to the research by others, with whom I have worked at UNSW or elsewhere, is explicitly acknowledged in the thesis. I also declare that the intellectual content of this thesis is the product of my own work, except to the extent that assistance from others in the project's design and conception or in style, presentation and linguistic expression is acknowledged.

COPYRIGHT STATEMENT

☑ I hereby grant the University of New South Wales or its agents a non-exclusive licence to archive and to make available (including to members of the public) my thesis or dissertation in whole or part in the University libraries in all forms of media, now or here after known. I acknowledge that I retain all intellectual property rights which subsist in my thesis or dissertation, such as copyright and patent rights, subject to applicable law. I also retain the right to use all or part of my thesis or dissertation in future works (such as articles or books).

For any substantial portions of copyright material used in this thesis, written permission for use has been obtained, or the copyright material is removed from the final public version of the thesis.

AUTHENTICITY STATEMENT

☑ I certify that the Library deposit digital copy is a direct equivalent of the final officially approved version of my thesis.

UNSW is supportive of candidates publishing their research results during their candidature as detailed in the UNSW Thesis Examination Procedure.

Publications can be used in the candidate's thesis in lieu of a Chapter provided:

- The candidate contributed **greater than 50%** of the content in the publication and are the "primary author", i.e. they were responsible primarily for the planning, execution and preparation of the work for publication.
- The candidate has obtained approval to include the publication in their thesis in lieu of a Chapter from their Supervisor and Postgraduate Coordinator.
- The publication is not subject to any obligations or contractual agreements with a third party that would constrain its inclusion in the thesis.

☑ The candidate has declared that **some of the work described in their thesis has been published and has been documented in the relevant Chapters with acknowledgement**.

A short statement on where this work appears in the thesis and how this work is acknowledged within chapter/s:

> Two publications are listed on page vii of the submitted thesis. Both of them were integrated into Chapter 4.

Candidate's Declaration

I declare that I have complied with the Thesis Examination Procedure.

Acknowledgement

I would like to dedicate this PhD thesis to my mum and dad, Kang Cuiwei and Dong Jianyou. Thank you for being the most supportive and caring parents that you are. I owe this great accomplishment to you and your believing in the power of education ever since I was a little kid. You have faith in me, always have my back and support all my decisions, even if that means I'd be living somewhere thousands of miles away from you and only get to see you once a year. You made me this strong and independent woman I am today by telling me to always chase my dreams and never give up without a fight. I love you!

I would also like to take this opportunity to express my gratitude to my PhD supervisors, Professor S Travis Waller and Dr David Rey. Travis, thank you for keeping me in check and making sure that I can always see the bigger picture rather than fixating on what's in my sight. David, thank you for always being there for me. I still believe that I am one of the luckiest PhD candidates to have had you as my supervisor. Your passion, talent and integrity inspire me every day. The conversations we had over the past few years helped me navigate through difficult times when I was having doubts on my career and my life. Thank you for your positivity and unique sense of humour that made my PhD journey so enjoyable. I truly enjoyed working with you and have learned so much from you. I'd also like to say thank you to my rCITI family - Shantanu, Fernando, Malith, Siroos, Amolika, Sai, Neeraj and Divya. Thanks to all of you, I felt so welcomed and felt like a part of the family straight away since the first day I joined rCITI. I'm so grateful to be a part of this loving family and for all the time we spent together inside and outside the office - coffee runs, movie nights, bush walks, snow trips... you guys made my PhD journey so much fun.

Doing a PhD through the COVID-19 pandemic was not easy, especially when international travels were banned for almost two years. Things do get tough when you are forced to be away from family for too long. I want to say a special thank-you to Ted's mum and dad, Bich and Clive. Thank you for welcoming me into your family with open arms and showering me with love and care. Thank you for taking such good care of me and for everything you have done for us. And Ted, words can't express how grateful I am to have you by my side throughout my PhD, as well as in my life. Thank you for always being my rock, I certainly could not have done it without you.

Last but not least, a special shout-out to Yu, Maggie, Ellison, Daniel, Yao, Dian and all my friends in Australia and China. Your friendships mean the world to me in this stressful time. Thanks for being there to celebrate every little achievement in life with me, listening to my rants and whinges, talking sense into me when I got carried away and being there for me whenever I get too stressed out and need a drink (or ten ;)). I love you beautiful people!!!

Abstract

The classic Dial-A-Ride Problem (DARP) aims at designing the minimum-cost routing solution that accommodates a set of user requests under constraints at the operations planning level. It is a highly constrained combinatorial optimization problem initially designed for providing door-to-door transportation for people with limited mobility (e.g. the elderly or disabled). It consists of routing and scheduling a fleet of capacitated vehicles to service a set of requests with specified pickup and drop-off locations and time windows. With the details of requests obtained either beforehand (static DARP) or en-route (dynamic DARP), dial-a-ride operators strive to deliver efficient and yet high-quality transport services that satisfy each passenger's individual travel needs.

The goal of this thesis is threefold: (1) to propose rich DARP formulations where users' preferences are taken into account, in order to improve service quality of Demand-Responsive Transport (DRT) services and promote ridership strategically; (2) to develop novel and efficient solution methods where local search, column generation, metaheuristics and machine learning techniques are integrated to solve large-scale DARPs; and (3) to conduct real-life DARP case studies (using data extracted from NYC Yellow Taxi trip records) to test the practicality of proposed models and solution methods, as well as to emphasise the importance of connecting algorithms with real-world datasets. These aims are achieved and presented in the three core chapters of this thesis. In the first core chapter (Chapter 3), two Mixed Integer Programming (MIP) formulations (link-based and path-based) of DARP are presented, alongside with their objective functions and standard solution methods. This chapter builds the foundation of the thesis by elaborating the base models

and algorithms that this thesis is based on, and by running benchmark experiments and reporting numerical results as the base line of the whole thesis. In the second core chapter (Chapter 4), two DARP models (one deterministic, one stochastic) integrated with users' preferences from dial-a-ride service operators' perspective are proposed, facilitating them to optimise their overall profit while maintaining service quality. In these models, users' utility users' preferences are considered within a dial-a-ride problem. A customized local search based heuristic and a matheuristic are developed to solve the proposed Chance-Constrained DARP (CC-DARP). Numerical results are reported for both DARP benchmark instances and a realistic case study based on New York City yellow taxi trip data. This chapter also explores the design of revenue/fleet management and pricing differentiation. The proposed chance-constrained DARP formulation provides a new decision-support tool to inform on revenue and fleet management, including fleet sizing, for DRT systems at a strategic planning level. In the last core chapter (Chapter 5), three hybrid metaheuristic algorithms integrated with Reinforcement Learning (RL) techniques are proposed and implemented, aiming to increase the scale-up capability of existing DARP solution methods. Machine learning techniques and/or a branching scheme are incorporated with various metaheuristic algorithms including VNS and LNS, providing innovative methodologies to solve large-instance DARPs in a more efficient manner. Thompson Sampling (TS) is applied to model dual values of requests under a column generation setting to negate the effect of dual oscillation (i.e. promote faster converging). The performance of proposed algorithms is tested benchmark datasets, and strengths and weaknesses across different algorithms are reported.

Overall, this thesis explores new formulations and solution algorithms of classical and rich Dial-a-Ride Problems. Results of this work provide not only insightful modelling perspectives that facilitate DRT operators strategically, but also efficient integrated solution algorithms to solve NP-hard optimization problems.

List of Publications

Journal Articles (Published)

- Dong, X., Rey, D. and Waller, S. T., 2020. Dial-a-ride problem with users' accept/reject decisions based on service utilities. *Transportation Research Record*, 2674(10), pp.55-67. (Dong et al., 2020)

- Dong, X., Chow, J.Y., Waller, S. T. and Rey, D., 2022. A chance-constrained dial-a-ride problem with utility-maximising demand and multiple pricing structures. *Transportation Research Part E: Logistics and Transportation Review*, 158, p.102601. (Dong et al., 2022)

Journal Article Under Preparation

- Dong, X., Rey, D., Monteil, J. and Waller, S. T., 2022. Scaling Up DARP with Thompson Sampling.

Conference Presentations

- Dong, X., Rey, D., and Waller, S. T., 2019. Integrating Dial-a-Ride with Mode Choice, Presented at the *7th Workshop of the EURO Working Group on Vehicle Routing and Logistics optimization (VeRoLog)*, Seville, June 2019.

- Dong, X., Rey, D., and Waller, S. T., 2019. Stable dial-a-ride formulation with users' preferences, Presented at *at 2nd TRANSW Symposium*, Sydney, December 2019.

- Dong, X., Rey, D. and Waller, S.T., 2020. Dial-a-ride problem with users' accept/reject decisions based on service utilities. Presented at *99th Transportation Research Board Annual Meeting*, Washington D. C., January 2020.

Contents

Acknowledgement **iv**

Abstract **vi**

List of Publications **viii**

Contents **x**

Abbreviation Table **xvi**

List of Figures **xvi**

List of Tables **xviii**

1 Introduction **1**

1.1 Background 1

1.2 Problem Statement 4

1.3 Research Objectives 4

1.4 Thesis Overview 5

2 Literature Review **9**

2.1 Introduction 9

2.2 DARP and Its Variants 11

2.3 Choice Modeling in Mobility Systems 13
2.4 Solution Methods 15
2.5 Applicable Machine Learning Techniques 17
2.6 Summary 20

3 Classic DARP Formulations **23**
3.1 Introduction 23
3.2 MIP Formulations for the DARP 24
3.2.1 Link-based Formulations 24
3.2.2 Set Partitioning Formulation 29
3.3 Objective Functions 30
3.4 Main Solution Methods 31
3.4.1 Column Generation and Branch-and-Price Algorithm 31
3.4.2 Classic Insertion Heuristic 32
3.4.3 Local Search 33
3.4.4 Metaheuristics Based on Local Seach: Variable Neighbourhood Search (VNS) and Large Neighbourhood Search (LNS) 34
3.5 Dataset Description 36
3.5.1 Cordeau (2006)'s Dataset 37
3.5.2 NYC Taxi Dataset 38
3.6 CPLEX Implementation 39
3.6.1 Pre-processing Techniques 39
3.6.2 Implementation Results 40

4 Integrating Users' Utility in DARP **45**
4.1 Introduction and Notations 45
4.2 Deterministic Model: DARP-UP 47
4.2.1 Mathematical Formulation 47
4.2.2 Numerical Results 52

4.3 Stochastic Model: Chance-constrained DARP 61
4.3.1 Utility Functions and Chance Constraints 62
4.3.2 Class-based Fare Structures 65
4.3.3 Chance-constrained DARP (CC-DARP) Formulation 67
4.4 Local-Seach Based Solution Method 69
4.4.1 Construction of an Initial Solution 69
4.4.2 Local Search 73
4.4.3 LS-H Algorithm Summary 76
4.5 Matheuristic Solution Method 78
4.6 Numerical Results 80
4.6.1 Comparison between Classic DARP and CC-DARP 80
4.6.2 LS-H and MATH-H Performance Benchmarking 81
4.6.3 Experiments on NYC Data 87
4.7 Conclusion 101

5 Hybrid Metaheuristics and Reinforcement Learning Approaches for the DARP 105
5.1 Introduction 105
5.1.1 Hybrid Metaheuristics and Column Generation Approaches 108
5.1.2 Thompson Sampling 109
5.1.3 Chapter Outline 111
5.2 Branch-and-Price with Hybrid Metaheuristics 112
5.3 Column Generation with Hybrid Metaheuristics and Learning 121
5.4 Branch-and-Price with Hybrid Metaheuristics and Learning 126
5.5 Numerical Experiments 128
5.5.1 Algorithms Summary 128
5.5.2 Parameters Tuning 129
5.5.3 Numerical Experiments 133
5.6 Conclusion 137

6 Conclusion **139**

6.1 Summary 140

6.2 Limitations and Future Work 142

6.3 Final Remarks 143

Abbreviation	Definition
DARP	Dial-a-ride Problem
DRT	Demand-Responsive Transport
VoT	Value of Time
MILP	Mixed-Integer-Linear-Programming
DARP-UP	Dial-a-ride Problem with Users' Preferences
CC-DARP	Chance-Constrained DARP
VRP	Vehicle Routing Problem
VRPTW	Vehicle Routing Problem with Time Windows
PDPTW	Pick up and Delivery Problem with Time Windows
CG	Column Generation
ML	Machine Learning
RL	Reinforcement Learning
RNN	Recurrent Neural Network
TS	Thompson Sampling
LNS	Large Neighbourhood Search
VNS	Variable Neighbourhood Search
B&P	Branch&Price
LS-H	Local Search based Heuristic
MATH-H	MATH-Heuristic
HYB-TS	Hybrid heuristic with Thompson Sampling
B&P-HYB	Branch&Price integrated Hybrid heuristic
B&P-HYB-TS	Branch&Price integrated Hybrid heuristic with Thompson Sampling

Table 1: Abbreviation Table

List of Figures

3.1 Branch-and-Price Solution Method Overlook 42

3.2 4 Cases in Jaw's insertion Heuristic (Jaw et al., 1986) 43

3.3 NYC Dataset (143 requests, 286 nodes) 44

4.1 The average and standard deviation of critical parameters 55

4.2 The average and standard deviation of critical parameters 56

4.3 The average and standard deviation of critical parameters 57

4.4 The average and standard deviation of critical parameters 58

4.5 The average and standard deviation of critical parameters 60

4.7 MATH-H Solution Method Overlook 79

4.9 Results from the NYC instance under a flat fare structure $f_{im} = f$ for all requests $i \in \mathcal{P}$ and user class $m \in \mathcal{M}$. 90

4.10 Results from the NYC instance when changing p_{M1} and p_{M2} under a flat fare structure. 91

4.11 Results from the NYC instance when changing *Capacity* under a flat fare structure. 92

4.12 Results from the NYC instance when changing α_{M1} and α_{M2} under a distance-based fare structure. 94

4.13 Results from the NYC instance when changing p_{M1} and p_{M2} under a distance-based fare structure. 95

4.14 Results from the NYC instance when changing the vehicle capacity Q under a distance-based fare structure. 96
4.15 Results from the NYC instance when changing f_{m1} and f_{m2} under a zone-based fare structure. 98
4.16 Results from the NYC instance when changing p_{m1} and p_{m2} under a zone-based fare structure. 99
4.17 Results from the NYC instance when changing *Capacity* under a zone-based fare structure. 100
4.6 LS-H Solution Method Overlook 102
4.8 Profit trends for CC-DARP and Classic DARP formulations 103

5.1 Column Generation Outline 107
5.2 B&P-HYB Solution Method Overlook 113
5.3 Branch-and-Price TS Solution Method Overlook (B&P-HYB-TS) 127

List of Tables

1 Abbreviation Table xv

3.1 Notations in the Classic 3-index DARP formulation 25

3.2 CPLEX Benchmark Implementation 41

4.1 Notations 46

4.2 Optimal Profit and Computational Time for All Instances 53

4.3 Comparison Between Classic DARP and DARP-UP for Instance a2-16 54

4.4 Optimal Profit and Computational Time for the First 4 Instances 56

4.5 Sensitivity Analysis on p ($s = 10, f = 20$) 83

4.6 Sensitivity Analysis on s ($p = 0.95, f = 20$) 84

4.7 Sensitivity Analysis on f ($p = 0.95, s = 1$) 85

4.8 Comparison between LS-H and MATH-H 86

4.9 Algorithms comparison with the NYC Dataset instance using a flat fare 88

5.1 Algorithms Summary 128

5.2 N^{new} (VNS) and N^{int} Tuning 130

5.3 N^{new} (VNS) and N^{int} Tuning 131

5.4 HYB and HYB-TS Performance 133

5.5 B&P-HYB and B&P-HYB-TS Performance 134

5.6 HYB And HYB-TS Performance - Time Split (%) 135

5.7 B&P-HYB And B&P-HYB-TS Performance - Time Split (%) 135

Chapter 1

Introduction

1.1 Background

The world of transportation and mobility is facing emerging challenges due to continuous population growth and global urbanisation. With the ever growing size of cities and towns with limited parking spaces, residents and commuters are in need of safe and reliable modes of public transportation to fulfill their travel needs of various purposes (e.g., work, education, leisure, etc.). At the same time, concerns are being raised worldwide regarding increasing carbon emissions caused by fossil fuel vehicles. Greenhouse gas emissions created by vehicles, especially private vehicles, is one of the major causes of global warming and climate change. Traffic congestion on urban networks is another major issue resulted from the increasing volume of traffic on the road, causing significant economic loss. The increasing demand and escalating challenges call for new mobility solutions that integrate the latest developments in technology, renewable energy and management to adapt to the fast-paced modern world.

Demand-Responsive Transport (DRT) is widely considered to be the future of public transport. Originally designed for people with limited mobility, DRT delivers flexible and user-oriented services to their user group, dynamically allocating resources to demand in a temporally and/or

spatially efficient manner. DRT services encourage passengers to plan ahead before travelling and book a ride using phone calls, websites or mobile apps, where they inform the service operator their intended travel date, time, pick-up and drop-off locations. A customised trip will then be scheduled and deliver tailored to meet the passenger's requirements. DRT also offers significant social and environmental benefits, including connecting passengers and communities to major public transport hubs/networks by providing first/last mile transportation services, reducing greenhouse gas emissions, alleviating traffic congestion in metropolitan areas, etc. Facing global challenges in population growth and escalated environmental concerns, DRT services might be the new mobility solution that countries and organisations are searching for.

Comparing to the fixed routes and timetables of traditional transit services, the flexibility and convenience offered by DRT are favoured by many. Behind the scenes, operators optimising DRT services are essentially solving a combinatorial optimisation problem called the Dial-a-Ride Problem (DARP). Advancing the studies on DARPs can in return benefit the DRT operations and their service quality, making this mode of transport more appealing to passengers. DARPs consist of designing minimal-cost routing and scheduling solutions for a fleet that satisfies the demand and service quality requirements of a group of passengers. From a mathematical modelling perspective, the DARP is often seen as a generalisation of the Vehicle Routing Problem (VRP) family (Cordeau and Laporte, 2003a). The closest "relative" of the DARP would be the Pick-up and Delivery Problem with Time Windows (PDPTW), where a fleet of vehicles are required to transport goods from their origins to their destinations, with respect to time window and capacity constraints. The difference between the DARP and the PDPTW lies in the fact that in DARPs, the vehicles are transporting people rather than products. This difference introduces the dilemma faced by all DRT operators: how can operational costs be minimised without sacrificing service quality for passengers? Quality of service can be difficult to define since the measurable and tangible elements only represent a small part of the service, whereas most elements are intangible and heterogeneous. For DRT operators, service improvements they need to make depend largely on the characteristics and particularities of the targeted area and individuals (Redman et al., 2013). For a standard DARP, the goal is to

service all requests (if rejecting requests is not allowed) while minimising the operational cost of the fleet. Usually, passengers also have requirements on service quality that DRT operators need to satisfy. For example, a passenger could have a maximum ride duration and a time window on departure/arrival time that he/she is willing to accept. The final solution proposed by the operator must satisfy all the service quality related requirements of passengers. These requirements can be translated to constraints in the modelling of the DARP. Other constraints that are shared by DARPs and PDPTWs include: (a) All trips must start and end from a depot; (b) The load of a vehicle must not exceed its capacity at any time; (c) The duration of any tour must not be longer than a certain threshold.

Besides, DRT services of different features can be offered by different operators, based on their fleet, management capability and the demand of their target customers. For example, DRT services may operate on two modes: static or dynamic. Static DRT services only accept requests that are received beforehand, and no new request can be added once the routing and scheduling process is finalised. On the contrary, dynamic DRT services allows real-time adjustments in routes to pick-up passengers en route. Another example is that DRT operators could manage fleets of either identical vehicles, or vehicles of various capacities. All these features of DRT services above (static/dynamic mode, homogeneous/heterogeneous fleet) can be reflected in the modelling of DARPs.

Although promising in a wide range of applications, the DARP is notoriously hard to solve. A number of exact and approximate solution methods have been developed by researchers to solve DARPs over the past decades. Most exact algorithms, striving to solve DARPs to optimality, struggle to scale up, given the NP-hardness of the problem (Baugh Jr et al., 1998). Therefore, approximate/heuristic solution methods are more often developed and implemented to solve large-size DARPs and/or real-life DARP applications with customised characteristics.

1.2 Problem Statement

With the development of the sharing economy and DRT services around the world, research in DARPs has attracted considerable attention over the years. The DARP is rich in variations, each of them stemming from different features in real-life applications (Ho et al., 2018). The research trend in DARPs is to incorporate features of real-life (operational) DRT applications to further enhance the practical value of DARP models and their solution algorithms. DARP models should strive to take into account various characteristics to mimic and/or simulate not only the operational process of the DRT system, but also the complex interactions between operators and customers. Models of this nature are invaluable from an operator's point of view, because it enables the operators to plan and maintain their service strategically. Another main challenge lies in solving a highly constrained model in an effective manner, given that each problem-specific feature translate to additional constraints from a modelling point of view. Further research is needed for better understanding the effect of users' preferences and the interaction between customers and DRT systems from an operator's strategic point of view. Furthermore, innovative solution methods are needed for the DARP and its variants, since solving large-scale real-life problems effectively remains challenging.

1.3 Research Objectives

The research objectives for this thesis is threefold:

- To propose modelling frameworks of DARPs that integrates users' preferences into the DARP to broaden the existing DARP formulations, which can further provide management guidance in real-life DRT operations.

- To develop algorithms that solve standard and rich DARPs; To integrate applicable Machine Learning techniques into optimisation algorithms to scale up the computational efficiency on

solving large DARP instances .

- To conduct DARP case studies in a realistic setting; To generate realistic dataset and explore the practical applications of the proposed (rich) DARP formulations and solution algorithms.

1.4 Thesis Overview

The three research objectives of this thesis is fulfilled in the following chapters:

Chapter 2. Literature Review

This chapter presents a thorough review of studies relevant to the problem statement and research objectives of this thesis.

Chapter 3. Classic DARP Formulations

This chapter presents the base Mixed Integer Linear Programming (MILP) formulations of the DARP, on which extended DARP models in following chapters are built on. The link-based formulation of the DARP focuses more on modelling of problem-specific constraints explicitly, whereas the path-based set partitioning formulation is more solution method oriented. Standard MILP formulations of the DARP (and some basic variations) are presented in this chapter, as well as applicable pre-processing techniques. Standard form of basic algorithms that are often used for solving the DARP, including Column Generation, Local Search, and some Metaheuristic algorithms, are also presented in this chapter, which act as the backbone of the sophisticated algorithms developed in the following chapters. Furthermore, data cleaning and processing of the datasets used in this thesis and implementations of Classic DARP using commercial solver CPLEX are documented. In short, this chapter serves as the foundation of the thesis, providing references in mathematical notations, standard modelling, mathematical formulations, solution methods and benchmark testing datasets

for the rest of the thesis.

Chapter 4. Integrating Users' Utility in DARP

In this chapter, two extended DARP models are introduced, capturing users' preferences in a DRT context from a strategic point of view. Users' utilities for alternative transportation modes are taken into account, including driving and a Demand-Responsive Transportation (DRT) service. This chapter considers utility-maximizing users and proposes mixed-integer programming formulations for deterministic and stochastic DARP models. In the stochastic DARP model, Chance-constrained modelling is applied to capture users' preferences in the long run via a Logit model. Furthermore, a multi-class DARP formulation is proposed, which enables the models to explore the behaviours of various demographics under different price structures. Models in this chapter provides insights to DRT operators on pricing policies design as well as revenue management strategies. Two customized heuristic solution methods with layered local search structure are developed to optimise both routing decisions and user selections. Numerical results showed that the customized local search based heuristic algorithm (LS-H) and matheuristic (MATH-H) performed well on DARP benchmarking instances of the literature when compared to the best integer solution returned by an exact MILP solution. The proposed solution methods are further implemented on a realistic case study derived from yellow taxi trip data from New York City (NYC). The results obtained on the realistic case study reveal that a zonal fare structure is a profit-maximizing strategy for DRT operators.

Chapter 5. Hybrid Metaheuristics and Reinforcement Learning Approaches for the DARP

In this chapter, three hybrid metaheuristic algorithms based on a Column Generation (CG) structure are developed and implemented to solve classic DARP, aiming to scale up the computational efficiency when solving large-size problems. Hybrid structures integrating elements including

Machine Learning, Branch-and-Bound and Metaheuristics are proposed, and their performances are tested with the same set of benchmark instances. Specifically, a Reinforcement Learning (RL) technique called Thompson Sampling is applied to CG structured solution methods in order to model the behaviour of dual values associated to the constraints of the Master Problem (MP). The incorporation of this machine learning technique enables the Sub-Problem (pricing) to be solved intelligently, and therefore accelerates the improvement in solution quality over iterations. Furthermore, an additional branching scheme is implemented to some of the CG structured algorithms, providing an alternative way of generating integer solutions. Numerical results in this chapter compared the performances across all proposed algorithms, and the results reveals that integrating Thompson Sampling as a part of the CG hybrid solution method is non-trivial and can significantly improve the computational efficiency of the algorithms on large instances. Overall, the integration of Thompson Sampling speeds up the algorithm by 31.56% on average. When running under the same time limit (1 hour) on large instances, algorithms with TS integrated found better solutions overall by 7.25% better in objective values (at best 13.4% better).

Chapter 6. Conclusion

This chapter concludes the thesis with summarising the novelty and contribution of the work and proposing potential future research directions.

Chapter 2

Literature Review

2.1 Introduction

Shared mobility system problems have been thoroughly studied, and different facets of the problems have been explored in literature in terms of service design. *Strategy design* looks at the problem from a rather long-term point of view, addressing the organizational structure from the perspective of a vehicle sharing program (VSP) operator. Nair and Miller-Hooks (2014) suggest that the nature of optimal designs are the trade-offs between the objectives of VSP operators and users, and to provide managerial insights that feed into public policies for urban communities wanting to implement such systems. They propose a mixed-integer bi-level program where the different objectives of the VSP operator and users are considered. The upper-level problem, from the perspective of the VSP operator, is to determine the optimal VSP configuration such that the revenue resulting from shared-vehicle flows is maximized, considering the limitations such as budgets, policies, number of stations, etc. The lower level problem, from the perspective of users, is to determine the flows on the network that maximize travel utility.

Users travelling between various OD pairs minimize their travel and waiting time. Xu et al. (2015) propose a Ride-sharing User Equilibrium (RUE) model, showing how traffic congestion and

people's participation in ride sharing interact with each other in a mathematical way. When an OD flow is at RUE, the generalized costs of all used paths joining the same OD pair must be equal and not greater than the generalized costs of the unused paths joining the same OD pair, which means no one can improve their generalized travel cost by unilaterally switching routes or mode. Di et al. (2018) reform RUE into link-node presentation to reduce computational load and problem size, focusing on whether the existing roads should be retrofit into HOT lanes. Since user reaction needs to be taken into consideration, bi-level programming is widely used in strategy design in transportation planning in general, in which public sectors at the upper level make decisions seeking the best output of the network as a whole, while the users at the lower level make choices based on some traffic assignment principle (Migdalas, 1995).

Operational design, on the other hand, is more microscopic and focuses on short-term, trajectory-level control. Vehicle routing problem (VRP) concerns designing a minimized cost routing for a fleet of vehicles that depart from the same depot, so that they visit a set of customers with known demand once (Dumas et al., 1991). The generalizations of VRP include pickup and delivery problem with time windows (PDPTW) and vehicle routing problem with time windows (VRPTW). In VRPTW, each request is constrained with an earliest and latest service time, whereas in terms of PDPTW, all requests contain a pickup node and a delivery node that need to be served under capacity and caring constraints. The former one appears to be a particular case of the latter in that all requests share the same destination depot. As a special case of PDPTW, the Dial-a-Ride Problem (DARP) looks into transportation of people instead of goods. Hence, requirements related to service quality must be satisfied, either through additional constraints of the model or with extra terms in the objective function. Note that VRPs and their generalisations are all combinatorial optimisation problems, for which determining optimal solutions is NP-hard.

2.2 DARP and Its Variants

Dial-a-ride services originally stemmed from providing door-to-door transportation services for people with limited mobility (e.g., the elderly and the disabled people). The factor that makes DARPs challenging is the trade-off between operational cost and user inconvenience (Cordeau and Laporte, 2003b) (Cordeau, 2006). The standard DARP and its mathematical formulations was first proposed by Cordeau and Laporte (2003b), which has been studied extensively and extended broadly for practical real-life applications. The aim of the problem is to design minimal cost routes for a fleet of vehicles with a set of constraints such as time windows, duration, load, pairing, preceding, etc., provided that all demands are met. Ropke et al. (2007) present a 2-index formulation without index k, given that the route duration is not bounded and the travel cost is not vehicle-dependent.

The extensions of the standard DARP can be categorized into three types (Molenbruch et al., 2017). The first type involves advanced service design due to users' heterogeneous needs (Parragh, 2011), transfers between vehicles and different transport modes and further specifications of service quality (Jaw et al., 1986). The second type concerns the objective functions in the optimization model which can be extended either as single objective such as minimize fleet size (Diana et al., 2006), maximize occupancy rate (Garaix et al., 2011) or maximize revenue (Parragh et al., 2014), or as multiple objectives that is usually a combination of costs and quality-related objectives (user's inconvenience, etc). The last type of extension relates to whether the requests information available to the service provider is static or dynamic. In the first case, all the requests are known beforehand and the requests that are made during the execution of routing are not considered, while in the second case, new requests are automatically taken into consideration and the routes and schedules are adjusted accordingly to accommodate the new demand (Psaraftis, 1980). However, pure dynamic cases rarely exist in practise since a certain amount of requests are often known in advance (Cordeau, 2006). The most frequently used objective function in the DARP is to minimize the operational cost, subject to full demand satisfaction and other service design constraints (Molenbruch et al., 2017). However, it is sometimes unrealistic to satisfy all demand, as the operational cost

would be too large, and service level would drop significantly because of the detours required to serve all user requests.

Alternatively, a variant of TSP assigns profit values to links and/or nodes in the network but relaxes the constraints of visiting all nodes. The Team Orienting Problem (TOP), which can be formulated as a TSP with Profit, maximises profits collected by traversing through locations within a fixed amount of time. In TOPs, all locations in the network are associated with scores (Chao et al., 1996). Minimising travelling/routing cost is posed as a constraint, as opposed to as the objective functions in classic VRPs. Tang and Miller-Hooks (2005)'s work considers Stochastic Selective TSP (SSTSP), where the system maximises profit with stochastic travel time on each arc. In Feillet et al. (2005b)'s study, the two conflicting objectives of TSPs with profits, namely to collect more profit and to minimise travel cost, are formulated either as a weighted sum, or one of the two objectives is constrained with a bound value. In the first case, the problem is also identified as the Profitable Tour Problem (PTP) (Dell'Amico et al., 1995), which occurs more often as a subproblem in a column generation based solution algorithm in various routing problems. The problems of the second case are also known as the Orienteering Problem (OP) or the Prize-Collecting TSP (PCTSP, also known as quota TSP), depending on which one of the objectives is stated as a constraint. Archetti et al. (2009) explored two versions of capacitated VRPs with profits: an extension of the capacitated OP with multiple tours (Capacitated Team Orienteering Problem, shortened to CTOP), as well as the capacitated PTP (CPTP) with a single vehicle. A column generation based exact approach (branch-and-price) and various metaheuristics are implemented to solve the problems. A branch-and-cut algorithm for CPTP is later developed by (Jepsen et al., 2014) with a number of valid inequalities introduced. (Chow and Liu, 2012) introduced a new class of problems, named the generalised PTPs (GPTPs), in order to cater to the need of location-based activity routing systems. In GPTPs, nodes are categorised into different clusters for different activity types. The objective is to maximize the summed utility of all selected activities and minimize the disutility of travelling, with a set of constraints including time windows and fuel constraints, etc. In a separate paper, Feillet et al. (2005a) also introduced a similar routing problem called the profitable arc tour

problem (PATP), where the profit is associated with arcs, instead of vertices.

Although profit-driven TSPs and VRPs have been studied by many, DARPs with profit has hardly ever been explored. Schönberger et al. (2003) first link profit and the Pickup and Delivery Problem with Time Windows (PDPTW) by proposing the Pickup and Delivery Selection Problem (PDSP), which is an extension of the PDPTW with an accept decision of the requests. With a revenue associated to each request, service providers aim to maximise the total profit by identifying a selection of requests to service. Parragh et al. (2015) introduced the concept of the DARP with Split Requests and Profits (DARPSRP). A path-based formulation of the DARPSRP is proposed, where a revenue is associated to each request and the objective is to maximise the total profit. Unlike classic DARPs, requests can be rejected by Demand Responsive Transportation (DRT) operators if they are not profitable. A branch-and-price algorithm for the DARPSRP is developed alongside a Variable Neighbourhood Search (VNS) metaheuristic method. To the best of the author's knowledge, the existing studies on Dynamic DARP request accept/reject decisions are simply made simultaneously as routing is being optimised. No exploration has been done aiming to optimise the selection of requests as an independent procedure on account of users' utility and preferences, mainly because it can be tricky to observe and determine utility in a dynamic setting. However, in this case, utility theory can be well applied in capturing and modelling passengers' preferences and facilitate the operator's decision of selecting users (especially from a strategic planning and revenue management point of view).

2.3 Choice Modeling in Mobility Systems

Service attributes including in-vehicle travel time, waiting time, price, safety and reliability influence travellers when deciding whether to take a trip by public transport. When modelling and analysing mode choice decisions, multinomial logit models have been widely applied for estimating mode split (Train, 2009). Utility functions are estimated in logit models to quantify users' satisfaction level in

each travel mode. Basic utility functions of public transport travel mode take into consideration of travel time, waiting time, price and headway between two consecutive services. Follow-up studies introduce additional characteristics that might also affect users' choices including density of standees per square meter, availability of information panels, proportion of seats been used, etc (Tirachini et al., 2014) (Dell'Olio et al., 2011). Parameters of utility functions, including value of travel time (VOT), value of waiting time and schedule delay and their correlation can vary with different time of the day, demographic characteristics of an area, etc (Athira et al., 2016; Lisco, 1968; Mohring et al., 1987; Small, 1982).

Transit pricing research has received a wide attention for decades, with objectives such as generating more revenue and encouraging users to switch from private transport modes to public transportation. Cervero (1990) examined how the changes in pricing policy, including fare levels, fare structures and forms of payment, affect ridership and revenue. Ridership response studies are often broken down into segments by user groups and trip characteristics. Users from different segments of age, income, auto access, trip purpose and trip lengths tend to respond with various sensitivities to fare changes. Furthermore, the operating environment such as land use (density, composition, etc) and location (CBD, suburb, etc) also play a role on fare elasticities. For example, Paulley et al. (2006) demonstrate in their work that people in areas with low population densities rely more on cars and less on public transport, and therefore are more likely to switch to private travel when fares increase. Fare elasticity values of off-peak time periods are about twice the peak values. Also, travellers with higher incomes tend to have higher elasticity. Another aspect of public transport pricing scheme is fare structure. As a component within public transport systems, fare structure refers to the spatial structure reflecting the relationship between fare levels and distance travelled (Streeting and Charles, 2006). There are four commonly used fare structures in public transport system: flat fare, distance-based fare, time-based fare and zone-based fare (Chin et al., 2016). A well-suited fare structure provides a balance of efficiency, ridership and equity benefits. Reforming fare structure can be a less costly way of boosting ridership in the short term (Liu et al., 2019).

Fares that maximize the overall social welfare can be rather low and might require a subsidy that

covers the operating cost (Tirachini et al., 2014), due to the need of easing congestion, promoting social equity and environmental considerations. A low fare can attract users of other travel modes to switch to public transport, but at the cost of increasing travel time of all users, more stopping delay and more discomfort due to overcrowding services. Cervero (1990) point out that transit users are approximately twice more sensitive in service quality changes than in fare modifications, which implies that public transport services would be able to compete with private travel mode better when service quality is improved significantly, rather than subsidizing fares. On the other hand, from an economic perspective, optimal pricing policies that maximize social welfare as a whole vary based on the economic logic behind (first-best pricing, second best pricing, etc) (Tirachini and Hensher, 2012; Gentile et al., 2005).

However, quality of service can be difficult to define since the measurable and tangible elements only represent a small part of the service, whereas most elements are intangible and heterogeneous. Knowing exactly what improvements need to be made depend largely on the characteristics and particularities of the targeted area and individuals (Redman et al., 2013). Customer based quality is firstly mentioned by Falcocchio (1979) with three main dimensions: convenience, comfort and safety. This list was further extended to eight dimensions with 41 attributes. According to the survey conducted by Gunay et al. (2016), the reasons why DRT services are not favoured among travellers are chiefly proximity (25%) and fare (23%). The modal shift from automobiles to DRT services is estimated to be 52% once an intelligent transport service is introduced which provides flexible and reliable services.

2.4 Solution Methods

Although studies over the years have developed a series of exact and approximate solution method applied to the DARP, solving the DARP and its variants remains challenging, given that it is NP-hard. For exact solution methods, Cordeau (2006) proposed a branch-and-cut algorithm to solve the arc-based MILP formulation of DARPs. Pre-processing techniques including time window tightening and infeasible arc eliminations are applied, and valid inequalities regarding to the constraints

are added to further strengthen the algorithm. Other studies continued adding valid inequalities to improve the algorithm: Ropke et al. (2007) added valid inequalities on reachability constraints and fork constraints to the formulation, while Parragh (2011) further developed capacity inequalities to introduce heterogeneous users and vehicles into models. The column generation technique, embedded in a branch-and-bound scheme as a branch-and-price method, is also widely applied to solve larger instances. Starting with a small pool of columns (paths), the branch-and-price method solves the linear relaxation of the path-based master problem, obtains dual values and adds paths with negative reduced costs dynamically by pricing (Parragh et al., 2012) (Garaix et al., 2011).

Approximate solution methods, on the other hand, are favoured by many researchers since they can generate close to optimal solutions with a cheaper computational cost. Most heuristics for the DARP are based on Jaw et al. (1986)'s time-sequential insertion heuristic (Diana and Dessouky, 2004) (Toth and Vigo, 1997). To better escape local optima, more advanced (local-search based) heuristics including metaheuristics (e.g. Tabu Search, VNS, LNS) and matheuristics for DARP have also been studied by many. Cordeau and Laporte (2003b) proposed a Tabu Search framework for DARP that allows intermediate infeasible solutions and deterioration, which allows the algorithm to jump out of local optima. Calvo and Touati-Moungla (2011) and Kirchler and Calvo (2013) proposed granular Tabu Search, where a granular neighbourhood is considered based on the temporal and spatial relationship among requests in order to reduce the size of the neighbourhood. Parragh et al. (2010) introduced a Variable Neighbourhood Search (VNS) framework for the DARP, where the neighbourhoods are defined by three types of operators. The Large Neighbourhood Search (LNS) framework, first introduced by Shaw (1998) to solve VRP, was introduced by Ropke and Pisinger (2006) to solve PDPTW and by Parragh and Schmid (2013) to solve the DARP in a hybrid model. (Ho and Haugland, 2011) (Archetti et al., 2006) (Masmoudi et al., 2016) (Masmoudi et al., 2018).

2.5 Applicable Machine Learning Techniques

Besides the conventional exact and heuristic solution methods, various Machine Learning (ML) techniques have also been applied to tackle the NP-hard combinatorial optimisation problems. Given that machine learning is more data driven rather than model focused, the integration of combinatorial optimisation problems and machine learning methodologies might yield promising results especially when the problem is stochastic or dynamic.

Learning methods can be categorised into two types: imitation setting (e.g., supervised learning/demonstration) and experienced setting (e.g., Reinforcement Learning). In the imitation setting, policies are learned through supervised targets which are provided by an expert (oracle) without any reward at every step. On the contrary, in Reinforcement Learning, agents learn through accumulating rewards. When applied to solving combinatorial optimisation problems, Machine Learning models usually have one out of the following three algorithmic structures. (Bengio et al., 2021)

Fist of all, End-to-End learning is often applied, where ML models are directly trained to solve discrete optimisation problems on a network level. introduce Pointer Networks architecture which allows the length of the output sequence is defined by input sequence Vinyals et al. (2015) introduce Pointer Networks (Ptr-Net) architecture which allows the length of the output sequence is defined by input sequence based on Seq2Seq models. This new model overcomes the shortcomings of vanilla Sequence to sequence models where the output dictionary is fixed, meaning the models need to be retrained for every new instance. The proposed Ptr-Net architecture integrates the Seq2Seq model (with an RNN encoder and and RNN decoder) and attention model which enables it to solve combinatorial optimisation problems where the output dictionary depends on the elements of the input sequence. The model is trained with supervised learning to heuristic solutions, which results in the performance of the model largely depends on the performance of the designated heuristics algorithms. Bello et al. (2016) propose a framework to solve combinatorial optimisation problems (TSP

and Knapsack problem) using RL and neural networks. They propose two approaches: RL pre-training which uses a training set to optimise the parameters of a stochastic policy using negative tour length as objective (reward); and active search which involves no pre-training and optimise the RNN parameters as well as the best sampled solution using one test instance. In their neural network architecture for TSP, they follow the Vinyals et al. (2015)'s Ptr-Net approach with the chain rule to define the probability of a tour, aiming to learn the parameters of a stochastic policy given a set of input nodes. They use an actor-critic algorithm to train two sets of parameters (pointer network and critic network) simultaneously, where critic network is trained by the mean squared error between predicted and actual tour length to improve the action value function. Nazari et al. (2018) propose a framework for solving VRPs using RL. They train a stochastic (single) policy model by observing reward signals using policy gradient algorithm. Their proposed structure performs on a set of problems sampled from a given distribution (of locations and demands) without having to train the policy for each instance, while Bello et al. (2016) assume a static system in their model. In their proposed pointer network, they omit the encoder RNNs since the sequence of the input is trivial. Instead, they use a set of graph embeddings to encode structured data inputs and an RNN decoder network that points to an input at each decoding step.

Second, learning is often used to configure algorithms, where additional information to a combinatorial optimisation algorithm is provided by ML models. Zhao et al. (2020) propose an actor-critic Deep Reinforcement Learning (DRL) model with an actor, an adaptive critic and a routing simulator for solving VFPTW (VRP with Time Windows). The actor generates policies (strategies) on routing; the adaptive critic with a baseline (to reduce variance) predicts the advantage function and changes the network structure adaptively in order to accelerate convergence in training. The routing simulator provides dynamic states (graphic information and reward) at each time step, and the output of DRL model serves as an initial solution for local search heuristic. Bongiovanni et al. (2020) propose an extension to Large Neighbourhood Search (LNS) Metaheuristics which employs machine learning (supervised learning) to select destroy/repair couple from alternative algorithms. At each iteration, a destroy/repair couple is selected from the regression learning model (random

forest) and employed to incumbent solution until the maximal number of non-improvement iterations is exceeded or time out.

Third, ML could be applied alongside optimisation algorithms, where a problem with a high-level master optimisation problem structure is defined while machine learning models are called iteratively to make lower level decisions. Montoya et al. (2016) solve the sub-problem of a set partitioning formulation of Green VRP by assembling columns using a multi-space sampling heuristic. Lodi and Zarpellon (2017) review the (supervised, unsupervised and reinforcement) learning techniques applied on branch and bound algorithm to deal with variable and node selections. Shi et al. (2019) propose an off-policy RL framework integrating decentralised learning and centralised decision making to solve Electric Vehicle (EV) fleet dispatch problem. The decentralised learning process employs s parameterised deep Feed-forward Neural Network (FNN) to approximate the state value functions of EVs, while the centralised fleet dispatch decision making component avoid conflicts in fleet scheduling. The optimisation of fleet dispatching (maximising action-value function) is converted into a linear assignment problem that can be solved in polynomial time. Iommazzo et al. (2020) propose a MP based model to select solver configuration for a given instance. The model is formulated as an mixed integer nonlinear program with encoded learnt information and finds the optimal configuration for each instance.

Thompson Sampling (TS) (Thompson, 1933), also known as Posterior Sampling or Probability Matching, has recently attracted attention in Stochastic Optimisation Society on a ride range of online decision problems. First proposed to tackle the multi-armed bandit problems under Bayesian setting, Thompson Sampling offers an flexible modelling approach that balances between exploration and exploitation, and maximise the accumulated reward in a fixed amount of time. (Russo et al., 2018) By sampling each option according to their posterior distribution and choosing the action that maximises rewards, TS offers competitive expected regret bounds, which has been shown by various experimental and theoretical studies. (Agrawal and Goyal, 2012, 2013b,a; Russo and Van Roy, 2014) In Operations Research, Ferreira et al. (2018) estimate stochastic demand function

using TS for online network revenue management with constrained inventory. Åkerblom et al. (2020) use Bayesian approaches including TS and Upper Confidence Bound (UCB) to model the stochastic energy consumption for each road segment, attempting to build an online learning model that learns parameters en-route, navigates vehicles adaptively and optimise energy consumption. To the author's best knowledge, TS is scarcely applied in discrete combinatorial optimisation problems such as VRP and its generalisations.

2.6 Summary

This thesis is built upon the literature of the modeling and existing solution methods of VRP and its variations (especially DARPs). Base on literature review, the following gaps have been identified:

1. Existing studies on DARPs fall short of taking users' preferences into consideration. Since most DARP models impose identical time window and ride time constraints to all customers, users' preferences including their tolerances towards travel time, schedule delay, fare, etc. are left unaccounted.

2. Most Demand Responsive Transportation (DRT) operators using classic DARP models assume that user demand is inelastic even when it is not the case (Sayarshad and Chow, 2015). Consequences including risks of no-show and/or cancellation after booking the service (as customers might not be happy with the trip offer) may lead to poor service quality and reduce ridership in the long-run.

3. Thompson Sampling, although proven to be a powerful RL tool for solving online decision problems, has hardly ever been applied in solving discrete combinatorial optimisation problems such as VRP and its variations which are NP-hard.

With an attempt to bridge some of those gaps in literature, this thesis proposes a DARP model integrated with users' preferences from DRT operators' perspective, facilitating them to optimise their overall profit while maintaining service quality. A multi-class Chance Constrained DARP (CC-DARP) model that equips DRT operators with an accept/reject mechanism for user selection is

proposed. The proposed class-base feature of the model takes into account users socio-demographic attributes, which is unique for analysing various fare structures and revenue management at a strategic level. A customized local search based heuristic and a Matheuristic are developed to solve the CC-DARP. The proposed solution method combines local search moves on both user selection and route optimization.

Furthermore, to rise to the computational challenges faced by solving DARPs and its variants derived from various service design, this thesis explores and reproduces some of the state of the art meta-heuristic algorithms. By integrating machine-learning techniques, metaheuristics and the branch-and-price scheme, this thesis proposes a number of hybrid structures to solve DARP, which is a concept that could be further generalised to solve other combinatorial optimisation problems.

Chapter 3

Classic DARP Formulations

3.1 Introduction

The Dial-a-Ride Problem is rich in formulations. Although the features and functions of different DARP models may vary, most of the extended DARP formulations stem from two standard Mixed-Integer Linear Program (MILP) DARP formulations from a modelling perspective: Link-based DARP model or Path-based DARP model. The Link-based formulation uses the flow of each arc (link) of the network as a base unit, presenting constraints at each node and each arc of the system explicitly. On the other hand, the Path-based formulation (often acts as the Master Problem in Column Generation related algorithms) deals with complete paths that start from the origin depot and end at the destination depot, leaving the constraints implicit. Different modelling approaches of the DARP affect the compatibility of certain solution methods when it comes to solving the problem effectively. Link-based formulations are in particular suitable for Branch-and-Cut approaches, whereas Path-based formulations are more tailored for Column Generation based solution methods, where the algorithms start with a small set of known feasible paths, and gradually finding better paths to improve the objective value.

In this chapter, a thorough review on existing Dial-a-ride Problem (DARP) formulations from a

wide range of aspects is presented. Algorithms that are often used for solving the DARP, including Column Generation, Local Search, and some Metaheuristic algorithms, are also presented in this chapter, which act as the backbone of the sophisticated algorithms developed in the following chapters. Mathematical notations and benchmark datasets that are frequently used in DARPs are explained and summarised, for the convenience of further references in the following chapters. Furthermore, data cleaning and processing of the datasets used in this thesis and implementations of Classic DARP using commercial solver CPLEX are documented as a benchmark for the rest of this thesis.

3.2 MIP Formulations for the DARP

3.2.1 Link-based Formulations

In the past few decades, different forms of the DARP have been studied. A standard definition of the DARP with explicitly imposed time-window constraints has been proposed by Cordeau and Laporte (2003b). Cordeau (2006) then established the classic 3-index Mixed-Integer Linear Program (MILP) formulation. In the 3-index DARP formulation, n is defined as the number of requests. The problem is defined on a complete directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, where $\mathcal{N} = \mathcal{P} \cup \mathcal{D} \cup \{0, 2n+1\}$. $\mathcal{A}$ is the set of arcs (i, j), where $i \in \mathcal{N}$ and $j \in \mathcal{N}$. Subset $\mathcal{P} = \{1, \ldots, n\}$ contains pick-up nodes and subset $\mathcal{D} = \{n+1, \ldots, 2n\}$ contains drop-off nodes, while nodes 0 and $2n+1$ represent the origin and destination depots respectively. Each node pair $(i, n+i)$ represents a travel request from origin node i to destination node $n+i$. Note that $\mathcal{P}$ could also represent the set of requests. Set $\mathcal{K}$ is the set of vehicles that are providing dial-a-ride services in this network (travelling at the same speed). Each vehicle $k \in \mathcal{K}$ has a capacity Q_k and its maximal duration of the whole route is T_k. Each request i has an associated load q_i (number of passengers) and a non-negative service duration d_i at its pick-up. Note that $q_0 = q_{2n+1} = d_0 = d_{2n+1} = 0$, and $q_i = -q_{n+i}$. The earliest and latest time that service may begin at node i is the time window $[e_i, l_i]$. For each arc $(i, j) \in \mathcal{A}$ in the network, c_{ij} represents its travel cost and t_{ij} represents its direct travel time. L is the maximum acceptable travel time.

Variables and parameters	
x_{ij}^k	Binary variable equal to 1 if vehicle k travels from node i to j
B_i^k	The time at which vehicle k begins/finishes services at depot
B_i	The time at which vehicle begins services user i
Q_i^k	The load on vehicle k when it arrives at node i
Q_i	The on-board load at node i
$[e_i, l_i]$	The time window within which the service may begin at node i
L_i	The ride time of user i
t_{ij}	The travel time between node i and j
c_{ij}	The routing cost between node i and j
q_i	The load of request i
Q_k	The maximum capacity of vehicle k
T_k	The maximum route duration of vehicle k
Sets	
$\mathcal{N}$	Set of nodes, $\mathcal{N} = \mathcal{P} \cup \mathcal{D} \cup \{0, 2n+1\}$
$\mathcal{A}$	Set of arcs (i, j)
$\mathcal{P}$	Set of pick-up nodes or set of requests, $\mathcal{P} = \mathcal{IB} \cup \mathcal{OB}$
$\mathcal{D}$	Set of drop-off nodes
$\mathcal{IB}$	Set of origins of inbound trips
$\mathcal{OB}$	Set of origins of outbound trips
$\mathcal{K}$	Set of vehicles

Table 3.1: Notations in the Classic 3-index DARP formulation

For each arc $(i, j) \in \mathcal{A}$ and each vehicle $k \in \mathcal{K}$, let x_{ij}^k be a binary decision variable equal to 1 if the shared mobility service vehicle k travels directly from node i to node j. For each node $i \in \mathcal{N}$, let B_i^k be the time at which node i is served by vehicle k of the dial-a-ride system and let Q_i^k be the load of vehicle k after visiting node i. L_i^k is defined as the travel time of user i if assigned to vehicle k. Note that for homogeneous vehicles, one can reduce the number of constraints and variables by using aggregated time and load variables B_i and Q_i at every node except for the depot nodes $\{0, 2n+1\}$ (Cordeau, 2006). A summary of the mathematical notations used throughout the formulation is provided in Table 3.1. The mixed-integer linear programming formulation for the Classic 3-index DARP model is summarised as follows.

Model 1.

$$\min z = \sum_{k \in \mathcal{K}} \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} c_{ij} x_{ij}^k, \tag{3.1a}$$

subject to:

$$\sum_{j\in\mathcal{N}}\sum_{k\in\mathcal{K}} x_{ij}^k = 1, \qquad \forall i \in \mathcal{P}, \tag{3.1b}$$

$$\sum_{j\in\mathcal{N}} x_{ij}^k - \sum_{j\in\mathcal{N}} x_{n+i,j}^k = 0, \qquad \forall i \in \mathcal{P}, k \in \mathcal{K}, \tag{3.1c}$$

$$\sum_{j\in\mathcal{N}} x_{0j}^k = 1, \qquad \forall k \in \mathcal{K}, \tag{3.1d}$$

$$\sum_{i\in\mathcal{N}} x_{i,2n+1}^k = 1, \qquad \forall k \in \mathcal{K}, \tag{3.1e}$$

$$\sum_{j\in\mathcal{N}} x_{ji}^k - \sum_{j\in\mathcal{N}} x_{ij}^k = 0, \qquad \forall i \in \mathcal{P}\cup\mathcal{D}, k \in \mathcal{K}, \tag{3.1f}$$

$$B_j \geq (B_0^k + d_0 + t_{0j})x_{0j}^k, \qquad \forall j \in \mathcal{N}, k \in \mathcal{K}, \tag{3.1g}$$

$$B_j \geq (B_i + d_i + t_{ij})\sum_{k\in\mathcal{K}} x_{ij}^k, \qquad \forall i,j \in \mathcal{N}, \tag{3.1h}$$

$$B_{2n+1}^k \geq (B_i + d_i + t_{i,2n+1})x_{i,2n+1}^k, \qquad \forall i \in \mathcal{N}, k \in \mathcal{K}, \tag{3.1i}$$

$$Q_j \geq (Q_0^k + q_j)x_{0j}^k, \qquad \forall j \in \mathcal{N}, k \in \mathcal{K}, \tag{3.1j}$$

$$Q_j \geq (Q_i + q_j)\sum_{k\in\mathcal{K}} x_{ij}^k, \qquad \forall i,j \in \mathcal{N}, \tag{3.1k}$$

$$Q_{2n+1}^k \geq (Q_i + q_{2n+1})x_{i,2n+1}^k, \qquad \forall i \in \mathcal{N}, k \in \mathcal{K}, \tag{3.1l}$$

$$L_i = B_{n+i} - (B_i + d_i), \qquad \forall i \in \mathcal{P}, \tag{3.1m}$$

$$B_{2n+1}^k - B_0^k \leq T_k, \qquad \forall k \in \mathcal{K}, \tag{3.1n}$$

$$e_i \leq B_i \leq l_i, \qquad \forall i \in \mathcal{P}\cup\mathcal{D}, \tag{3.1o}$$

$$t_{i,n+i} \leq L_i \leq L, \qquad \forall i \in \mathcal{P}, \tag{3.1p}$$

$$\max\{0, q_i\} \leq Q_i \leq \min\{Q_k, Q_k + q_i\}, \qquad \forall i \in \mathcal{N}, k \in \mathcal{K}, \tag{3.1q}$$

$$x_{ij}^k \in \{0,1\}, \qquad \forall i \in \mathcal{N}, j \in \mathcal{N}, k \in \mathcal{K}, \tag{3.1r}$$

In Model 1, the objective function minimises the overall routing cost of all vehicles. Constraints (3.1b) ensure that all requests must be served. Constraints (3.1c) guarantee that pick-up and drop-

off nodes for each request are visited by the same vehicle, whereas constraints (3.1d) and (3.1e) ensure that each route starts at the origin depot and ends at the destination depot. Constraints (3.1f), (3.1g)–(3.1i) and (3.1j)–(3.1l) guarantee flow conservation, time and load consistency respectively. Constraint (3.1m) defines the ride time of each user, which is bounded by (3.1p). The duration of routes are bounded by constraints (3.1n), while (3.1o) and (3.1q) are the time window and capacity constraints respectively. Service time and load consistency constraints (3.1g)–(3.1l), which are the aggregated variables form of

$$B_j^k \geq (B_i^k + d_i + t_{ij})x_{ij}^k, \qquad \forall i, j \in \mathcal{N}, k \in \mathcal{K}, \tag{3.2}$$

$$Q_j^k \geq (Q_i + q_j)x_{ij}^k, \qquad \forall i, j \in \mathcal{N}, k \in \mathcal{K}, \tag{3.3}$$

in order to reduce the number of variables, are nonlinear. They could be easily linearised by introducing a large constant each to (3.2) and (3.3):

$$B_j^k \geq B_i^k + d_i + t_{ij} - M_{ij}^k(1 - x_{ij}^k), \qquad \forall i, j \in \mathcal{N}, k \in \mathcal{K}, \tag{3.4}$$

$$Q_j^k \geq Q_i + q_j - W_{ij}^k(1 - x_{ij}^k), \qquad \forall i, j \in \mathcal{N}, k \in \mathcal{K}. \tag{3.5}$$

Under the assumption of all vehicles share the same arc travel time t_{ij}, Ropke et al. (2007) introduce a 2-index formulation of the DARP, omitting the k indices of variables. To impose precedence constraints and pairing constraints (3.1c), Ropke et al. (2007) introduce set $\mathcal{S}$,a set of all subsets of $S \subseteq \mathcal{N}$ such that $0 \in S$, $2n + 1 \notin S$, and there exists at least one request whose pick-up node $i \in S$ but its drop-off node $n + i \notin S$. The 2-index DARP can then be formulated as the following MIP:

Model 2.

$$\min z = \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} c_{ij} x_{ij}, \tag{3.6a}$$

subject to:

$$\sum_{i \in \mathcal{N}} x_{ij} = 1, \qquad \forall j \in \mathcal{P} \cup \mathcal{D}, \tag{3.6b}$$

$$\sum_{j \in \mathcal{N}} x_{ij} = 1, \qquad \forall i \in \mathcal{P} \cup \mathcal{D}, \tag{3.6c}$$

$$\sum_{i,j \in S} x_{ij} \leq |S| - 2, \qquad \forall S \in \mathcal{S}, \tag{3.6d}$$

$$B_j \geq (B_i + d_i + t_{ij}) x_{ij}, \qquad \forall i, j \in \mathcal{N}, \tag{3.6e}$$

$$Q_j \geq (Q_i + q_j) x_{ij}, \qquad \forall i, j \in \mathcal{N}, \tag{3.6f}$$

$$L_i = B_{n+i} - (B_i + d_i), \qquad \forall i \in \mathcal{P}, \tag{3.6g}$$

$$t_{i,n+i} \leq L_i \leq L, \qquad \forall i \in \mathcal{P}, \tag{3.6h}$$

$$B_{2n+1} - B_0 \leq T, \tag{3.6i}$$

$$e_i \leq B_i \leq l_i, \qquad \forall i \in \mathcal{P} \cup \mathcal{D}, \tag{3.6j}$$

$$\max\{0, q_i\} \leq Q_i \leq \min\{Q, Q + q_i\}, \qquad \forall i \in \mathcal{N}, \tag{3.6k}$$

$$x_{ij} \in \{0, 1\}, \qquad \forall i \in \mathcal{N}, j \in \mathcal{N}, \tag{3.6l}$$

In Model 2, the objective function (3.6a) minimises overall routing cost of the system. Similar to Model 1, constraints (3.6b) and (3.6c) ensures that every node is visited exactly once. Constraints (3.6d) is the new precedent and pairing constraints, which enforce that both pick-up and drop-off of any request is visited by the same vehicle, pick-up visited before drop off. Constraints (3.6e) and (3.6f) ensure the consistency of service time and load variables. Ride-time variables are defined and bounded by constraints (3.6g) and (3.6h), while the respects of route time, time window and capacity are enforced by constraints (3.6i) - (3.6k). Siimlarly, nonlinear constraints (3.6e) and (3.6f) could be linearised using the big M term as indicated by (3.4) and (3.5).

3.2.2 Set Partitioning Formulation

Given the fact that the relaxed LP bound of Model 1 is of poor quality, path-based reformulations of the problem are often used when it comes to solving the problem (especially with column generation). The Dantzig-Wolfe decomposition could be applied to the 3-index MIP formulation to formulate a set partitioning problem. Let Ω denote the set of all feasible paths that satisfy constraints in Model 1. For each path $p \in \Omega$, define binary path variable λ^p, which equals to 1 if the path is used in the solution. The cost of a path p is defined as c_p. Let constant a_i^p indicates the number of times request i is visited by path p. The DARP can then be formulated as a set partitioning problem as follows:

$$\min \sum_{p \in \Omega} c^p \lambda^p \tag{3.7}$$

subject to:

$$\sum_{p \in \Omega} a_i^p \lambda^p = 1, \qquad \forall i \in \mathcal{P}, \tag{3.8}$$

$$\sum_{p \in \Omega} \lambda^p \leq K, \tag{3.9}$$

$$\lambda^p \in \{0, 1\}, \qquad p \in \Omega. \tag{3.10}$$

The objective function (3.7) minimises the overall routing cost of the system. Constraints (3.8) ensures each request is served exactly once, while constraints (3.9) enforce the maximum number of vehicle (routes) in the final solution. This compact set partitioning formulation of the DARP, as well as the set covering formulation (where the equal signs in constraints 3.8 are replaced by $\geq$ for better dual bounds), are often used in the Master Problem of solution methods that involve a column generation based component. A lower bound on the optimal solution value of (3.7) could

be obtained by solving the LP relaxation of problem, which replaces binary constraints (3.10) with:

$$\lambda^p \geq 0, \qquad p \in \Omega. \tag{3.11}$$

3.3 Objective Functions

Given that each DARP may have different priorities when balancing between operational cost and users' inconvenience, dial-a-ride system operators often have the liberty to choose an objective that benefits the systems the most. Minimising the system operational (routing) cost, as seen in various formulations in 3.2, is the most commonly used single objective function in DARP formulations. This objective is often considered equivalent as minimising total distance travelled, since routing cost is normally proportional to geographical distance and/or travel time. Another common single objective is to maximise vehicle occupancy rate, or similarly, minimise fleet size (Diana and Dessouky, 2004). This objective function is applied when reducing users' inconvenience is not emphasised in the system. Furthermore, as reviewed in 2.2, when requests are allowed to be left unserved, the objective can also be related to revenue and profit of the system. An example is Parragh et al. (2014)'s split requests DARP model: with a revenue and a cost associated with each request, the objective is to maximise total system profit.

When there are more incentives to improve service quality in the system, some constraints regarding customers' inconvenience could be transferred into penalty terms in the objective function. In this way, these constraints are soft constraints (a.k.a., could be violated but violations are penalised). These multiple objectives in a model are often aggregated into one weighted-sum objective function, see Mauri and Lorena (2006) and Jorgensen et al. (2007) for examples of weighted-sum objective functions.

3.4 Main Solution Methods

To tackle this NP-hard problem, various exact and heuristic solution methods have been studied and tailored to cater the need of specific DARP constraints in the last few decades, as reviewed in 2.4. In this section, some of these solution methods, which some algorithms in this thesis are based on, will be elaborated more in detail for references. Furthermore, some pre-processing techniques are explained and discussed, as they were also applied in some algorithms in this thesis.

3.4.1 Column Generation and Branch-and-Price Algorithm

The Branch-and-Price algorithm, which is Column Generation embedded in a Branch-and-Bound scheme, is an exact solution method that is often used to solve large-scale instances of (Mixed) Integer Programming problems. In the branch-and-bound tree, each node is solved as a Master Problem (MP) by column generation (pricing). An overlook of the algorithm is summarised in Figure 3.1.

The MP in column generation is often the path-based formulation of the problem after Dantzig-Wolfe decomposition with the integer constraint relaxed (LP relaxation). Due to the large size of the set of all paths, the MP can not be solved directly. Instead, a restricted version of the MP with a subset of limited number of paths is solved, and more paths are gradually generated and added to the pool by Column Generation (CG). In CG, each path is treated as a column in the pool, and more columns with negative reduced costs get added to the pool iteratively. As shown in Figure 3.1, in each iteration, RMP is solved and the dual values associated to the constraints are obtained, which in turn guides the generation of paths with negative reduced costs. Following the notations in the set partitioning formulation in 3.2.2, if d_i is denoted as the dual values associated with constraints 3.8 and σ as the dual value associated to 3.9, the reduced cost of a new path (column) p could be calculated as follows:

$$\bar{c_p} = c_p - \sum_{i \in \mathcal{P}} a_i^p d_i - \sigma \tag{3.12}$$

With the additional Branching scheme, the global Upper Bound (UB) and Lower Bound (LB)

get updated accordingly when branching proceeds and/or when new integer solutions are discovered. The process terminates when UB converges with LB or no more paths with negative reduced cost could be found (note that in the last iteration, all columns are explored extensively), which guarantees the optimality of the solution and makes Branch-and-Price an *exact* solution method. In this thesis, the Branch-and-Price structure is applied in some of the algorithms developed to solve the DARP. However, in these applications, columns are generated using various heuristic algorithms. This variation of the Branch-and-Price algorithm does not necessarily explore all columns when searching for paths with negative reduced costs in the last iteration, which is less time-consuming, but also makes algorithm an approximate solution method by nature, with no guarantee on optimality. Another application of CG in this thesis does not involve branching in the progress: Columns are generated using multiple heuristics and integer solutions are obtained by solving the integer MP every now and then on the growing pool of columns. These algorithms will be further demonstrated in chapter 5.

3.4.2 Classic Insertion Heuristic

In this thesis, various heuristic solution methods are explored, customised and improved to better solve the NP-hard DARP, including local-search based classic heuristics and metaheursitics. An initial solution, preferably feasible, is constructed first before more sophisticated techniques and operators are applied to further improve the solution. In this thesis, the initial solution is constructed based on Jaw et al. (1986)'s Classic Insertion Heuristic solution method, with modifications applied to further satisfy problem-specific requirements.

In Jaw's heurstic, requests are inserted in a temporal sequential process: all requests are ranked by their earliest pick-up time from the earliest to the latest, and inserted one by one according to the increasing order of pick-up times. Schedule time blocks (when the vehicle is transporting passengers) and slack time blocks (when the vehicle is idling, shown as shadowed blocks in Figure 3.2) are constructed and updated after each request is inserted. At each insertion of a request, the time windows of the request is compared against the time slots of each schedule and slack block in a vehicle, and categorise the potential insertion as one of the four following cases (see Figure

3.2): (a) Both the pick-up and drop-off of the request is inserted at the end of the work schedule of a vehicle (i.e. after the end of the last schedule block); (b)Both pick-up and drop-off of the request is inserted consecutively into an active schedule block; (c) The pick-up of the request is inserted within a schedule block, while the drop-off of the request is inserted at the end of this schedule block; and (d) Both pick-up and drop-off of the request is inserted in to an active schedule block, but separated by existing request(s) within the block. All possible insertion combinations will be tried and feasibility will be checked. When there exists more than one feasible insertion, *best insertion* is preferred, meaning the request will be inserted in scenario with minimal additional cost to the existing routes. Readers are refered to Jaw's PhD thesis (Jaw, 1984) for more detail of this base algorithm.

In this thesis, other inserting orders are applied. Therefore, more specific cases other than those in Figure 3.2 need to be considered. For example, when inserting requests from the earliest to the latest, scenarios including inserting a request before an existing schedule block or even before the first schedule block do not exist, since later requests are always inserted into the later part of the vehicle or the existing schedule blocks. These additional scenarios are not trivial, since when implementing the algorithm, the handling of slack time varies depending on the location inside the vehicle schedule as well as the schedule block where the request is being inserted.

3.4.3 Local Search

Local search solution methods, as a group, are built on the idea of attempting to improve the incumbent feasible solution of a problem by making small perturbations. Starting with an initial solution, simple operators (such as "add", "remove" or "swap") are applied to paths of the incumbent solution to generate new solutions. All solutions generated by applying a simple operator to the incumbent solution make up the neighbourhood of the incumbent solution. Improvements are identified within the neighbourhood if the newly generated solution is feasible and yields a lower routing cost comparing to the incumbent solution, which in turn replaces the current solution and becomes the new incumbent. The process terminates when no better solutions could be found in

the neighbourhood, and the best solution returned is called a local optimal solution. In order to break through the bottleneck of local optima and approach global optimality, a number of variants of local search (e.g. Tabu Search, Annealing, LNS, VNS, etc.) have been developed, most of which either expand the neighbourhood by further perturbs the routes or accept temporary deterioration in the incumbent solution. Two Local Search based Metaheuristics, namely Variable Neighbourhood Search (VNS) and Large Neighbourhood Search (LNS), are further introduced in detail in 3.4.4, as the two base metaheuristic algorithms of which part of this thesis is based on.

3.4.4 Metaheuristics Based on Local Seach: Variable Neighbourhood Search (VNS) and Large Neighbourhood Search (LNS)

VNS

The basic Variable Neighbourhood Search (VNS) framework defines a set of neighbourhood structures, each associated with a particular operator and a size of perturbation. Neighbourhoods are often placed in a nested sequence, with different operators applied on successive neighbourhoods and with the size of perturbation increasing from the first neighbourhood to the last one. Starting with an incumbent initial solution, at each iteration n, the algorithm first proceeds to the *shaking* phase, where one of the neighbourhood structures (starting from the first one in the sequence) is applied to modify the incumbent solution. A follow-up intra-route *local search* is then applied to the new solution. If the new solution improves the incumbent solution, it replaces the incumbent and the search restarts from the first neighbourhood structure in the nested sequence. Otherwise, the new solution is discarded and the search proceeds to the next neighbourhood structure in line. The neighbourhood structure used will circulate back to the first one once every structure has been iterated. The procedure terminates once the stopping criterion is met. Commonly used stopping criteria include maximum CPU time exceeded or maximum iterations reached between two improvements found. The overlook of the VNS framework that is adapted in this thesis is summarised in Algorithm 1.

Algorithm 1: VNS Framework

Input: Neighbourhood Structures N_k, $k = 1, ..., k_{max}$, An Initial Solution s_0, Stopping Criterion.

Output: Best solution Found $s^\star$

```
/* Initialisation */
```

$s_{incumbent} \leftarrow s_0$

$k = 1$

while *Stopping Criterion is not met* **do**

```
/* VNS Shaking phase */
```

Generate s' using Neighbourhood Structure N_k

```
/* Local Search */
```

$s'' \leftarrow$ Apply intra-route local search on s'

```
/* Move or Not */
```

if $c_{s''} < c_{s_{incumbent}}$ **then**

$s_{incumbent} \leftarrow s''$, $k \leftarrow 1$

else

$k \leftarrow (k \bmod k_{max}) + 1$

$s^\star \leftarrow s_{incumbent}$

return $s^\star$

LNS

The Large Neighbourhood Search (LNS) framework destroys and repairs the incumbent solution to a large extend. A number of removal and insertion operators could be tailored to be problem-specific, and applied to perturb the incumbent solution. In each LNS iteration, a removal and an insertion operator as well as the number of requests q to be removed and reinserted in this iteration, are all determined randomly. q requests are then removed from the incumbent routes, and reinserted back into the routes using the previously selected operators, attempting to find better

places to insert these q requests. Commonly used removal operators include: (a) *Shaw Removal*: A first request is chosen randomly, and the rest $q-1$ requests are selected based on the similarity (which is a problem-specific concept) to the first request. Requests that are more temporally and/or spatially related are more likely to be removed together; (b) *Random Removal*: Requests are randomly selected and removed; and (c) *Worst Removal*: Requests that saves the most routing cost if removed from the routes are more likely to be selected and removed. Common insertion operators include: (a) *Greedy Insertion*: Requests are inserted into the places where they would cause minimal additional routing cost; and (b) *Regret Insertion*: Requests are inserted into places with higher regret values. If the new repaired solution yields a lower cost comparing to the incumbent solution, it replaces the incumbent solution; Otherwise the new solution gets discarded, and LNS moves on to the next iteration. The procedure terminates when a predefined stopping criterion is met.

LNS could be further extended and improved in various ways, one of which being choosing the next set of operators in an adaptive manner, based on the recent performance of each removal/insertion operator, rather than randomly. Other techniques including allowing small deterioration when nominating incumbent solutions (i.e., annealing), adding noise when calculating routing cost at each iteration, and applying intra-route local search to further improve the repaired solution. The overlook of the LNS framework that is adapted in this thesis is summarised in Algorithm 2.

3.5 Dataset Description

In this thesis, two datasets are used to test the performances of the models and algorithms. The first dataset is an artificial, randomly generated dataset proposed by Cordeau (2006). This dataset is widely used for benchmarking the performances of algorithms designed for solving the DARP and its variants. The second dataset is generated solely for the study of integrating users' utility

Algorithm 2: LNS Framework

Input: An Initial Solution s_0, Stopping Criterion.
Output: Best solution Found $s^\star$

$s_{incumbent} \leftarrow s_0$
while *Stopping Criterion is not met* **do**
  Randomly generate integer q.
  Select Removal and Insertion Heuristics for this iteration.
  Remove q requests from $s_{incumbent}$.
  Reinsert the removed requests into $s_{incumbent}$, obtaining s'
  if $c_{s'} < c_{s_{incumbent}}$ **then**
    $s^\star \leftarrow s'$
  if s' *meets the annealing acceptance criteria* **then**
    $s_{incumbent} \leftarrow s'$
return $s^\star$

into DARP in this thesis (presented in Chapter 4). As these trips are observed data from taxi meter records, taxi is proven to be these passengers' best travel option, which fits in this thesis where these trips are compared against the dial-a-ride service as the alternative mode. The details of these mentioned daetasets are presented as follows.

3.5.1 Cordeau (2006)'s Dataset

The first set of instances used in the experiments corresponds to the 'A' instances of Cordeau (2006) (`http://neumann.hec.ca/chairedistributique/data/darp/`) with homogeneous vehicles of capacities $Q = 3$. Instances in this dataset contains 16-96 requests served by 2-8 vehicles. The requests in each instance are evenly divided into 50% outbound trips and 50% inbound trips. The coordinates of the pickup and drop-off locations are randomly generated and uniformly distributed in the [-10, 10] $\times$ [-10, 10] square. For each request, a 3-min service time and a 15-min time window is imposed on either its departure or arrival time (but not both), depending on if it is an inbound or an outbound trip. The maximum ride time L is set to be 30 minutes, while the length of planning horizon T is 60 minutes. In Cordeau (2006)'s experiments, the routing cost c_{ij} between location i and j is set to the Euclidean distance $dist_{ij}$ between nodes i and j (i.e., $c_{ij} = dist_{ij}$). In the experiments of this thesis, it is assumed that the routing cost is set to $c_{ij} = 2dist_{ij}$ ($\forall i, j \in N$).

3.5.2 NYC Taxi Dataset

The second set of instances is extracted from Yellow Taxi Trip Data in NYC (2006) (`https://data.cityofnewyork.us/Transportation/2016-Yellow-Taxi-Trip-Data/k67s-dv2t`). 23,772 trip records of the morning peak hours (6am-9am) on a random weekday (1st of February 2016) were extracted, which contain the information of the pickup and drop off coordinates, number of passengers, pickup and drop off time of each trip. All instances used in this thesis has been made public and available at the repository: `https://github.com/davidrey123/DARP`.

For preparation of the further discussion on fare structures, the data is further processed as follows:

- Step 1: Zoning. In this thesis, Manhattan is divided into two Zones (Zone 1 and Zone 2) as shown in Figure 3, and only keep the trips of which the trip destination is located within these two zones. trips with a drop-off location outside Zone 1 and Zone 2 are then filtered out. Note that only the trips with destinations outside Manhattan were filtered out, the pickup locations of the trips are not limited.

- Step 2: Labelling. On the map, the area of Zone 1 and Zone 2 is split into 120,000 35m $\times$ 30m mutually exclusive sub-regions by grouping up the latitude and longitude range of the area into bins, and label each trip with two sub-regions: one for pickup coordinates and one for drop off coordinates, based on which sub-regions of the trip's OD falls into.

- Step 3: Ranking. In this thesis, most visited sub-regions are obtained by ranking the frequency of each sub-region is visited and labeled by trip ODs. By mapping the 50 most popular pickup sub-regions overall and the 20 most popular drop off sub-regions in each drop-off zone, 143 eligible trips were selected with most frequently visited origins and destinations of the day and various numbers of passengers in each request (see Figure 3.3).

The details of these 143 trip records (including Origin-Destination Coordinates, fares, etc) were

further extracted and reformed into instances of requests for testing the performances of DARP models in this thesis.

3.6 CPLEX Implementation

This section presents a number of simple pre-processing techniques (applied prior to the solving process), as well as some valid inequalities (aka cuts) applied in this thesis to speed up the performance of solution algorithms. Instances from Cordeau (2006)'s dataset solved by the latest version of commercial server CPLEX (v12.10) is then presented, which serves as a benchmark for the rest of the thesis in terms of CPU time and best solution found (within a time limit).

3.6.1 Pre-processing Techniques

Time Windows Tightening

For requests in a DARP setting, a tight time window is often imposed on either pick-up or the drop-off location, but not both, leaving the service time at the other location technically unbounded. Combined with the maximum ridetime constraint, the time window of the unbounded node can be tightened. More specifically, if L and T denote maximal ride time and maximal trip time respectively, time windows (e_i, l_i) at the pick-up node i for outbound trips could be tightened as follows:

$$l_i = \min\{l_{n+i} - t_{i,n+i} - d_i, T\}, \qquad \forall i \in \mathcal{OB},$$
$$e_i = \max\{0, e_{n+i} - L - d_i\}, \qquad \forall i \in \mathcal{OB}.$$

Similarly, for inbound trips, the time windows at the drop-off nodes can be tightened as follows:

$$l_{n+i} = \min\{l_i + d_i + L, T\}, \qquad \forall i \in \mathcal{IB},$$
$$e_{n+i} = \min\{0, e_i + d_i + t_{i,n+i}\}, \qquad \forall i \in \mathcal{IB}.$$

Arc Elimination

Given the constraints of the DARP, the following links are by default infeasible:

- Link $(0, n+i)$, $(i, 2n+1)$ and $(n+i, i)$, $\forall i \in \mathcal{P}$, given the precedence constraints of the DARP.
- Link (i, j), if $e_i + d_i + t_{ij} > l_i$, given the time window constraints. (d_i is the service time needed at location i.)
- Link $(i, 0)$ and $(2n + i, i)$, $\forall i \in \mathcal{P}$, given no cycle is allowed.

Generalised Order Constraints

For every pair of requests $i, j \in \mathcal{P}$, no more than one of four following links among the nodes can be established simultaneously:

$$x_{ij} + x_{n+i,j} + x_{n+j,i} + x_{j,n+1} \leq 1$$

3.6.2 Implementation Results

With the aforementioned pre-processing techniques applied prior to solving, Cordeau (2006)'s Set 'A' instances were solved by commercial solver CPLEX with a one-hour time limit. Table 3.2 reports the number of vehicles (K), the number of requests (n), best integer solution found within time limit (Bound), CPU time and optimality gap of each instance. All experiments were implemented in Python on a personal computer with 16GB of RAM and an Intel i7 processor at 2.9GHz. and CPLEX 12.10 is used to solve the classic DARP MILPs. All instances with integer solutions found by CPLEX within the 1-hour time limit are listed in table 3.2.

Instance	K	n	Objective Value	CPU (s)	Gap (%)
a2-16	2	16	294.25*	2.47	$< 1e^{-4}$
a2-20	2	20	344.83*	12.86	$< 1e^{-4}$
a2-24	2	24	431.12*	14.06	$< 1e^{-4}$
a3-18	3	18	300.48*	212.47	$< 1e^{-4}$
a3-24	3	24	344.83*	527.25	$< 1e^{-4}$
a3-30	3	30	510.25	3600.00†	14.47%
a3-36	3	36	583.19	3600.00†	11.10%
a4-16	4	16	282.68	3600.00†	11.58%
a4-24	4	24	376.00	3600.00†	13.17%
a4-32	4	32	486.57	3600.00†	25.84%

†: CPLEX timed out

*: Converged (Optimal Solution Found)

Table 3.2: CPLEX Benchmark Implementation

Results in Table 3.2 will serve as a benchmark in terms of computing power of the commercial solver and algorithmic performance within a time limit in this thesis.

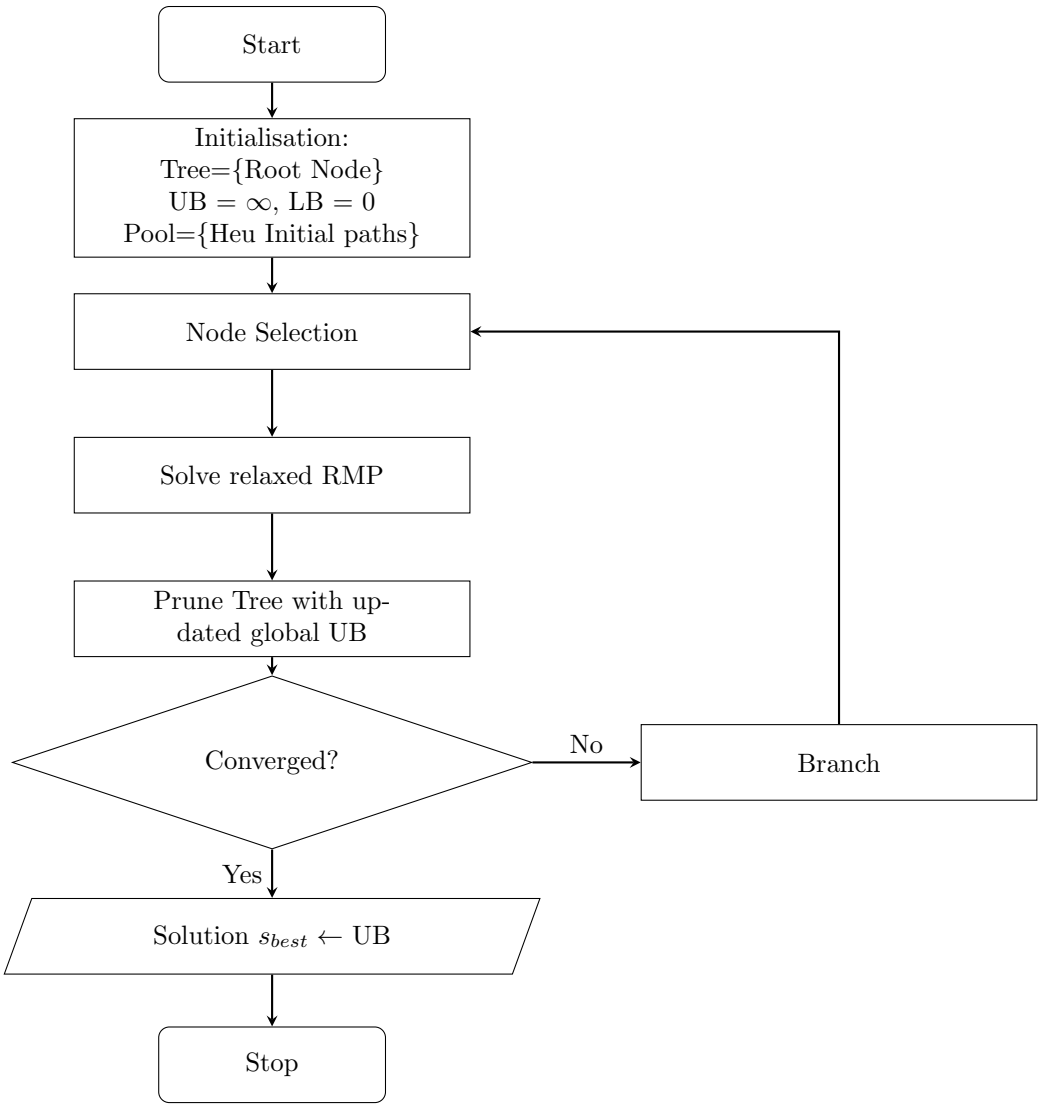


Figure 3.1: Branch-and-Price Solution Method Overlook

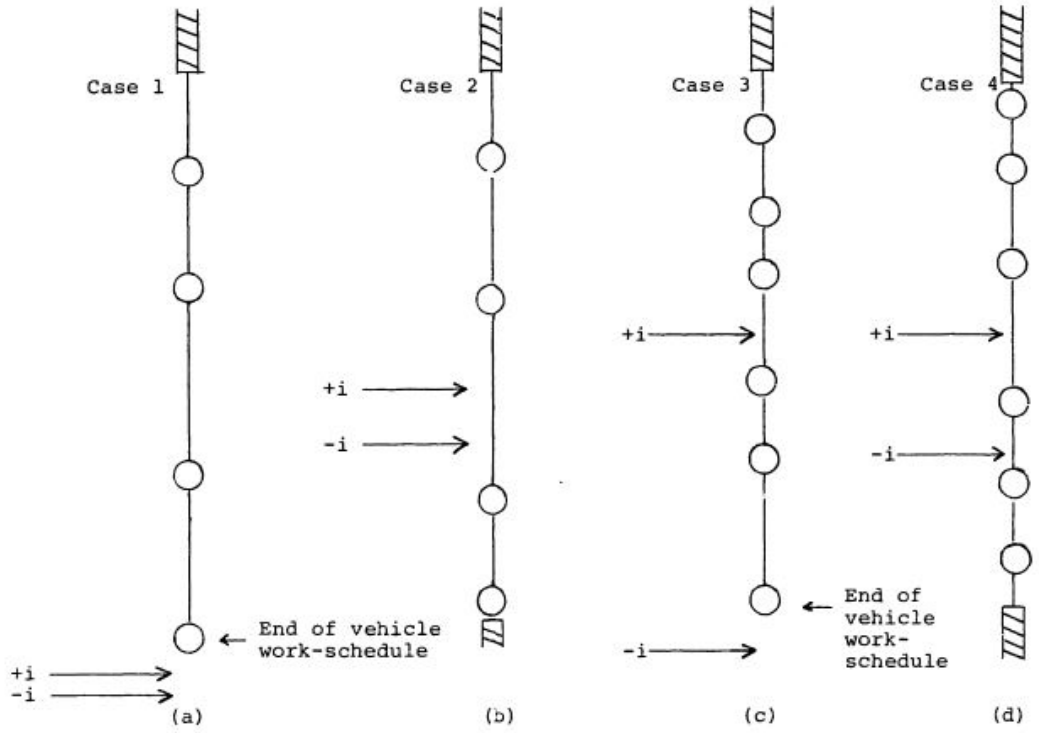
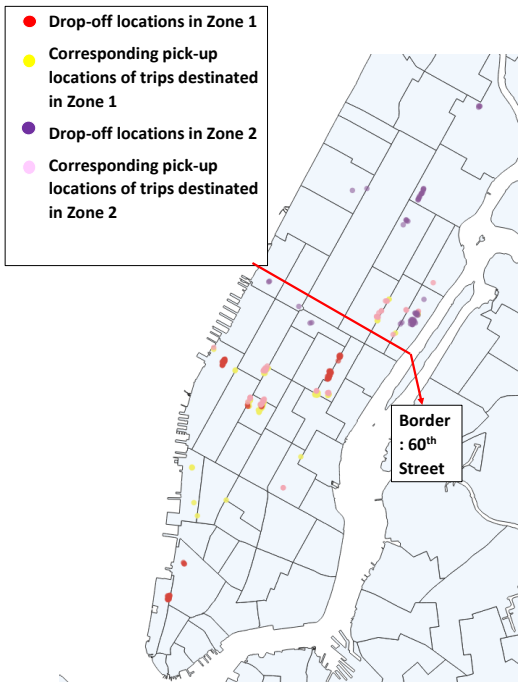
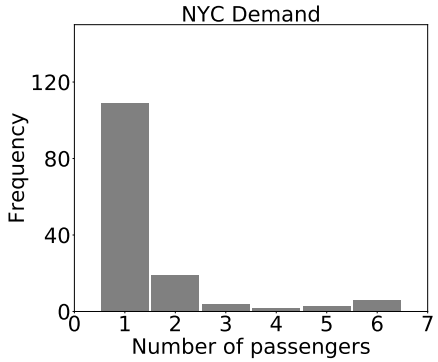


Figure 3.2: 4 Cases in Jaw's insertion Heuristic (Jaw et al., 1986)



(a) Map of pickup and dropoff node locations.



(b) Histogram of number of passengers per request.

Figure 3.3: NYC Dataset (143 requests, 286 nodes)

Chapter 4

Integrating Users' Utility in DARP

4.1 Introduction and Notations

The standard formulation of the DARP (Cordeau, 2006) has been extensively studied over decades. Research on the DARP tends to incorporate the problem with various real-life characteristics to extend its applicability on practical matters (e.g. rich vehicle routing problems). One of the principals of classic (static) DARP is to serve all requests, and to optimise the routing and scheduling of vehicles in order to service all users. Sometimes, in order to serve one or two requests, significant adjustments in routing must be made, without necessarily benefiting either system routing cost or the overall service quality of users. Demand Responsive Transportation (DRT) providers/operators are often ridership-dependent. Persistently serving all requests at the cost of compromising users' experiences of on-board passengers can be rather myopic , considering service quality is the major incentive for stable future ridership (Yoon et al., 2020). Therefore, strategic planning, including designing fare policies for a fleet that operates as a DARP, is also critical (albeit overlooked) in DRT services.

Classic DARP	
Variables:	
x_{ij}^k	Binary variable equal to 1 if vehicle k travels from node i to j
B_i^k	The time at which vehicle k begins/finishes services at depot
B_i	The time at which vehicle begins services user i
Q_i^k	The load on vehicle k when it arrives at node i
Q_i	The on-board load at node i
L_i	The ride time of user i
Parameters:	
$[e_i, l_i]$	The time window within which the service may begin at node i
t_{ij}	The travel time between node i and j
c_{ij}	The routing cost between node i and j
q_i/q_{im}	The load of request i (in class m)
Choice model	
Variables:	
V_i	Representative utility value of DRT service for user i;
y_i	Binary variable equal to 1 if user i is served and 0 otherwise
Parameters:	
$\hat{V}_i$	Representative utility value of 'private travel' mode for user i;
β_T, β_S	Parameters that convert travel time and schedule delay into monetary units
f_i/f_{im}	Fare charged to user i (in class m) according to a specified fare structure
$\hat{c}_i$	Travel cost of private travel mode for user i
β_F	Coefficient for travel cost
Sets	
$\mathcal{N}$	Set of nodes, $\mathcal{N} = \mathcal{P} \cup \mathcal{D} \cup \{0, 2n+1\}$
$\mathcal{A}$	Set of arcs (i, j)
$\mathcal{P}$	Set of pick-up nodes or set of requests, $\mathcal{P} = \mathcal{IB} \cup \mathcal{OB}$
$\mathcal{D}$	Set of drop-off nodes
$\mathcal{IB}$	Set of origins of inbound trips
$\mathcal{OB}$	Set of origins of outbound trips
$\mathcal{K}$	Set of vehicles
$\mathcal{M}$	Set of classes
$\mathcal{Z}$	Set of geographic zones

Table 4.1: Notations

Furthermore, existing studies on DARPs fall short of taking users' preferences into consideration. Since most DARP models impose identical time window and ride time constraints to all customers, users' preferences including their tolerances towards travel time, schedule delay, fare, etc. are left unaccounted. Most DRT operators using classic DARP models assume that user demand is inelastic even when it is not the case (Sayarshad and Chow, 2015). Consequences including risks of no-show and/or cancellation after booking the service (as customers might not be happy with the trip offer) may lead to poor service quality and reduce ridership in the long-run.

In this chapter, two DARP models (one deterministic, one stochastic) integrated with users' preferences from DRT operators' perspective are proposed, facilitating them to optimise their overall profit while maintaining service quality. In these models, users' utility users' preferences are taken into account within a dial-a-ride problem. Notations used in this chapter are summarised in Table 4.1

4.2 Deterministic Model: DARP-UP

In this section, a deterministic model that incorporates users' preference decisions within a rich DARP formulation is proposed. The purpose of this model is to build a DARP model that helps with boosting ridership. To achieve this goal, this thesis identifies Users' Preferences (UP) decisions through utility functions, and integrate them within the DARP. This thesis hereby refer this model as DARP-UP model. Specifically, it is assumed that two travel modes are available: a DARP service and a private travel option. In this thesis, utility functions for each travel mode are integrated within a classic DARP formulation. In this section, a arc-based formulation of the DARP-UP model is detailed and the integration of DARP and users' preferences decisions is explained.

4.2.1 Mathematical Formulation

This thesis follow the formulation of Cordeau (2006) for the classical part of the proposed DARP model and briefly recall the main elements of this formulation hereafter. Let n be the number of

travellers' travel requests. The classical DARP is defined on a complete directed graph $G = (V, A)$, where $N = P \cup D \cup \{0, 2n+1\}$. Subset $P = \{1, \ldots, n\}$ contains pick-up nodes and subset $D = \{n+1, \ldots, 2n\}$ contains drop-off nodes, while nodes 0 and $2n+1$ represent the origin and destination depots. Each node pair $(i, n+i)$ represents a travel request from origin node i to destination node $n+i$. K is denoted as the set of vehicles that are providing shared-mobility service in this network. Each vehicle $k \in K$ has a capacity Q_k and its maximal duration of the whole route is T_k. Each request i has an associated load q_i (number of passengers) and a non-negative service duration d_i. In this section, it is assumed that $q_0 = q_{2n+1} = d_0 = d_{2n+1} = 0$, and $q_i = -q_{n+i}$. The earliest and latest time that service may begin at node i is the time window $[e_i, l_i]$. For each arc $(i, j) \in A$ in the network, c_{ij} represents its travel cost and t_{ij} represents its travel time. Denote L the maximum acceptable travel time of a traveller when using the shared mobility service.

For each arc $(i, j) \in A$ and each vehicle $k \in K$, let x_{ij}^k be a binary decision variable equal to 1 if the shared mobility service vehicle k is used from node i to node j. For each node $i \in N$, let B_i^k be the time at which node i is serviced by dial-a-ride system and let Q_i^k be the load of vehicle k after visiting node i. Denote L_i^k as the travel time of user i using the shared mobility service if assigned to vehicle k. Note that for homogeneous vehicles, one can reduce the number of constraints and variables by using aggregated time and load variables B_i and Q_i at every node except for the depot nodes $\{0, 2n+1\}$Cordeau (2006).

Let β_i be the trip utility for user i expressed in monetary units, this utility is independent of the mode used to make the trip. Let β_T and β_S be parameters that convert travel time and schedule delay into monetary units. For private travel, travel time is assumed to be fixed and equal to $t_{i,n+i}$ and that schedule delay is null. For the shared mobility service, travel time is represented by variable L_i which depends on the route servicing user i. Schedule delay for shared mobility service is defined based on trip types. For inbound trips (trips with a tight time window at pick-up nodes), schedule delay is defined as waiting time $B_i - e_i$, whereas for outbound trips, schedule delay is defined as the amount of time that user i arrives earlier than expected $l_{n+i} - B_{n+i}$, assuming the upper bound of time window l_{n+i} is the preferred arrival time. Let β_F be the coefficient for monetary travel cost, which is the fare f_i for shared mobility service and cost $\hat{c}_i$ (including gas, maintenance, registration,

etc.) for private travel mode. The utility functions for the shared mobility service U_i and private travel $\hat{U}_i$ are summarized below:

$$U_i = \beta_i - \beta_T L_i - \beta_S(B_i - e_i) - \beta_F f_i \qquad \forall i \in IB, \tag{4.1}$$

$$U_i = \beta_i - \beta_T L_i - \beta_S(l_{n+i} - B_{n+i}) - \beta_F f_i \qquad \forall i \in OB, \tag{4.2}$$

$$\hat{U}_i = \beta_i - \beta_T t_{i,n+i} - \beta_F \hat{c}_i \qquad \forall i \in P \tag{4.3}$$

Observe that the utility of the shared mobility service depends on the collective choice of travellers whereas the utility of private travel is fixed. Let $y_i \in \{0, 1\}$ represent the shared mobility service operator's choice to serve user i (1) or not (0). Assuming that travellers are rational and seek to maximize their trip utility, the following constraints are introduced:

$$\sum_{j \in N} \sum_{k \in K} x_{ij}^k = y_i \qquad \forall i \in P \tag{4.4}$$

$$(\hat{U}_i - \underline{U}_i)(1 - y_i) + U_i \geq \hat{U}_i \epsilon \qquad \forall i \in P \tag{4.5}$$

Constraint (4.4) links variable y_i to the route optimization variable x_{ij}^k enforcing that request i is serviced only if $y_i = 1$. ϵ is a tolerance parameter. In this case, constraint (4.5) ensures that if user i is serviced (i.e. $y_i = 1.$), U_i must be greater than $\hat{U}_i$ by $\hat{U}_i \epsilon$. On the other hand, when user i is not serviced by dial-a-ride system (i.e. $y_i = 0.$), Constraint (4.5) is redundant.

$$\underline{U}_i \leq U_i \leq \overline{U}_i \qquad \forall i \in P \tag{4.6}$$

$$\underline{U}_i = \beta_i - \beta_T L - \beta_S(l_i - e_i) - \beta_F \overline{f_i} \qquad \forall i \in P, \tag{4.7}$$

$$\overline{U}_i = \beta_i - \beta_T t_{i,n+i} - \beta_F \underline{f_i} \qquad \forall i \in P, \tag{4.8}$$

Constraint (4.7) defines the lower bound of U_i by setting both travel time and schedule delay to the maximum, whereas constraint (4.8) calculates the upper bounds by minimizing travel time to direct travel time between origin and destination $t_{i,n+i}$ and eliminating schedule delay. $\underline{f_i}$ and $\overline{f_i}$ are upper and lower bounds of fare f_i, respectively.

A small pre-processing step is performed prior to solving DARP-UP. Since U_i is bounded by $[\underline{U}_i, \overline{U}_i]$ and $\hat{U}_i$ is fully fixed, the value of $\hat{U}_i$ could be compared with the bounds of U_i for each user i:

$$If \quad \hat{U}_i \leq \underline{U}_i \qquad then \quad y_i = 1; \tag{4.9}$$

$$If \quad \hat{U}_i \geq \overline{U}_i \qquad then \quad y_i = 0. \tag{4.10}$$

By fixing y_i when $\hat{U}_i$ lies outside $[\underline{U}_i, \overline{U}_i]$, the number of variables could be lessened and hence slightly speed up computational time.

After incorporating utility functions (4.1), (4.2) and (4.3) within constraint (4.5), the mixed-integer linear programming formulation for the (deterministic) DARP with users' preference is summarised as following:

$$\max z = \sum_{i \in P} f_i y_i - \sum_{k \in K} \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ij}^k \tag{4.11}$$

subject to

$$\sum_{j \in N} \sum_{k \in K} x_{ij}^k = y_i \qquad \forall i \in P, \tag{4.12}$$

$$\sum_{j \in N} x_{ij}^k - \sum_{j \in N} x_{n+i,j}^k = 0 \qquad \forall i \in P, k \in K \tag{4.13}$$

$$\sum_{j \in N} x_{0j}^k = 1 \qquad \forall k \in K, \tag{4.14}$$

$$\sum_{i \in N} x_{i,2n+1}^k = 1 \qquad \forall k \in K, \tag{4.15}$$

$$\sum_{j\in N} x_{ji}^k - \sum_{j\in N} x_{ij}^k = 0 \qquad \forall i \in P \cup D, k \in K, \tag{4.16}$$

$$B_j \geq (B_0^k + d_0 + t_{0j}) x_{0j}^k \qquad \forall j \in N, k \in K, \tag{4.17}$$

$$B_j \geq (B_i + d_i + t_{ij}) \sum_{k\in K} x_{ij}^k \qquad \forall i \in N, j \in N, \tag{4.18}$$

$$B_{2n+1}^k \geq (B_i + d_i + t_{i,2n+1}) x_{i,2n+1}^k \quad \forall i \in N, k \in K, \tag{4.19}$$

$$Q_j \geq (Q_0^k + q_j) x_{0j}^k \qquad \forall j \in N, k \in K, \tag{4.20}$$

$$Q_j \geq (Q_i + q_j) \sum_{k\in K} x_{ij}^k \qquad \forall i \in N, j \in N, \tag{4.21}$$

$$Q_{2n+1}^k \geq (Q_i + q_{2n+1}) x_{i,2n+1}^k \qquad \forall i \in N, k \in K, \tag{4.22}$$

$$L_i = B_{n+i} - (B_i + d_i) \qquad \forall i \in P, \tag{4.23}$$

$$B_{2n+1}^k - B_0^k \leq T \qquad \forall k \in K, \tag{4.24}$$

$$e_i \leq B_i \leq l_i \qquad \forall i \in N, \tag{4.25}$$

$$t_{i,n+i} \leq L_i \leq L \qquad \forall i \in P, \tag{4.26}$$

$$\max\{0, q_i\} \leq Q_i \leq \min\{Q, Q + q_i\} \quad \forall i \in N, \tag{4.27}$$

$$\hat{U}_i = \beta_i - \beta_T t_{i,n+i} - \beta_F \hat{c}_i \qquad \forall i \in P, \tag{4.28}$$

$$U_i = \beta_i - \beta_T L_i - \beta_S (B_i - e_i) - \beta_F F \quad \forall i \in IB, \tag{4.29}$$

$$U_i = \beta_i - \beta_T L_i - \beta_S (l_{n+i} - B_{n+i}) - \beta_F F \quad \forall i \in OB, \tag{4.30}$$

$$(\hat{U}_i - \underline{U}_i)(1 - y_i) + U_i \geq \hat{U}_i \epsilon \qquad \forall i \in P, \tag{4.31}$$

$$\underline{U}_i \leq U_i \leq \overline{U}_i \qquad \forall i \in P, \tag{4.32}$$

$$\underline{U}_i = \beta_i - \beta_T L - \beta_S (l_i - e_i) - \beta_F F \qquad \forall i \in P, \tag{4.33}$$

$$\overline{U}_i = \beta_i - \beta_T t_{i,n+i} - \beta_F F \qquad \forall i \in P, \tag{4.34}$$

$$x_{ij}^k \in \{0, 1\} \qquad \forall i \in N, j \in N, k \in K, \tag{4.35}$$

$$y_i \in \{0, 1\} \qquad \forall i \in P. \tag{4.36}$$

$$\underline{f_i} \leq f_i \leq \overline{f_i} \qquad \forall i \in P, \tag{4.37}$$

The objective function (4.11) is designed to maximize profit, which is calculated by deducting routing cost from revenue. Constraint (4.13) guarantees that pick-up and drop-off nodes for each request are visited by the same vehicle, whereas constraints (4.14) and (4.15) ensure that the route starts at the origin depot and ends at the destination depot. Constraints (4.16),(4.17)-(4.19) and (4.20)-(4.22) guarantee flow conservation, time and load consistency respectively. Constraint (4.23) defines the ride time of each user, which is bounded by (4.26). The duration of routes are bounded by constraint (4.24), while (4.25) and (4.27) guarantee the time window and capacity constraints.

Note that in the objective function (4.11), the product of user-based fare variable formulation f_i and binary variable y_i results in a bilinear term $\sum_{i \in P} f_i y_i$. This could be linearised by using the classical exact linearisation for binary-continuous bilinear terms Fortet (1960). This thesis introduce variable $G_i = f_i y_i$ and substitute the bilinear term with G_i in objective function. Furthermore, the following constraints are added to the model for linearisation purposes:

$$G_i \leq y_i \overline{f_i} \qquad \forall i \in P, \tag{4.38}$$

$$G_i \geq y_i \underline{f_i} \qquad \forall i \in P, \tag{4.39}$$

$$G_i \leq f_i - (1 - y_i)\underline{f_i} \qquad \forall i \in P, \tag{4.40}$$

$$G_i \geq f_i - (1 - y_i)\overline{f_i} \qquad \forall i \in P, \tag{4.41}$$

The purpose of introducing DARP-UP model is to improve service quality by taking into account choice model. Note that the proposed DARP-UP model is a MILP which can be solved by commercial optimisation software.

4.2.2 Numerical Results

In this section, the DARP-UP model is implemented using Cordeau (2006)'s benchmark dataset. Parameters β_T, β_S and β_F are generated based on a cost and benefit analysis conducted in Aus-

Instances	Classic DARP				DARP-UP			
	Profit ($)	Customers serviced ($\sum_{i \in P} y_i$)	Computational time (s)	Optimality gap	Profit ($)	Customers serviced ($\sum_{i \in P} y_i$)	Computational time (s)	Optimality gap
a2-16	2611.50*	16	2.422	-	1511.82*	10	0.203	-
a2-20	3310.33*	20	9.735	-	1569.52*	10	0.203	-
a2-24	3971.88*	24	30.563	-	2449.75*	16	1.766	-
a3-18	2997.77*	18	271.953	-	1575.53*	10	0.969	-
a3-24	4106.38*	24	1323.063	-	1519.42*	10	1.516	-
a3-30	5002.60	30	10801.438	2.39%	2726.77*	17	311.828	-
a3-36	5973.39	36	10200.780	4.56%	3119.47*	20	107.734	-
a4-16	2634.65*	16	1792.625	6.49%	1739.52*	11	20.500	-
a4-24	4044.89	24	10801.780	2.66%	2014.40*	13	99.875	-
a4-32	5388.16	32	10800.734	5.93%	2372.24*	15	108.688	-

Flat fare is set to $f_i = \$2.00 \quad \forall i \in P$
Time limit is 2h
Instances marked with * are solved to optimal within time limit

Table 4.2: Optimal Profit and Computational Time for All Instances

tralia and New Zealand Litman (2009). Note that the monetary unit '$' in this section represents Australian Dollar (AUD). This thesis implements DARP-UP using CPLEX's Python API v12.8 on a personal computer with 16GB of RAM and an Intel i7 processor at 2.9GHz. Some of the pre-processing user cuts by Cordeau (2006) are applied.

Comparison Between Classic DARP and DARP-UP

To explore the effect of choice model, the model was run on the same set of instances with and without the users' preference constraint (4.5), and refer them as classic DARP model and DARP-UP model respectively. Fare is set to a flat rate of two dollars for all users for comparison purposes. The corresponding profit, serviced customers, computational time and optimality gap are shown in Table 4.2. In classic DARP, dial-a-ride operators always design routes with all requests included without taking into account the assessment of users' preferences. Counting all requests as guaranteed customers does not consider the possibility of cancellation or no-show, even though leads to higher profits, is not a realistic assumption. Furthermore, it is worth mentioning that the difference in computational time between solving DARP-UP and classic DARP is neglectable despite the increased number of variables (y_i). This is because with the choice model constraints, not all users are serviced by dial-a-ride system. Therefore, DARP-UP ends up solving a problem with fewer nodes comparing to classic DARP when using the same instance, and the saved computational

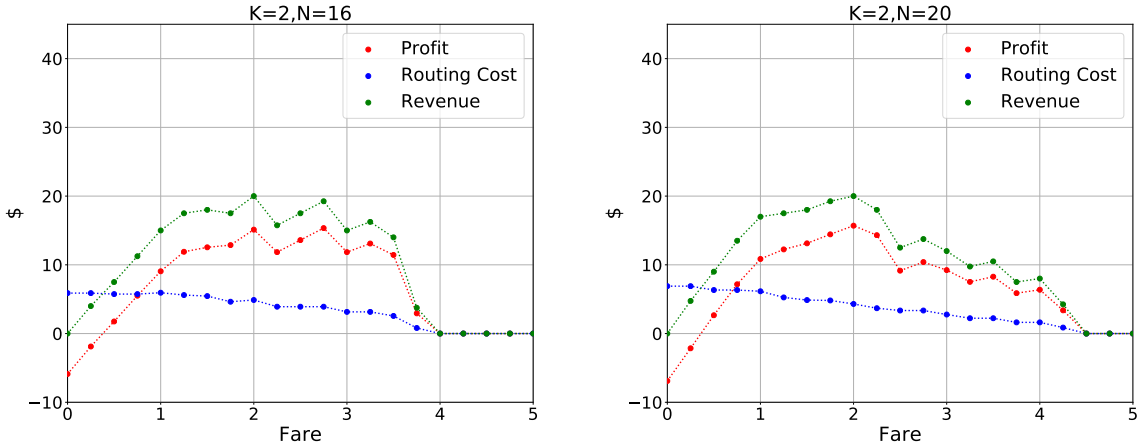
Fare ($AUD)	Classic DARP				DARP-UP			
	Profit ($)	Revenue ($)	Customers serviced	Routing Cost ($)	Profit ($)	Revenue ($)	Customers serviced	Routing Cost ($)
0.00	-5.88	0.00	16	588.50	-5.88	0.00	16	588.50
0.25	-1.88	4.00	16	588.50	-1.88	4.00	16	588.50
0.50	2.11	8.00	16	588.50	1.75	7.50	15	574.22
0.75	6.11	12.00	16	588.50	5.50	11.25	15	574.22
1.00	10.11	16.00	16	588.50	9.05	15.00	15	594.08
1.25	14.11	20.00	16	588.50	11.89	17.50	14	560.20
1.50	18.11	24.00	16	588.50	12.54	18.00	12	545.16
1.75	22.11	28.00	16	588.50	12.86	17.50	10	463.11
2.00	26.11	32.00	16	588.50	15.11	20.00	10	488.18
2.25	30.11	36.00	16	588.50	11.84	15.75	7	390.55
2.50	34.11	40.00	16	588.50	13.59	17.50	7	390.55
2.75	38.11	44.00	16	588.50	15.34	19.25	7	390.55
3.00	42.11	48.00	16	588.50	11.84	15.00	5	315.26
3.25	46.11	52.00	16	588.50	13.09	16.25	5	315.26
3.50	50.11	56.00	16	588.50	11.43	14.00	4	256.24
3.75	54.11	60.00	16	588.50	2.94	3.75	1	80.97
4.00	58.11	64.00	16	588.50	0.00	0.00	0	0.00
4.25	62.11	68.00	16	588.50	0.00	0.00	0	0.00
4.50	66.11	72.00	16	588.50	0.00	0.00	0	0.00

Table 4.3: Comparison Between Classic DARP and DARP-UP for Instance a2-16

time compensates the burden caused by extra variables.

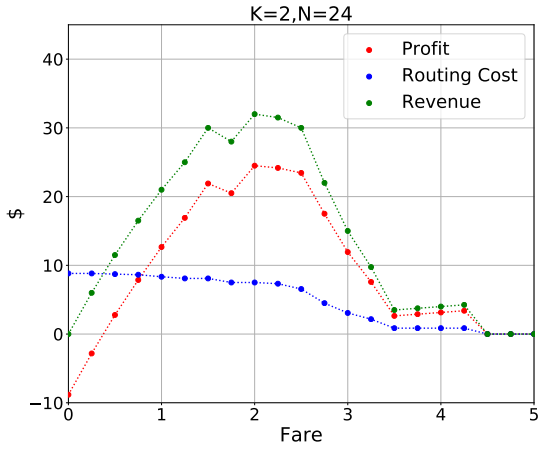
For each instance, classic DARP and DARP-UP were solved iteratively with a series of flat fare values within [$0,$5] with an interval of $0.25. Table 4.3 shows profit, routing cost and revenue for instance a2-16 with different fare values. Classic DARP models are designed to accommodate all requests. Therefore it is insensitive towards changes in fare. On the contrary, DARP-UP takes fare as an incentive, and the number of serviced customers falls when the price of shared mobility services becomes too expensive, indicating users switch to other optional transportation modes. Dial-a-ride operators therefore exclude some users when designing the optimal routes, which will maximize utilities of on-board customers by eliminating detours caused by no-shows and cancellations, and improve service quality.

Note that the discussion above is based on the default model where tolerance parameter $\epsilon = 1$. However, if ϵ is set to 0 instead, (4.5) will always hold despite the relationship between U_i and $\hat{U}_i$. In this case, the choice model constraint (4.5) is relaxed, and the number of customers serviced solely depends on the trade-off between growing revenue and increasing routing cost. Therefore, if

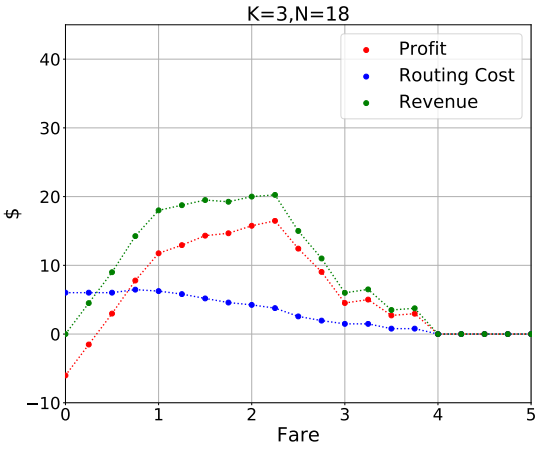


(a) Instance a2-16

(b) Instance a2-20

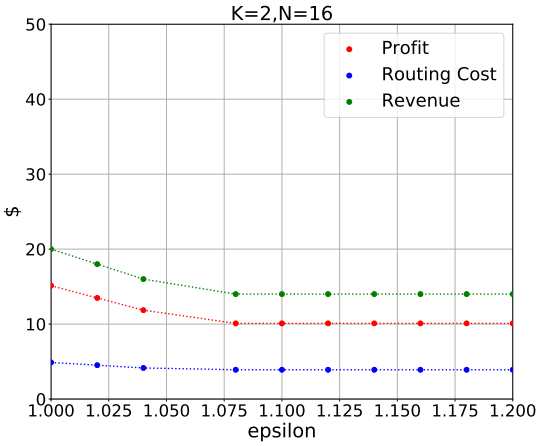


(c) Instance a2-24

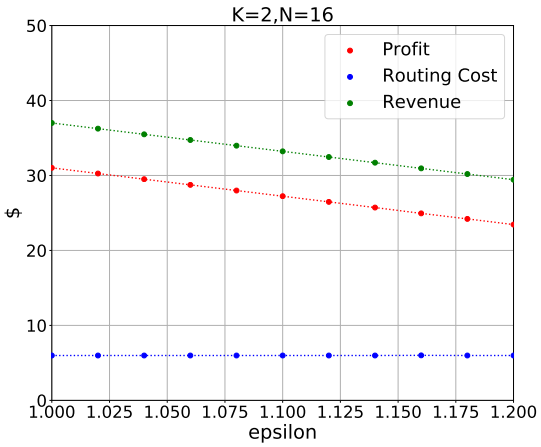


(d) Instance a3-18

Figure 4.1: Sensitivity Analysis Results on Fare on the First Four Instances



(a) Flat fare $f_i = \$2.00 \quad \forall i \in P$



(b) Variable fare f_i

Figure 4.2: Tolerance Analysis on Instance a2-16

Instances	Flat fare $f_i = \$2.00 \quad \forall i \in P$			Variable fare f_i		
	Profit ($)	Customers serviced ($\sum_{i \in P} y_i$)	Computational time (s)	Profit ($)	Customers serviced ($\sum_{i \in P} y_i$)	Computational time (s)
a2-16	1511.82*	10	0.234	3101.73*	16	2.016
a2-20	1569.52*	10	0.187	3395.85*	19	3.891
a2-24	2449.75*	16	5.063	4557.32*	24	75.326
a3-18	1575.53*	10	1.907	3186.65*	18	255.937

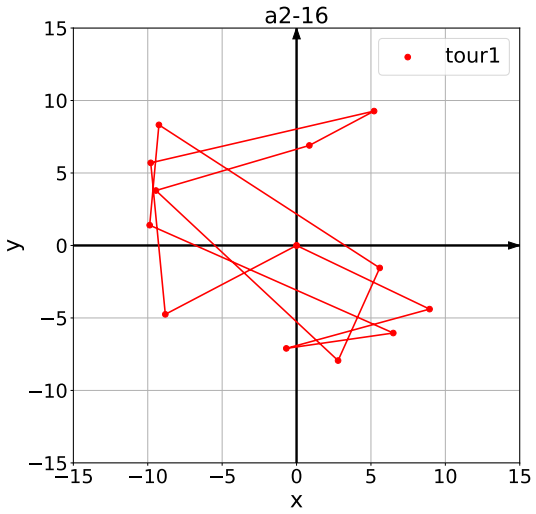
Time limit is set to 6h
Instances marked with * are solved to optimality within the time limit

Table 4.4: Optimal Profit and Computational Time for the First 4 Instances

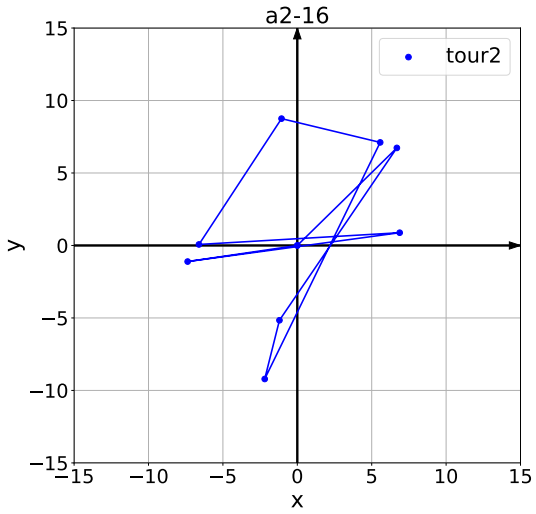
the fare value is high enough, DARP-UP will converge to classic DARP and service all requests. On the other hand, if the fare is too low, the marginal revenue by servicing an additional customer may be offset by the associated routing costs, hence the number of serviced customers will drop.

Fare Variation

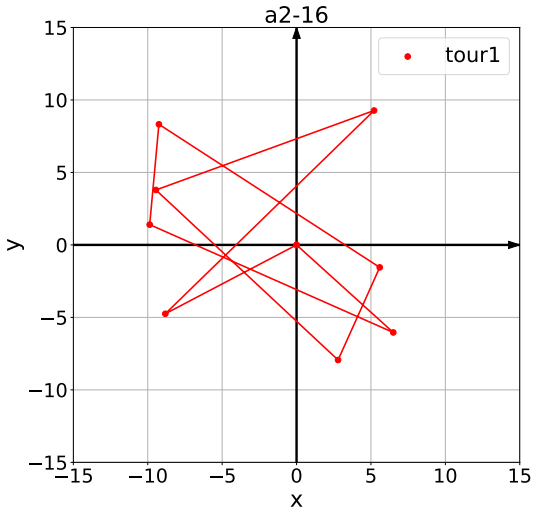
Figure 4.1 presents how profit, routing cost and revenue change in the first four instances (i.e., instance a2-16, a2-20, a2-24 and a3-18) with the increase in fare up to $5.0. Note that experiments in Figure 4.1 are carried out with a flat fare scheme, where an identical fee is charged with all customers. It is observed that although fare clearly has a significant impact on profit, profit does



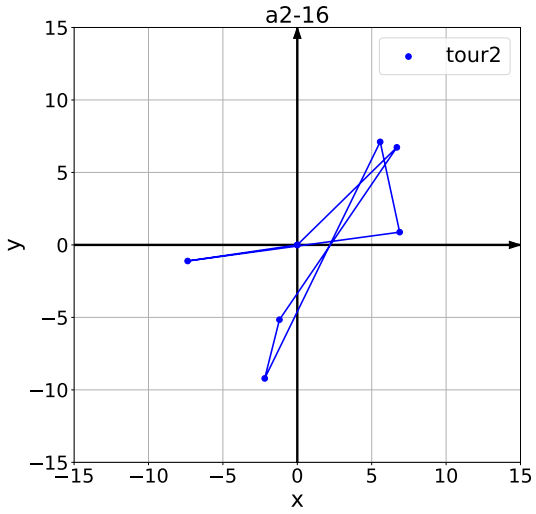
(a) Tour of vehicle 1 with $\epsilon = 1.00$



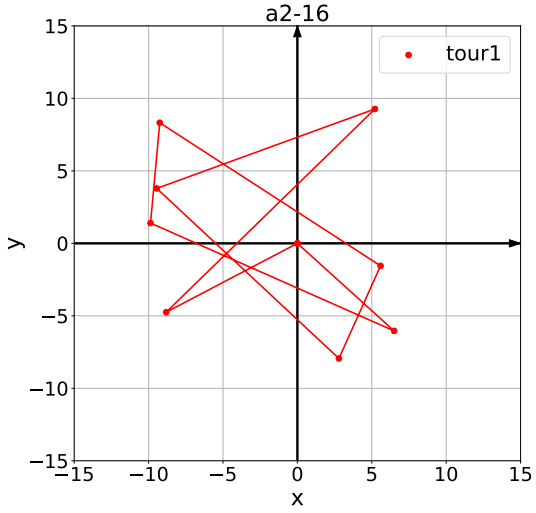
(b) Tour of vehicle 2 with $\epsilon = 1.00$



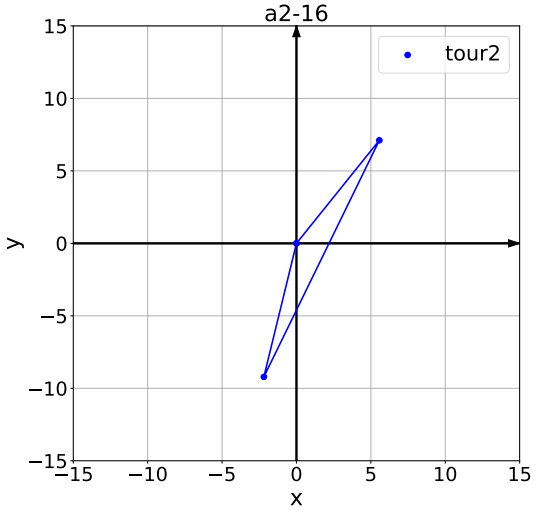
(c) Tour of vehicle 1 with $\epsilon = 1.20$



(d) Tour of vehicle 2 with $\epsilon = 1.20$

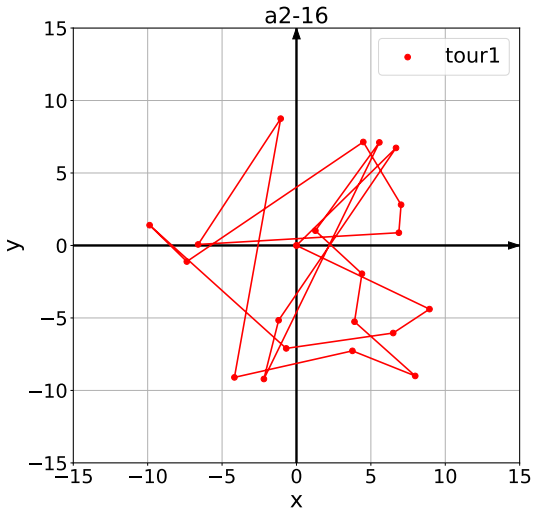


(e) Tour of vehicle 1 with $\epsilon = 1.40$

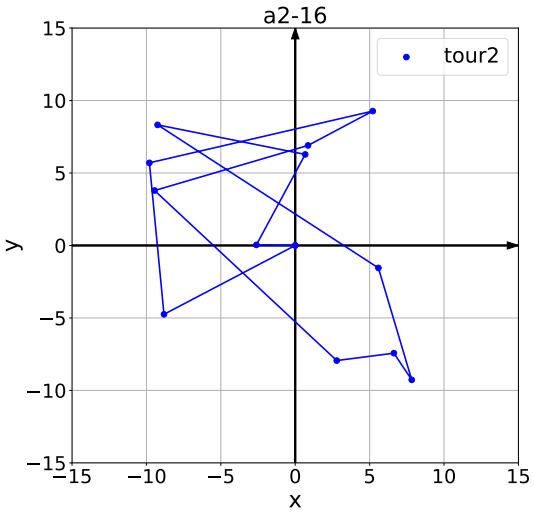


(f) Tour of vehicle 2 with $\epsilon = 1.40$

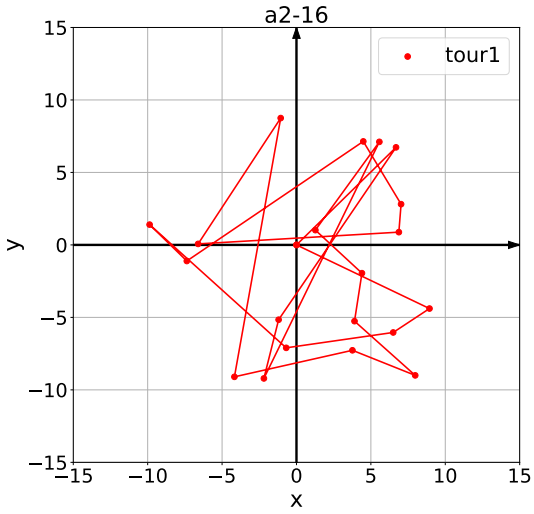
Figure 4.3: Optimal Routes for Flat fare $f_i = F \quad \forall i \in P$



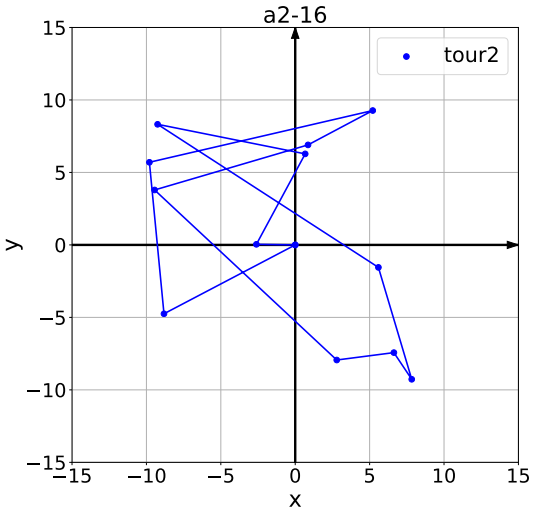
(a) Tour of vehicle 1 with $\epsilon = 1.00$



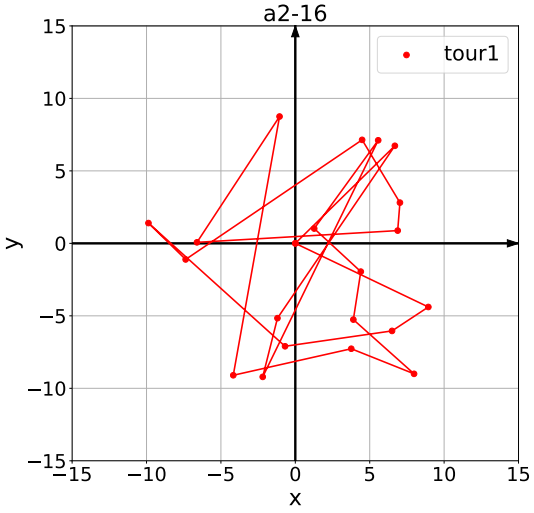
(b) Tour of vehicle 2 with $\epsilon = 1.00$



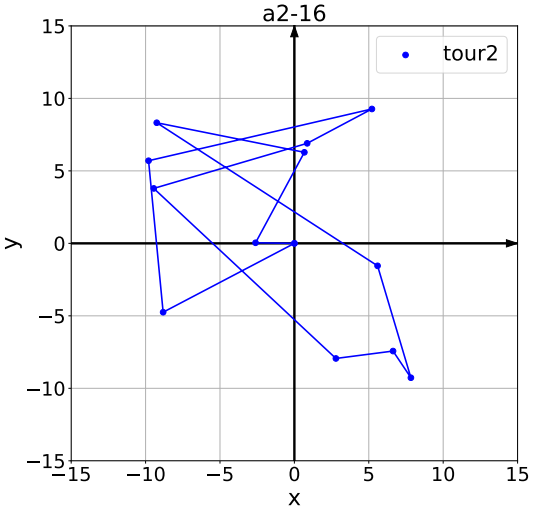
(c) Tour of vehicle 1 with $\epsilon = 1.20$



(d) Tour of vehicle 2 with $\epsilon = 1.20$



(e) Tour of vehicle 1 with $\epsilon = 1.40$



(f) Tour of vehicle 2 with $\epsilon = 1.40$

Figure 4.4: Optimal Routes for Variable Fare f_i

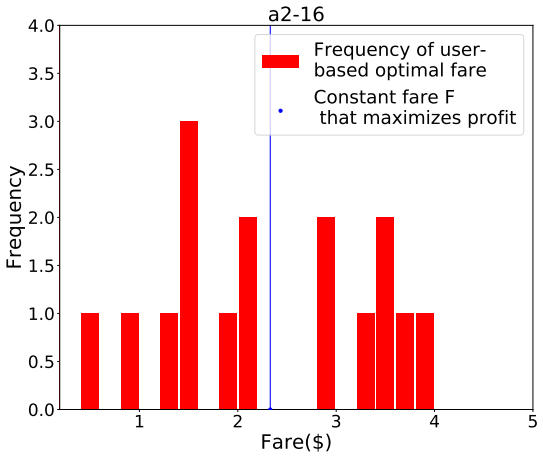
not increase monotonically with fare. In this section, this thesis explores user-based variable fare f_i optimises system profit and affects system performances.

The optimal profit and computational time for each instance with flat fare and customised fare are displayed in Table 4.4. From the results it is observed that introducing individualised fares boost up the system profit, while it is rather computationally demanding at the same time. Hence scalable solution methods are required. The distribution of user-based optimal fare f_i is displayed Figure 4.5 for the first four instances (i.e., instances a2-16, a2-20, a2-24 and a3-18), with the flat fare values where profit peaked in Figure 4.1 marked by a vertical line. The bar chart indicates that the distribution of individualised optimal fare varies substantially, which highlights the benefits of customized fares for each user in order to maximise profit overall. Given that each user has their own preferences and personal conditions. Variable fare can be viewed as a user-based incentive that can help in nudging users' behaviour Rey et al. (2016), and charging a universally fixed fare may be inefficient with a high opportunity cost.

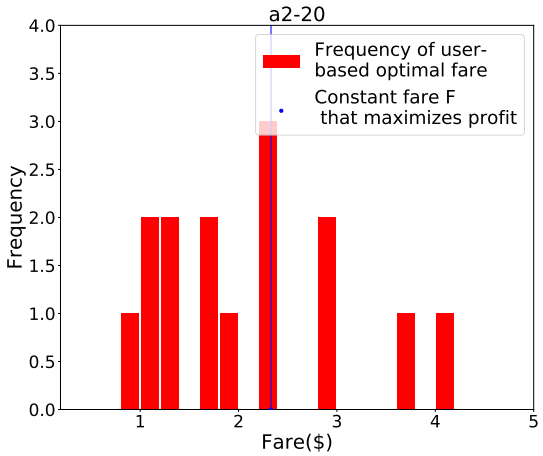
Tolerance Analysis

In practical applications, dial-a-ride operators may have different tolerances towards users' relative utility. A tolerance analysis is carried out on DARP-UP with tolerance parameter ϵ set to different values representing various behaviours of dial-a-ride operators, risk-taking (with a lower ϵ) or risk-averse (with a higher ϵ).

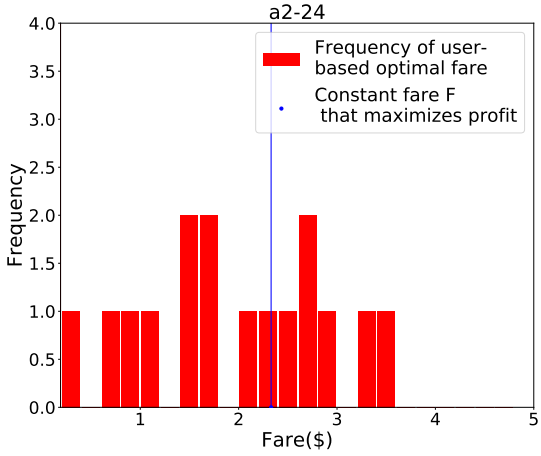
In the tolerance analysis, this thesis explores how the increase in ϵ affect DARP's routing decisions with different fare formulations. For flat fare setting, the amount of customers serviced by shared mobility system drops with the increase of ϵ, given that an extra ϵ can be considered as a penalty on dial-a-ride services since the utility for shared mobility services has to surpass its counterpart for a certain amount $(\epsilon - 1)\hat{U}_i$ for being chosen by user i. On the other hand, DARP-UP with variable fare formulation sees a better performance in tolerance and maintaining the same routing decisions with the same tolerance parameter ϵ. Take instance a2-16 as an example, Figure 4.2 shows how profit, routing cost and revenue are influenced by an increasing ϵ parameter up to 120% with the formulation of constant and variable fare, respectively. Especially in Figure



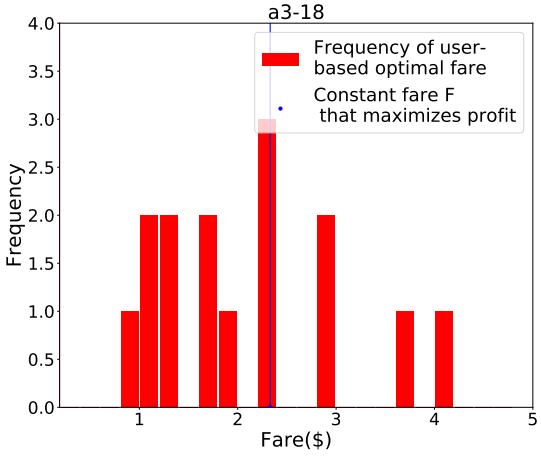
(a) Instance a2-16



(b) Instance a2-20



(c) Instance a2-24



(d) Instance a3-18

Figure 4.5: Distribution of Optimal User-based Fare for the First 4 Instances

4.2(b), despite the inevitable drop in revenue and profit due to the adjustments made in optimal fare to satisfy constraint (4.5) with an increasing ϵ, the routing cost remained stable. Furthermore, Figure 4.3 and Figure 4.4 display how the optimal routes change with different ϵ. In Figure 4.3, in DARP-UP with a flat fare setting, routing decisions vary significantly due to the decrease in number of customers. On the contrary, Figure 4.4 shows that the routing decisions remain the same for DARP-UP with variable fare formulation with ϵ up to 140%.

4.3 Stochastic Model: Chance-constrained DARP

In this section, a DARP model integrated with users' preferences from DRT operators' perspective is proposed, facilitating them to optimise their overall profit while maintaining service quality. In this model, users' utility and users' preferences are taken into account within a dial-a-ride problem. This thesis proposes a Chance-Constrained Programming (CCP) (Charnes and Cooper, 1959) approach, which aims to identify users that are better off travelling by DRT services than using a reserve travel mode. The proposed CCP formulation incorporates operator's accept/reject decisions of users' requests. Only accepted users are served by the DRT operator, while the rest of the requests are rejected and left out of consideration of routing and scheduling. This CCP accept/reject mechanism is expected to guide service providers on a strategic planning level to avoid blindly accepting all requests at the cost of driving down system profit or the user experiences of on-board passengers.

This study also explores the design of revenue/fleet management and pricing differentiation (Talluri and Van Ryzin, 2004) under the proposed chance-constrained DARP (CC-DARP) model. Ridership response studies often define price differentiation policies targeting different segments (classes) based on user groups and trip characteristics (Cervero, 1990). Users from different segments (classes) of age, income, auto access and trip purpose tend to respond with various sensitivities to fare changes. Accordingly, multiple user classes are introduced in the chance-constrained DARP model. The proposed multi-class, chance-constrained DARP is formulated as a Mixed-Integer Linear Program (MILP). By embedding class-based formulation into various fare structures, the proposed model also helps design and evaluate pricing strategies and revenue management.

4.3.1 Utility Functions and Chance Constraints

In this section, the stochastic DARP model again follows the formulation of Cordeau (2006) for the base classic DARP model. A summary of the mathematical notations used throughout the section is provided in Table 4.1.

For simplicity purposes, two travel modes are considered for users to choose between a DRT service and the best option available to users other than DRT. In this discussion, it is assumed that the other travel mode to be a private travel option with its utility function defined accordingly. The representative utility for each travel mode are defined as functions of travel time, schedule delay and monetary cost (Dong et al., 2020).

For a request $i \in \mathcal{P}$, the representative utility for private travel is:

$$\hat{V}_i = -\beta_T t_{i,n+i} - \beta_F \hat{c}_i, \qquad \forall i \in \mathcal{P}. \tag{4.42}$$

For the DRT service, as the same as traditional DARP models, inbound and outbound user requests are distinguished, denoted by sets $\mathcal{IB}$ and $\mathcal{OB}$, respectively. Each user mark their trip as either an inbound (with a tight time window on departure) or an outbound (with a tight time window on arrival) trip when they send out their request. Note that $\mathcal{P} = \mathcal{IB} \cup \mathcal{OB}$ and $\mathcal{IB} \cap \mathcal{OB} = \emptyset$. The representative utility function for the DRT service are:

$$V_i = -\beta_F F - \beta_T L_i - \beta_S (B_i - e_i), \qquad \forall i \in \mathcal{IB}, \tag{4.43}$$

$$V_i = -\beta_F F - \beta_T L_i - \beta_S (l_{n+i} - B_{n+i}), \qquad \forall i \in \mathcal{OB}. \tag{4.44}$$

In utility functions (4.42)-(4.44), β_T and β_S are parameters that convert travel time and schedule delay into monetary units. For private travel, it is assumed that travel time $t_{i,n+i}$ is fixed and that schedule delay is null. For the DRT service, travel time (ride time) is represented by variable L_i, which depends on the route serving user i. Schedule delay for DRT is defined based on trip types. For inbound trips (trips with a tight time window imposed on departure time), schedule delay is defined as waiting time at pickup locations $B_i - e_i$. Whereas for outbound trips, schedule delay is

defined as the amount of time that user i arrives earlier than expected $l_{n+i} - B_{n+i}$, assuming the upper bound of time window l_{n+i} is the preferred arrival time. Let β_F be the coefficient for monetary travel cost, which is the fare F for DRT service and cost $\hat{c}_i$ (including gas, maintenance, registration, etc.) for private travel mode. β_F consolidates among various currencies to a unified monetary value.

In this model, it is assumed that the difference in utility $\Delta U_i = (\hat{V}_i - V_i) + \epsilon$ is a random variable following a logistic distribution $\Delta U_i \sim (\overline{\Delta U_i}, s)$; wherein $\overline{\Delta U_i}$ is the deterministic utility gap defined as follows:

$$\overline{\Delta U_i} = \hat{V}_i - V_i = \begin{cases} \beta_T(L_i - t_{i,n+i}) + \beta_S(B_i - e_i) + \beta_F(F - \hat{c}_i), & \text{if } i \in \mathcal{IB}, \\ \beta_T(L_i - t_{i,n+i}) + \beta_S(l_{n+i} - B_{n+i}) + \beta_F(F - \hat{c}_i), & \text{if } i \in \mathcal{OB}, \end{cases} \tag{4.45}$$

and s is the scale parameter proportional to its standard deviation. Observe that $\overline{\Delta U_i}$ also serves as a decision variable in the proposed CC-DARP, since it is a function of variables L_i and B_i.

The sign of ΔU_i indicates the preference of user i: if $\Delta U_i \leq 0$, then in the long-run, user i is expected to experience a higher utility using DRT than the private travel alternative. To capture users' preferences in the long-run, the following chance constraints are introduced to ensure that $\Delta U_i \leq 0$, $\forall i \in \mathcal{P}$ holds with a confidence level of p:

$$\Pr(\Delta U_i \leq 0) \geq p, \qquad \forall i \in \mathcal{P}. \tag{4.46}$$

Let $F_{\Delta U_i}$ be the cumulative distribution function of random variableΔU_i:

$$\Pr(\Delta U_i \leq 0) = F_{\Delta U_i}(0), \qquad \forall i \in \mathcal{P}. \tag{4.47}$$

Combined with (4.47), for each $i \in \mathcal{P}$, chance constraints (4.46) can be written as:

$$F_{\Delta U_i}(0) \geq p,$$

$$\Leftrightarrow \qquad 0 \geq Q_{\Delta U_i}(p),$$
$$\Leftrightarrow \qquad 0 \geq \overline{\Delta U_i} + s \ln\left(\frac{p}{1-p}\right), \tag{4.48}$$

where $Q_{\Delta U_i}(p)$ is the inverse cumulative distribution function (quantile function) of the logistic distribution $\Delta U_i \sim (\overline{\Delta U_i}, s)$. Combining (4.48) and utility functions (4.45):

$$\beta_T(L_i - t_{i,n+i}) + \beta_S(B_i - e_i) + \beta_F(F - \hat{c}_i) \leq -s \ln\left(\frac{p}{1-p}\right), \qquad \forall i \in \mathcal{IB}, \tag{4.49}$$
$$\beta_T(L_i - t_{i,n+i}) + \beta_S(l_{n+i} - B_{n+i}) + \beta_F(F - \hat{c}_i) \leq -s \ln\left(\frac{p}{1-p}\right), \qquad \forall i \in \mathcal{OB}. \tag{4.50}$$

The goal of the proposed chance-constrained formulation is to ensure that the DRT service operator only serves users' which are better-off using DRT services than their private travel alternative with a confidence level of p. To incorporate the proposed chance constraints (4.49) and (4.50) in the formulation, a binary variable $y_i \in \{0, 1\}$ is introduced, representing if the DRT service operator's chooses to serve user request $i \in \mathcal{P}$ (1) or not (0). Let W_i be a large constant (the value of this constant is discussed later). Using variable y_i, Eqs. (4.49) and (4.50) are reformulated as on/off constraints as follows:

$$\beta_T(L_i - t_{i,n+i}) + \beta_S(B_i - e_i) + \beta_F(F - \hat{c}_i) - (1 - y_i)W_i \leq -s \ln\left(\frac{p}{1-p}\right), \qquad \forall i \in \mathcal{IB}, \tag{4.51}$$

$$\beta_T(L_i - t_{i,n+i}) + \beta_S(l_{n+i} - B_{n+i}) + \beta_F(F - \hat{c}_i) - (1 - y_i)W_i \leq -s \ln\left(\frac{p}{1-p}\right), \qquad \forall i \in \mathcal{OB}. \tag{4.52}$$

To link user requests accept/reject decisions with routing decision variables x_{ij}^k, the classical DARP constraints requesting that all users are served are reformulated as:

$$\sum_{j \in \mathcal{N}} \sum_{k \in \mathcal{K}} x_{ij}^k = y_i, \qquad \forall i \in \mathcal{P}. \tag{4.53}$$

The proposed chance-constrained formulation links user requests accept/decisions to the random

variable ΔU_i representing the relative utility of user i. This mechanism is sensitive to the required confidence level p and the scale parameter s of the assumed logistic distribution. The impact of these parameters on model performance are examined via sensitivity analyses in the numerical experiments.

4.3.2 Class-based Fare Structures

In the discussion of Section 4.3.1, it is assumed that all users are homogeneous, and that a flat fare F is charged for the DRT service. In this section, users are grouped into various classes by their socio-demographic characteristics (age, income, car ownership, etc) and introduce a class-based pricing system with various potential fare structures for DRT services, including distance-based fare and zone-based fare.

Let $\mathcal{M} = \{1, 2, ..., M\}$ be a set of classes. Each request i can be categorised into one of the mutually exclusive classes of requests $\mathcal{P}_m$ (i.e. $\cup_{m\in\mathcal{M}}\mathcal{P}_m = \mathcal{P}$). Furthermore, utility functions are redefined on a class basis with class-specific parameters $\beta_F^m, \beta_T^m, \beta_S^m$ and user-based fare f_{im}. Specifically, the deterministic part of the utility functions (4.42)-(4.44) is reformulated as follows.

For private travel:

$$\hat{V}_i = -\beta_T^m t_{i,n+i} - \beta_F^m \hat{c}_i, \qquad \forall m \in \mathcal{M}, \forall i \in \mathcal{P}_m. \tag{4.54}$$

For DRT service:

$$V_i = -\beta_F^m f_{im} - \beta_T^m L_i - \beta_S^m (B_i - e_i), \qquad \forall m \in \mathcal{M}, \forall i \in \mathcal{IB} \cap \mathcal{P}_m, \tag{4.55}$$

$$V_i = -\beta_F^m f_{im} - \beta_T^m L_i - \beta_S^m (l_{n+i} - B_{n+i}), \qquad \forall m \in \mathcal{M}, \forall i \in \mathcal{OB} \cap \mathcal{P}_m. \tag{4.56}$$

Accordingly, this model assumes that the scale parameter of the random variable is also class-dependent and the random variable is rewritten as $\Delta U_i \sim (\overline{\Delta U_i}, s_m)$, for each request $i \in \mathcal{P}_m$.

Recall that $\overline{\Delta U_i} = \hat{V}_i - V_i$. Observe that $\hat{V}_i$ is a constant, and that V_i is a real bounded variable which upper and lower bounds, denoted $\overline{V}_i$ and $\underline{V}_i$ respectively, can be determined by taking the maximum and minimum values of the right-hand sides of Eqs. (4.55)-(4.56). Details on the calculation of these bounds can be found in Dong et al. (2020). Let $W_{im} = \hat{V}_i - \underline{V}_i + s_m \ln\left(\frac{p_m}{1-p_m}\right)$. The chance constraints (4.51) and (4.52) are redefined with a specified confidence level p_m assigned to each class m and written compactly as:

$$0 \geq \hat{V}_i - V_i + s_m \ln\left(\frac{p_m}{1-p_m}\right) - (1-y_i)W_{im}, \qquad \forall m \in \mathcal{M}, \forall i \in \mathcal{P}_m. \tag{4.57}$$

Two fare structures that can be used to determine user-based trip fares f_{im} in DRT services are presented in the following subsections.

Distance-based Fare Structure

One of the most common fare structures for public transport is a distance-based fare structure, as widely applied in taxi services. Under distance-based fare structure, customized fares are charged to passengers based on the distances of their trips. Integrating user classes into this fare structure, the fare f_i^D charged to user i is defined as follows:

$$f_i^D = \alpha_m Dist_{i,n+i}, \qquad \forall m \in \mathcal{M}, \forall i \in \mathcal{P}_m. \tag{4.58}$$

In (4.58), $Dist_{i,n+i}$ represents the direct travel distance between the pickup and drop off locations of request i, while α_m is a class-based cost parameter that converts distances into monetary units.

Zone-based Fare Structure

Zone-based or zonal-based fare structures divide the service area into several districts (zones), and the fare for request i is calculated according to which zones the pickup location $i \in \mathcal{P}$ and drop-off location $(i+n) \in \mathcal{D}$ lie in. Let $\mathcal{Z} = \{1, 2, ..., Z\}$ be a set of geographical zones. The pickup and drop off locations of request i can be each categorised into one of the mutually exclusive zone $\mathcal{P}_z$.

Assume that for request i, its pickup location i lies in zone $\mathcal{P}_{z_p}$ (i.e. $i \in \mathcal{P}_{z_p}$), and its drop-off location $i+n$ belongs to zone $\mathcal{P}_{z_d}$ (i.e. $(i+n) \in \mathcal{P}_{z_d}$). The zone-based fare value of user request i, f_i^Z, is then calculated as follows:

$$f_i^Z = \theta_{z_p,z_d} f_m, \qquad \forall m \in \mathcal{M}, \forall z_p, z_d \in \mathcal{Z}, \forall i \in \mathcal{P}_m \cap \mathcal{P}_{z_p}, \forall (i+n) \in \mathcal{P}_{z_d}. \tag{4.59}$$

In (4.59), f_m is a base fare for user class m, whereas θ_{z_p,z_d} is a weight parameter considering the corresponding zones z_p and z_d that include pickup location i and drop off location $(n+i)$ of request i.

4.3.3 Chance-constrained DARP (CC-DARP) Formulation

The proposed class-based, chance-constrained formulation is then incorporated with the classic DARP model. To account for user request accept/reject decisions, this thesis proposes a profit maximization formulation for the objective function:

$$\max z = \sum_{m \in \mathcal{M}} \sum_{i \in \mathcal{P}_m} f_{im} q_{im} y_i - \sum_{k \in \mathcal{K}} \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} c_{ij} x_{ij}^k \tag{4.60}$$

The first term of the objective function (4.60) represents the total revenue of the DRT service operator and the second term represents the total routing cost of the operations. Note that q_{im} represents the load of request i in class m, considering different pricing strategies on the number of passengers could be applied to different classes. Note that fare f_{im} is charged per passenger rather than per trip. With users' preferences and chance constraints incorporated into the classic three-index DARP formulation, the mixed-integer linear programming formulation for the CC-DARP model is summarised in Model 3.

Model 3.

$$\max z = \sum_{m \in \mathcal{M}} \sum_{i \in \mathcal{P}_m} f_{im} q_{im} y_i - \sum_{k \in \mathcal{K}} \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} c_{ij} x_{ij}^k, \tag{4.61a}$$

subject to:

Classic DARP constraints:

$$\sum_{j\in\mathcal{N}} x_{ij}^k - \sum_{j\in\mathcal{N}} x_{n+i,j}^k = 0, \qquad \forall i \in \mathcal{P}, k \in \mathcal{K}, \tag{4.61b}$$

$$\sum_{j\in\mathcal{N}} x_{0j}^k = 1, \qquad \forall k \in \mathcal{K}, \tag{4.61c}$$

$$\sum_{i\in\mathcal{N}} x_{i,2n+1}^k = 1, \qquad \forall k \in \mathcal{K}, \tag{4.61d}$$

$$\sum_{j\in\mathcal{N}} x_{ji}^k - \sum_{j\in\mathcal{N}} x_{ij}^k = 0, \qquad \forall i \in \mathcal{P}\cup\mathcal{D}, k \in \mathcal{K}, \tag{4.61e}$$

$$B_j \geq (B_0^k + d_0 + t_{0j}) x_{0j}^k, \qquad \forall j \in \mathcal{N}, k \in \mathcal{K}, \tag{4.61f}$$

$$B_j \geq (B_i + d_i + t_{ij}) \sum_{k\in\mathcal{K}} x_{ij}^k, \qquad \forall i, j \in \mathcal{N}, \tag{4.61g}$$

$$B_{2n+1}^k \geq (B_i + d_i + t_{i,2n+1}) x_{i,2n+1}^k, \qquad \forall i \in \mathcal{N}, k \in \mathcal{K}, \tag{4.61h}$$

$$Q_j \geq (Q_0^k + q_{jm}) x_{0j}^k, \qquad \forall m \in \mathcal{M}, j \in \mathcal{N}_m, k \in \mathcal{K}, \tag{4.61i}$$

$$Q_j \geq (Q_i + q_{jm}) \sum_{k\in\mathcal{K}} x_{ij}^k, \qquad \forall i \in \mathcal{N}, m \in \mathcal{M}, j \in \mathcal{N}_m, \tag{4.61j}$$

$$Q_{2n+1}^k \geq (Q_i + q_{2n+1,m}) x_{i,2n+1}^k, \qquad \forall i \in \mathcal{N}, k \in \mathcal{K}, \tag{4.61k}$$

$$L_i = B_{n+i} - (B_i + d_i), \qquad \forall i \in \mathcal{P}, \tag{4.61l}$$

$$B_{2n+1}^k - B_0^k \leq T, \qquad \forall k \in \mathcal{K}, \tag{4.61m}$$

$$e_i \leq B_i \leq l_i, \qquad \forall i \in \mathcal{P}\cup\mathcal{D}, \tag{4.61n}$$

$$t_{i,n+i} \leq L_i \leq L, \qquad \forall i \in \mathcal{P}, \tag{4.61o}$$

$$\max\{0, q_{im}\} \leq Q_i \leq \min\{Q, Q + q_{im}\}, \qquad \forall m \in \mathcal{M}, i \in \mathcal{N}_m, k \in \mathcal{K}, \tag{4.61p}$$

$$x_{ij}^k \in \{0,1\}, \qquad \forall i \in \mathcal{N}, j \in \mathcal{N}, k \in \mathcal{K}, \tag{4.61q}$$

$$y_i \in \{0,1\}, \qquad \forall i \in \mathcal{P}, \tag{4.61r}$$

$$\underline{V}_i \leq V_i \leq \overline{V}_i, \qquad \forall i \in \mathcal{P}, \tag{4.61s}$$

Utility Functions: (4.55) – (4.56),

Chance Constraints: (4.57),

Accept/Reject: (4.53).

Aligned with Cordeau's classic DARP model (Cordeau, 2006), constraints (4.61b) guarantees that pick-up and drop-off nodes for each request are visited by the same vehicle, whereas constraints (4.61c) and (4.61d) ensure that each route starts at the origin depot and ends at the destination depot. Constraints (4.61e), (4.61f)–(4.61h) and (4.61i)–(4.61k) guarantee flow conservation, time and load consistency respectively. Constraint (4.61l) defines the ride time of each user, which is bounded by (4.61o). The duration of routes are bounded by constraints (4.61m), while (4.61n) and (4.61p) are the time window and capacity constraints respectively. Readers are referred to Cordeau (2006) for the linearization of constraints (4.61f)–(4.61k).

4.4 Local-Seach Based Solution Method

A local search based descent heuristic (LS-H) algorithm is proposed to solve the CC-DARP. For simplicity, this thesis assumes the passengers from one request cannot be split and serviced by different vehicles. Starting with an initial solution generated by a customized insertion heuristic based on Jaw et al. (1986)'s sequential insertion heuristics (Section 4.4.1), the LS-H algorithm iterates between executing local search on the search space of user selection and the search space of routing optimisation. The algorithm terminates when no better solution can be found.

4.4.1 Construction of an Initial Solution

Jaw et al. (1986) proposed a sequential insertion heuristic, which is deemed to be a decent starting point for generating initial solutions with a few modifications applied. Requests are indexed and sorted in an increasing order according to their earliest departure time. Note that for an outbound trip i with a tight time window on arrival time (i.e. $e_{n+i} \leq B_{n+i} \leq l_{n+i}$), a pickup time window

can be inferred using the direct travel time $t_{i,n+i}$ and the maximum ride time L:

$$B_i \leq l_{n+i} - t_{i,n+i}, \qquad \forall i \in \mathcal{OB},$$
$$B_i \geq e_{n+i} - L, \qquad \forall i \in \mathcal{OB}.$$

With specified information on vehicle capacity and fleet, the algorithm processes requests sequentially, inserting one request into vehicles' work-schedule at a time, checking all feasible insertions, and choosing the insertion where its additional cost is minimised. Violations of chance constraints (4.57) are checked after each insertion, and it must be satisfied for an insertion to be labelled 'feasible'. Evidently, some requests might not be inserted in this step, given the limited number of vehicles. These temporarily rejected requests are saved in a pool and for possible later insertion within the following local search step.

Although the literature shows that the quality of initial results does not have a significant influence on final results, structuring a good initial solution can still reduce computational costs (Xiang et al., 2006). In this regard, adjustments are made in Jaw's insertion heuristic. To avoid situations where more profitable requests could not be inserted in a later stage, the sorting rules are adjusted. Intuitively, requests with relatively higher expected profits should be inserted first. Breaking down the profitability of each request by looking into constraints (4.57) and the utility functions in the proposed CC-DARP model, two factors are identified that determine profitability: direct trip distance and decentralisation. To take into account decentralisation, (Diana and Dessouky, 2004) proposed a decentralisation index D_i, which can be used to the algorithm as follows:

$$D_i = \frac{\sum\limits_{j \in \mathcal{N} \setminus \{i, n+i\}} t_{ij} + \sum\limits_{j \in \mathcal{N} \setminus \{i, n+i\}} t_{n+i,j}}{\sum\limits_{i' \in \mathcal{P}} \sum\limits_{j \in \mathcal{N} \setminus \{i', n+i'\}} t_{i'j} + \sum\limits_{i' \in \mathcal{P}} \sum\limits_{j \in \mathcal{N} \setminus \{i', n+i'\}} t_{n+i',j}}, \qquad \forall i \in \mathcal{P}.$$

In this way, the decentralisation of request i from both the depot and other requests are taken into account. The higher D_i is, the more decentralised request i is. Therefore, requests with a higher decentralisation index D_i should be inserted in priority, given that requests that are more decentralised can be more difficult to be inserted later on.

Similarly, to take into consideration the impact of direct trip length, a direct travel time index TT_i is defined as:

$$TT_i = \frac{t_{i,n+i}}{\sum\limits_{i' \in \mathcal{P}} t_{i',n+i'}}, \qquad \forall i \in \mathcal{P}.$$

Requests with smaller direct travel time index TT_i should be inserted with priority since these requests are more likely to be profitable according to (4.45) and harder to be inserted successfully at a later stage. To combine the impact of these two factors, a general index G_i is introduced as a weighted sum of D_i and TT_i, which utterly determines the adjustments of sequential sorting indices by increasing G_i values (4.62):

$$G_i = \omega(1 - D_i) + (1 - \omega)TT_i. \tag{4.62}$$

The greater the parameter $\omega \in (0, 1)$ is, the more weight is put on the influence of decentralisation. After the initial sorting by e_i, each pair of consecutive request $(i-1)$ and i is checked again according to the default sequential ranking order. The two requests get swapped if the following inequality is verified:

$$G_{i-1} - G_i \geq \delta. \tag{4.63}$$

Then the i-th and $(i-2)$-th requests are checked. The parameter δ also takes on calibrated values between 0 and 1. With a smaller δ, the impact of original time sequential sorting is emphasised.

In the numerical experiments, parameters ω and δ are calibrated using benchmarking instances for the DARP. Additionally, to refine the initialisation process of the routes, the routes are initialised by assigning the first $|\mathcal{K}|$ seed requests with smallest G_i, one to each empty vehicle. The seed requests are, considered by the standard, more difficult to insert later on and supposedly more profitable.

The remaining requests are inserted one by one following the adjusted insertion order $InOr$ in

Algorithm 3. Initial solution s_0 is instructed following Jaw et al. (1986)'s insertion method: Schedule blocks (when vehicles are active) and slack periods (when vehicles are idling) of each vehicle are created at initialisation, and updated after each successful insertion. When inserting a request i, the algorithm systematically scans all feasible insertions of the request into the existing work schedules of all vehicles, and choose the best insertion with minimum additional cost. At each attempt of inserting request i, an insertion is only labelled as feasible if it satisfies time window constraints (4.25), maximum ride time constraints (4.26), load constraints (4.27) and chance constraints (4.57) for all existing requests in the current work schedule. New schedule blocks can also be created when examining for feasible insertions. For a more detailed description of the insertion heuristic, readers are referred to Jaw et al. (1986)'s paper. The goal of this customized insertion heuristic to construct an initial solution $s_0 = (\boldsymbol{x}, \boldsymbol{y})$, which contains both initial user selection and routing decisions required for starting off local search afterwards. The algorithm for the insertion heuristic is summarised in Algorithm 3.

Algorithm 3: Construction of an Initial Solution

Input: User request data, DRT service data
Output: Initial solution s_0,visited requests $\mathcal{V}$, request pool $\mathcal{U}$

$\mathcal{U} \leftarrow \emptyset$,$\mathcal{V} \leftarrow \emptyset$
for $i \in \mathcal{OB}$ **do**
    update $e_i = e_{n+i} - t_{i,n+i}$
Create sequential inserting order list $InOr = [\arg\min_{i\in\mathcal{P}} e_i, ..., \arg\max_{i\in\mathcal{P}} e_i]$ by increasing e_i
for $i \in \mathcal{P}$ **do**
    Calculate general index G_i using (4.62)
Replace the first $|\mathcal{K}|$ requests in $InOr$ by the $|\mathcal{K}|$ requests with the smallest G_i (seeds)
Adjust the remaining $InOr$ accordingly if (4.62) holds
for $i \in \mathcal{P}$ **do**
    for $k \in \mathcal{K}$ **do**
        Try to insert $InOr[i]$ into the work-schedule of vehicle k, record the additional cost of all feasible insertions
    if *No feasible insertion found* **then**
        Add request $InOr[i]$ into the unvisited request pool $\mathcal{U}$
    else
        Find the feasible insertion of lowest additional cost out of all vehicles and insert request $InOr[i]$
        Add request $InOr[i]$ into set of visited requests $\mathcal{V}$ and update s_0

4.4.2 Local Search

With an initial solution s_0 obtained, local search is executed iteratively on both user selection ($\boldsymbol{y}$) and routing variables ($\boldsymbol{x}$) until no improvement in the solution can be found.

Local Search on User Selection ($\boldsymbol{y}$)

For an incumbent solution $s = (\boldsymbol{x}, \boldsymbol{y})$, the neighbourhood $N(s)$ of solution s is defined as the set of all solutions that can be obtained by applying a single operator, namely ADD, REM and SWAP. Operators ADD and REM are defined as:

- ADD: Add (insert) a request into vehicle work-schedules. This operation can increase the number of operating vehicles, if it is cheaper to assign an empty vehicle to service this additional request. The target vehicles for insertion can be specified with a subscript (e.g. ADD $_k$ indicates the operation of only inserting the request into the work schedule of vehicle k). Note that there might be more than one feasible insertions when applying this operator. In this work, the *best insertion* is adopted, i.e. the operator scans all feasible insertions and returns the insertion s_i with minimum additional cost into the neighbourhood set. Otherwise, the operator returns null if no feasible insertion can be found.

- REM: Remove a request from its corresponding vehicle work-schedule and return the remaining work schedule. The rest of the routing sequence does not change, as the trip simply skips the removed request. This operation can reduce the number of operating vehicles as well as active schedule blocks in a vehicle.

A two-step local search is carried out on user selection. In step one, the neighbourhood $N_1(s)$ of solution s is explored, where the elements in this set are formed by applying either REM or ADD on solution s. In other words, all elements in $N_1(s)$ have a different $\sum_{i \in \mathcal{P}} y_i$ value compared to solution s (either one less or one more). A subsequent local search step on search space $N_1(s)$ is then applied to s, yielding s'. If s' is a better solution than s in terms of objective function z (profit), it replaces s and the search continues on $N_1(s')$ until no better solution can be found.

For step two, an additional operator SWAP is defined:

- SWAP: This operation explores all feasible combinations of simultaneously removing one existing request and adding one request from the pool of unserved requests $\mathcal{U}$, ensuring the total number of serviced requests remains unchanged. Only the best work schedule s_{ij} is returned if more than one feasible insertions are found.

Note that SWAP is essentially ADD and REM executed simultaneously. This operator is added simply to broaden the search space when searching for the best selection of requests, as solutions with the same size of serviced requests but with swapped elements cannot be obtained by applying ADD or REM on s. With a local optimal solution s obtained from the local search on $N_1(s)$, The algorithm then moves to step two, where it explores the extended neighbourhood $N_2(s)$ of solutions obtained by applying SWAP on s. In $N_2(s)$, all solutions have the same $\sum_{i \in \mathcal{P}} y_i$ as s. Similarly, a local search step is applied to s on $N_2(s)$. The algorithm loops over step one and step two until no improvement can be found in both steps. This algorithm is summarized in Algorithm 4.

Algorithm 4: Local Search on User Selection

Input: initial solution s_0, visited requests $\mathcal{V}$, request pool $\mathcal{U}$
Output: s_1

```
/* Initialisation */
s ← s_0
z(s'') = z(s) + 0.0001
/* Local Search */
while z(s'') > z(s) do
    /* Step one */
    repeat
        N_1(s) ← {s}
        for i ∈ U do
            if Feasible insertion(s) found then
                s_i ← Apply ADD to i
                N_1(s) ← N_1(s) ∪ {s_i}
        for j ∈ V do
            s_j ← Apply REM to j
            N_1(s) ← N_1(s) ∪ {s_j}
        s' ← arg max{z(s') : s' ∈ N_1(s)}
        if z(s') > z(s) then
            s ← s'
            Update U and V
    until z(s') = z(s);
    /* Step two */
    repeat
        N_2(s) ← {s'}
        for i ∈ V, j ∈ U do
            if Feasible insertion(s) found then
                s_ij ← Apply SWAP to (i, j)
                N_2(s) ← N_2(s) ∪ {s_ij}
        s'' ← arg max{z(s'') : s'' ∈ N_2(s)}
        if z(s'') > z(s') then
            s' ← s''
            Update U and V
    until z(s'') = z(s');
s_1 ← s''
```

Local Search on Routing Optimisation ($\boldsymbol{x}$)

Once the local search on user selection terminates, a solution with the current best selection of users is passed on to the second part of local search: local search on routing optimisation. In this local search process, the set of selected users is fixed and the routing variables ($\boldsymbol{x}$) are optimised in a classic DARP manner, where all selected requests must be serviced. In this local search, the fol-

lowing two operators are defined for a request i that is assigned to vehicle k in the current solution s:

- RA: This operator Re-Assigns request i to another vehicle. This can be broken down into the combination of operators REM $_k$ and ADD $_{K\setminus k}$ applied on i (i.e. to remove request i from vehicle k and to insert it into any other vehicle). This operator conducts inter-tour exchanges.
- RI: This operator tries to Re-Insert request i into its originally assigned vehicle k. The operator examines all feasible insertions in this vehicle, searching for a possibly better insertion rather than just inserting it back into its original place. Similarly, this can be broken down into the combination of operators REM $_k$ and ADD $_k$ applied on i (i.e. to remove request i from vehicle k and then to insert it back into vehicle k). This operator takes into account intra-tour exchanges.

For each visited request i in solution s, operator RA is first applied on the request, searching for feasible ways of inserting this request into another trip. If RA fails to find any feasible insertion, RI is then carried out to insert the request back to its original trip. In both operations, *best insertion* is adopted. The neighbourhood $N_3(s)$ of solution s is composed of all solutions that can be obtained from s by applying one of these two operators. This way, local search is conducted with both inter-tour and intra-tour exchanges. With RA and RI carried out in parallel, the efficiency of local search is greatly improved (Xiang et al., 2006). For each solution s, a local search is conducted on $N_3(s)$, yielding solution s'. If s' is a better solution than s in terms of objective function z (profit), s' replaces s and the local search restarts on the new incumbent solution. The algorithm terminates when no better solution can be found. The detailed algorithm of local search on routing optimisation is summarised in Algorithm 5.

4.4.3 LS-H Algorithm Summary

After an initial solution is constructed, the algorithm iteratively conduct local search on user selection and routing optimisation in a loop until there is no improvement. This customized heuristic

Algorithm 5: Local Search on Routing Optimisation

Input: s_1 from Algorithm 4, V
Output: s_2
/* Initialisation */
$s \leftarrow s_1$
$z(s') = z(s) + 0.0001$
/* Local Search */
repeat
  $N_3(s) \leftarrow \{s\}$
  for $i \in \mathcal{V}$ **do**
    /* Inter-tour Search */
    if *Feasible insertion(s) found* **then**
      $s_i \leftarrow$ Apply RA to i
      $N_3(s) \leftarrow N_3(s) \cup \{s_i\}$
    else
      /* Intra-tour Search */
      $s_i \leftarrow$ Apply RI to i
      $N_3(s) \leftarrow N_3(s) \cup \{s_i\}$
  $s' \leftarrow \arg\max\{z(s') : s' \in N_3(s)\}$
  if $z(s') > z(s)$ **then**
    $s \leftarrow s'$
until $z(s') = z(s)$;
$s_2 \leftarrow s'$

solution method is named 'Local Search-based Heuristic (**LS-H**)'. A summary of LS-H is depicted in Figure 4.6.

This thesis acknowledge that even though the proposed LS-H is proved to be capable of jumping out of local optima with the help of its layered loops, this solution method is substantially still a classic heuristic approach. Therefore, it can still get stuck in local optima occasionally. Metaheuristics, including Tabu Search and VNS, may yield results that are closer to optimal solutions. Since this work aims at exploring various fare structure formulations and their influence on class-based ridership, development of more advanced metaheuristic solution algorithms is reserved for future research. Instead, in this chapter, a matheuristic solution method is proposed as an alternative to LS-H in section 4.5.

With an initial solution s_0 obtained, local search is conducted iteratively on both user selection

($\boldsymbol{y}$) and routing variables ($\boldsymbol{x}$) until no improvement in the solution can be found.

4.5 Matheuristic Solution Method

The proposed local search algorithm alternates between user selection and route optimization. In the latter, the problem solved is equivalent to a classical cost-minimizing DARP, i.e. given a set of dial-a-ride requests, the goal is to find the minimal cost set of routes to serve all requests. In the CC-DARP, not all requests need to be served, thus it is likely that solving reduced-sized DARP problems with relatively small number of requests lead to competitive solutions if the selected user requests are close to the optimal choice. Since such DARP problems are expected to be smaller in terms of number of requests, they may be solved to optimality using commercial MILP solver. This motivates a matheuristic approach wherein Algorithm 3 in LS-H (Figure 4.6) is replaced by solving a DARP parameterized by the selected user requests $\boldsymbol{y}$. The overview of the MATH-H solution method is illustrated in Figure 4.7.

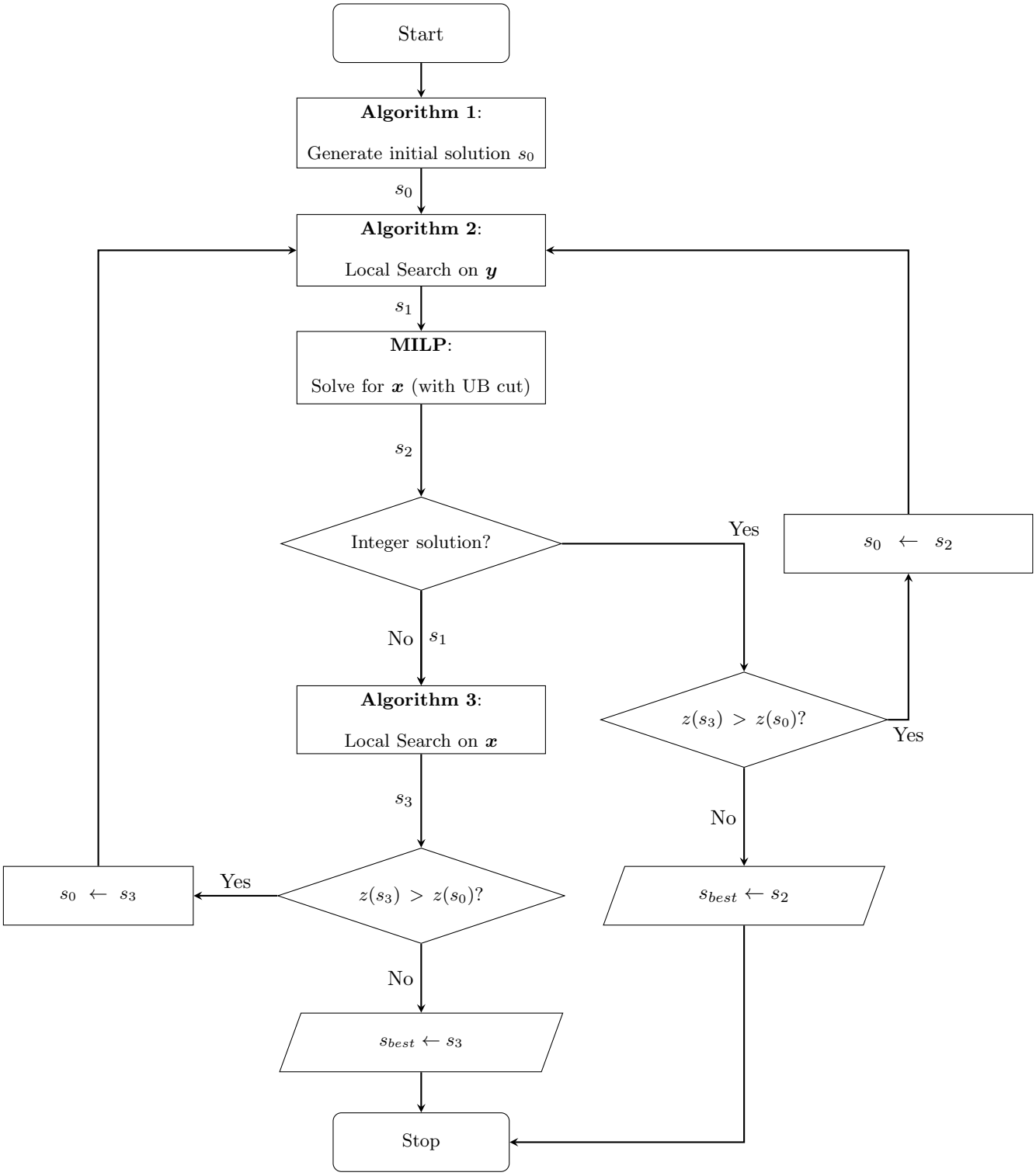


Figure 4.7: MATH-H Solution Method Overlook

In the proposed MATH-H algorithm, a classic DARP is formed with selected users and the current best solution (*aka* the upper bound of routing cost) of outcome s_1. This MILP is then solved using a MILP solver with a customized time limit. If the solution returned s_2 is integer and improves on the previous solution, then a local search on $\boldsymbol{y}$ based on s_2 is triggered using Algorithm 2 in an attempt to further improve the solution. Otherwise, if the solution is fractional, a local search on $\boldsymbol{x}$ based on s_1 is triggered using Algorithm 3 as in LS-H. To ensure that integer solutions returned by the MILP solver improve on s_1 a cut is added in the classical cost-minimizing DARP formulation to which require the solution to be no more costly than the route cost of s_1. Hence, MATH-H only differs from LS-H in that it attempts to solve the route optimization sub-problem parameterized by user selection $\boldsymbol{y}$ using an exact MILP solver. Although solving large instances is expected to remain a challenge for MILP solvers, the reduced set of user requests may help in finding competitive solutions from a route optimization perspective.

4.6 Numerical Results

4.6.1 Comparison between Classic DARP and CC-DARP

In the following experiments, it is assumed that users are all rational and seeking to maximise their own utility. Parameters β_T is set to \$10.6/h based on a cost and benefit analysis conducted in Australia and New Zealand (Litman, 2009). β_S is set to \$21.2/h, given that the value of waiting time is approximately two to three times of the value of travel time (Mohring et al., 1987). β_F is set to 10/\$ after calibration. ω and δ of LS-H are calibrated to be 0.1 and 0.03, respectively. These values are used in all experiments unless stated otherwise. Note that in this section the monetary unit '\$' in refers to Australian Dollar (AUD).

To better understand the differences between the CC-DARP model and the classic DARP model (Cordeau, 2006) from a revenue and profit management's perspective, both models were ran under the same parameter setting ($p = 0.95, s = 1$) with a fare range from \$5.0 to \$35.0, and report the profit obtained by both models. In these experiments, both models are solved by the commercial MILP solver CPLEX, and this section focuses on four instances (a2-16, a2-20, a2-24 and a3-18)

which could be solved to optimality within the one-hour time limit. Figure 4.8 depicts the variation of profit observed using the CC-DARP formulation and the classic DARP formulation.

Figure 4.8 shows that using the Classic DARP formulation, the profit increases in a linear manner proportional to the fare. This is expected since in the classic DARP all user requests must be served. In contrast, using the proposed CC-DARP formulation yields a nonlinear pattern for the profit curves which reflect the effect of increasing fare on the number of customers served. In all four instances examined, it is found that increasing fare first increases profit due to revenue growth, but that this trend is reversed due to the drop in the number of customers served if service fare becomes too expensive. Further, using a fare of \$30 in Instance a2-20 reveals a slightly non-monotonic decrease pattern which highlights the potential of capturing users' preferences in the decision process. The trend shown in Figure 4.8 is interesting from a revenue management perspective: clearly, it is unrealistic to assume that revenue will unlimitedly increase with the increasing fare in the Classic DARP model. On the contrary, CC-DARP model provides a more realistic simulation of profit trend with an adaptive selection of users when fare changes. This insight equips DRT operators with the information they need for designing pricing strategies. This analysis shows that unlike the classic DARP formulation, the proposed chance-constrained approach can inform on customer base selection and revenue management.

4.6.2 LS-H and MATH-H Performance Benchmarking

This section benchmarks the proposed Local Search based heuristic (LS-H) and Math-heuristic (MATH-H) algorithm using Cordeau's instances under a flat fare structure. Sensitivity analysis is conducted on p, s of chance constraints (4.48) and flat fare f. Each instance of the CC-DARP is solved with an exact MILP solver (CPLEX), LS-H and MATH-H separately, CPU times and optimality gaps are then reported. For the exact solution counterpart, CC-DARP is solved using CPLEX with default options and apply some of the pre-processing user cuts (namely 'Bounds on time and load variables', 'Capacity constraints', 'Precedence Constraints' and 'Generalised order constraints') proposed by Cordeau (2006). A time limit of 1 hour is set for all instances and report

time outs with the symbol †. Within MATH-H, the time limit for each CPLEX run is set to 5 minutes in order to keep the overall run time reasonable. Given the time limit, the parameter 'MIP emphasis switch' is set to 1 for MATH-H to emphasize feasibility over optimality, encouraging CPLEX to explore as many feasible solution as possible within limited time.

Sensitivity Analysis on p and s

Tables 4.5 and 4.6 summarize the sensitivity analysis results of p and s on Cordeau's instances. Outcomes of all solution methods including objective value (Profit) and CPU time (time). For CPLEX, the number of requests accepted ($\sum y_i^\star$) and optimality gap (Gap) are reported. The best lower bound (integer feasible solution) is reported as Profit if CPLEX could not solve the problem to optimality within the one-hour time limit. For LS-H and MATH-H, user selection results are compared with CPLEX's $y_i^\star$ and the absolute difference is reported as $\sum |y_i^\star - y_i|$ ($\forall i \in \mathcal{P}$). Profits are highlighted in bold if the optimal solution is found, otherwise a positive profit gap (PG) is recorded. In the PG columns, a negative value indicates that LS-H or MATH-H outperforms CPLEX's best lower bound within the time limit. Minimal, average and maximal PG values are provided at the bottom of each table. These results show that the performance of the heuristics is rather stable, especially when chance constraints (4.57) are binding (i.e. higher p and s values). Furthermore, The fact that for medium-sized instances, $\sum |y_i^\star - y_i|$ values are mostly zero combined with the optimal solutions found by MATH-H validate that for user selection, the heuristics almost always manage to select the optimal subset of requests to service within a few minutes. This suggests that although LS-H may fail to identify optimal solution by a small margin, it is often capable of identifying the optimal set of user requests to accept, which is valuable data for DRT operators. As for larger instances, the heuristics are competitive enough to generate high quality solutions using no more than one-third of the one-hour time limit at most. Note that the CPU time of MATH-H depends largely on the time limit set for its CPLEX segment, which is set to 5 minutes in the experiments. With a higher time limit, better or even optimal solutions may be found but at the cost of prolonging overall run time.

Instance	p	CPLEX				LS-H				MATH-H			
		Profit	Gap (%)	time(s)	$\sum y_i^\star$	Profit	time(s)	$\sum \lvert y_i^\star - y_i \rvert$	PG (%)	Profit	time(s)	$\sum \lvert y_i^\star - y_i \rvert$	PG (%)
a2-16	0.8	**114.84**	$< 1e^{-4}$	0.16	8	**114.84**	4.60	0	0.00	**114.84**	5.93	0	0.00
	0.95	**100.95**	$< 1e^{-4}$	0.08	7	**100.95**	3.82	0	0.00	**100.95**	5.49	0	0.00
	0.99	**100.95**	$< 1e^{-4}$	0.09	7	**100.95**	3.70	0	0.00	**100.95**	4.50	0	0.00
a2-20	0.8	**156.95**	$< 1e^{-4}$	0.08	10	148.78	4.79	0	5.21	**156.95**	6.15	0	0.00
	0.95	**123.13**	$< 1e^{-4}$	0.06	8	120.73	4.52	0	1.95	**123.13**	6.07	0	0.00
	0.99	**84.48**	$< 1e^{-4}$	0.03	6	83.83	3.81	0	0.78	**84.48**	4.82	0	0.00
a2-24	0.8	**225.29**	$< 1e^{-4}$	1.27	15	223.51	7.73	0	0.79	**225.29**	11.05	0	0.00
	0.95	**206.04**	$< 1e^{-4}$	1.61	14	205.61	7.17	0	0.21	**206.04**	10.06	0	0.00
	0.99	**174.61**	$< 1e^{-4}$	0.36	12	174.47	6.93	0	0.08	**174.61**	8.42	0	0.00
a3-18	0.8	**142.26**	$< 1e^{-4}$	0.42	9	137.92	4.74	0	3.05	**142.26**	5.99	0	0.00
	0.95	**142.26**	$< 1e^{-4}$	0.48	9	137.92	4.59	0	3.05	**142.26**	6.14	0	0.00
	0.99	**94.32**	$< 1e^{-4}$	0.23	6	89.97	3.85	0	4.61	**94.32**	4.58	0	0.00
a3-24	0.8	**150.66**	$< 1e^{-4}$	1.25	10	149.17	6.62	0	0.98	**150.66**	9.14	0	0.00
	0.95	**135.90**	$< 1e^{-4}$	0.36	9	133.40	5.82	0	1.84	**135.90**	7.91	0	0.00
	0.99	**135.90**	$< 1e^{-4}$	0.38	9	133.40	5.99	0	1.84	**135.90**	8.92	0	0.00
a3-36	0.8	**277.80**	$< 1e^{-4}$	10.08	18	273.09	14.04	0	1.70	**277.80**	29.10	0	0.00
	0.95	**245.67**	$< 1e^{-4}$	11.38	16	240.35	12.86	0	2.17	**245.67**	26.36	0	0.00
	0.99	**209.83**	$< 1e^{-4}$	4.53	14	206.03	12.09	0	1.81	**209.83**	19.03	0	0.00
a4-16	0.8	**157.79**	$< 1e^{-4}$	1.98	10	155.54	4.64	0	1.42	**157.79**	9.66	0	0.00
	0.95	**142.07**	$< 1e^{-4}$	0.77	9	139.74	4.46	0	1.64	140.83	7.07	1	0.87
	0.99	**92.33**	$< 1e^{-4}$	0.25	6	90.25	3.49	0	2.26	91.41	4.74	0	0.99
a4-32	0.8	**237.22**	$< 1e^{-4}$	46.31	15	224.23	15.03	0	5.48	234.47	201.59	0	1.16
	0.95	**203.32**	$< 1e^{-4}$	5.09	13	192.34	13.20	0	5.40	200.56	22.51	0	1.26
	0.99	**203.32**	$< 1e^{-4}$	5.06	13	192.34	13.63	0	5.40	200.56	23.13	0	1.26
a4-40	0.8	**325.68**	$< 1e^{-4}$	357.78	20	306.30	31.13	0	5.95	**325.68**	626.71	0	0.00
	0.95	**261.43**	$< 1e^{-4}$	4.73	16	247.83	19.38	0	5.20	**261.43**	30.16	0	0.00
	0.99	**243.77**	$< 1e^{-4}$	2.34	15	230.41	17.20	0	5.48	**243.77**	24.09	0	0.00
a4-48	0.8	**281.02**	$< 1e^{-4}$	178.66	18	270.80	41.80	0	3.64	**281.02**	477.33	0	0.00
	0.95	**263.49**	$< 1e^{-4}$	63.08	17	254.61	25.60	0	3.37	**263.49**	142.80	0	0.00
	0.99	**196.61**	$< 1e^{-4}$	3.23	13	190.87	17.23	0	2.92	**196.61**	31.99	0	0.00
a5-50	0.8	400.63	5.49	3600†	25	352.20	57.03	3	12.13	350.42	962.28	3	12.53
	0.95	383.00	5.35	3600†	24	298.82	28.21	5	21.98	332.84	1288.60	2	13.10
	0.99	322.72	3.16	3600†	21	295.94	50.73	2	11.05	298.67	651.60	2	5.66
a6-60	0.8	571.13	8.87	3600†	35	546.59	99.56	0	4.30	549.21	708.61	0	3.84
	0.95	465.03	7.28	3600†	29	445.80	78.26	0	4.14	462.73	696.06	0	0.49
	0.99	381.03	6.19	3600†	24	367.17	59.58	0	3.64	381.03	663.80	0	0.00
a6-72	0.8	563.97	7.99	3600†	35	514.89	81.33	2	8.70	557.82	1469.23	0	1.09
	0.95	516.36	6.26	3600†	32	458.67	159.25	2	11.17	465.61	1039.74	3	9.83
	0.99	443.71	3.63	3600†	28	409.30	69.51	1	7.76	425.95	1056.57	0	4.00
a7-84	0.8	786.59	15.95	3600†	50	**792.41**	292.87	0	-0.74	**790.22**	1248.13	0	-0.46
	0.95	710.42	13.78	3600†	45	710.45	557.46	1	0.00	686.35	921.35	1	3.39
	0.99	588.80	11.55	3600†	37	573.17	284.86	0	2.65	573.17	929.44	0	2.65
a8-96	0.8	760.81	16.63	3600†	49	**779.66**	276.95	0	-2.48	**779.66**	535.99	0	-2.48
	0.95	690.69	22.74	3600†	45	**738.16**	315.50	2	-6.87	**738.16**	529.18	2	-6.87
	0.99	628.39	13.82	3600†	40	623.24	335.50	0	0.82	623.24	488.17	0	0.82
								PG_{max}:	21.98			PG_{max}:	13.10
								PG_{min}:	-6.87			PG_{min}:	-6.87
								PG_{ave}:	3.57			PG_{ave}:	1.18

†: CPLEX timed out

Table 4.5: Sensitivity Analysis on p ($s = 10, f = 20$)

Instance	s	CPLEX				LS-H				MATH-H			
		Profit	Gap (%)	time(s)	$\sum y_i^\star$	Profit	time(s)	$\sum \lvert y_i^\star - y_i \rvert$	PG (%)	Profit	time(s)	$\sum \lvert y_i^\star - y_i \rvert$	PG (%)
	1	**134.86**	$< 1e^{-4}$	0.14	9	129.98	4.86	0	3.62	133.83	5.91	0	0.76
a2-16	10	**100.95**	$< 1e^{-4}$	0.08	7	**100.95**	3.58	0	0.00	**100.95**	4.97	0	0.00
	50	**54.38**	$< 1e^{-4}$	0.03	4	**54.38**	2.64	0	0.00	**54.38**	2.72	0	0.00
	1	**156.95**	$< 1e^{-4}$	0.08	10	148.78	4.64	0	5.21	**156.95**	6.59	0	0.00
a2-20	10	**123.13**	$< 1e^{-4}$	0.06	8	120.73	4.46	0	1.95	**123.13**	6.08	0	0.00
	50	**37.71**	$< 1e^{-4}$	0.03	3	**37.71**	2.60	0	0.00	**37.71**	2.50	0	0.00
	1	**244.98**	$< 1e^{-4}$	2.55	16	241.71	7.35	0	1.33	**244.98**	17.34	0	0.00
a2-24	10	**206.04**	$< 1e^{-4}$	1.59	14	205.61	7.35	0	0.21	**206.04**	11.28	0	0.00
	50	**11.43**	$< 1e^{-4}$	0.02	1	**11.43**	2.35	0	0.00	**11.43**	2.29	0	0.00
	1	**157.55**	$< 1e^{-4}$	0.7	10	153.68	5.06	0	2.46	**157.55**	7.25	0	0.00
a3-18	10	**142.26**	$< 1e^{-4}$	0.48	9	137.92	4.85	0	3.05	**142.26**	6.09	0	0.00
	50	**12.11**	$< 1e^{-4}$	0.02	1	**12.11**	3.07	0	0.00	**12.11**	2.40	0	0.00
	1	**151.94**	$< 1e^{-4}$	1.03	10	149.17	6.84	0	1.82	**151.94**	9.37	0	0.00
a3-24	10	**135.90**	$< 1e^{-4}$	0.38	9	133.40	6.08	0	1.84	**135.90**	9.02	0	0.00
	50	**23.32**	$< 1e^{-4}$	0.05	2	**23.32**	2.68	0	0.00	**23.32**	2.67	0	0.00
	1	**311.95**	$< 1e^{-4}$	22.33	20	308.37	16.60	0	1.15	**311.95**	94.77	0	0.00
a3-36	10	**245.67**	$< 1e^{-4}$	11.19	16	240.35	13.11	0	2.17	**245.67**	28.85	0	0.00
	50	**80.55**	$< 1e^{-4}$	0.11	6	**80.55**	5.74	0	0.00	**80.55**	5.17	0	0.00
	1	**173.95**	$< 1e^{-4}$	4.14	11	171.25	4.76	0	1.55	**173.95**	12.34	0	0.00
a4-16	10	**142.07**	$< 1e^{-4}$	0.75	9	139.74	4.38	0	1.64	140.83	7.20	0	0.87
	50	**13.01**	$< 1e^{-4}$	0.02	1	**13.01**	2.18	0	0.00	**13.01**	2.60	0	0.00
	1	**237.22**	$< 1e^{-4}$	31.11	15	224.23	14.89	0	5.48	234.47	209.10	0	1.16
a4-32	10	**203.32**	$< 1e^{-4}$	5.08	13	192.34	13.21	0	5.40	200.56	22.86	0	1.36
	50	**25.70**	$< 1e^{-4}$	0.06	2	**25.70**	2.89	0	0.00	**25.70**	3.87	0	0.00
	1	**343.38**	$< 1e^{-4}$	778.42	21	326.23	20.71	0	4.99	**343.38**	587.84	0	0.00
a4-40	10	**261.43**	$< 1e^{-4}$	4.78	16	247.83	17.37	0	5.20	**261.43**	30.03	0	0.00
	50	**56.05**	$< 1e^{-4}$	0.09	4	**56.05**	3.99	0	0.00	**56.05**	5.99	0	0.00
	1	**349.28**	$< 1e^{-4}$	2162.97	22	319.20	27.44	1	8.61	331.27	644.86	1	5.16
a4-48	10	**263.40**	$< 1e^{-4}$	34.28	17	254.61	21.82	0	3.34	**263.40**	147.47	0	0.00
	50	**85.29**	$< 1e^{-4}$	0.17	6	71.38	4.69	1	16.31	71.38	4.11	0	16.31
	1	400.23	5.57	3600†	25	352.02	48.51	3	12.04	350.42	961.77	3	12.45
a5-50	10	383.00	5.36	3600†	24	298.82	23.93	5	21.98	332.84	1288.60	3	13.10
	50	**37.98**	$< 1e^{-4}$	0.16	3	**37.98**	4.38	0	0.00	**37.98**	5.40	0	0.00
	1	607.86	9.01	3600†	37	580.82	86.55	0	4.45	580.82	654.92	0	4.45
a6-60	10	465.03	7.31	3600†	29	455.80	68.04	0	4.14	462.73	696.06	0	0.49
	50	**81.29**	$< 1e^{-4}$	0.27	6	80.32	7.60	0	1.19	**81.29**	13.79	0	0.00
	1	597.80	8.32	3600†	37	544.63	75.45	2	8.89	559.19	1061.76	2	6.46
a6-72	10	516.53	6.29	3600†	32	458.67	137.42	2	11.20	465.47	1039.74	3	9.89
	50	**106.17**	$< 1e^{-4}$	0.80	8	105.31	11.74	0	0.81	**106.17**	20.97	0	0.00
	1	790.40	15.37	3600†	50	789.75	199.19	0	0.08	789.75	1141.75	0	0.08
a7-84	10	717.06	12.69	3600†	45	694.26	407.79	0	3.18	693.14	921.35	1	3.18
	50	**98.03**	$< 1e^{-4}$	0.53	7	94.97	13.54	0	3.12	**98.03**	19.66	0	0.00
	1	800.49	20.45	3600†	51	**848.07**	232.45	2	-5.94	**848.07**	511.89	2	-5.94
a8-96	10	690.69	22.74	3600†	45	**738.16**	315.50	2	-6.87	**738.16**	529.18	2	-6.87
	50	**155.02**	$< 1e^{-4}$	3.6	11	147.75	29.51	0	4.69	**155.02**	51.61	0	0.00
								PG_{max}:	21.98			PG_{max}:	16.31
								PG_{min}:	-6.87			PG_{min}:	-6.87
								PG_{ave}:	3.12			PG_{ave}:	1.40

†: CPLEX timed out

Table 4.6: Sensitivity Analysis on s ($p = 0.95, f = 20$)

Instance	f	CPLEX				LS-H				MATH-H			
		Profit	Gap (%)	time(s)	$\sum y_i^\star$	Profit	time(s)	$\sum \lvert y_i^\star - y_i \rvert$	PG (%)	Profit	time(s)	$\sum \lvert y_i^\star - y_i \rvert$	PG (%)
a2-16	10.0	**83.98**	$< 1e^{-4}$	0.83	14	69.73	5.51	5	16.97	77.75	8.55	4	7.42
	20.0	**134.86**	$< 1e^{-4}$	0.14	9	129.98	3.78	1	3.62	133.83	6.52	0	0.76
	30.0	**118.47**	$< 1e^{-4}$	0.05	5	**118.47**	3.17	0	0.00	**118.47**	4.55	0	0.00
a2-20	10.0	**101.94**	$< 1e^{-4}$	0.98	16	95.69	11.82	0	6.13	**101.94**	13.68	0	0.00
	20.0	**156.95**	$< 1e^{-4}$	0.08	10	148.78	3.55	0	5.21	**156.95**	6.38	0	0.00
	30.0	**67.71**	$< 1e^{-4}$	0.03	3	**67.71**	2.61	0	0.00	**67.71**	2.52	0	0.00
a2-24	10.0	**126.62**	$< 1e^{-4}$	18.33	21	116.71	7.04	6	7.82	**126.62**	21.93	0	0.00
	20.0	**244.98**	$< 1e^{-4}$	2.53	16	241.87	4.77	0	1.33	**244.98**	16.92	0	0.00
	30.0	**119.30**	$< 1e^{-4}$	0.05	5	118.38	3.06	0	0.77	**119.30**	4.81	0	0.00
a3-18	10.0	**110.13**	$< 1e^{-4}$	203.64	17	96.83	13.97	1	12.08	103.30	55.42	1	6.20
	20.0	**157.55**	$< 1e^{-4}$	0.69	10	153.68	3.55	0	2.46	**157.55**	6.84	0	0.00
	30.0	**45.17**	$< 1e^{-4}$	0.05	2	**45.17**	2.50	0	0.00	**45.17**	2.50	0	0.00
a3-24	10.0	**109.64**	$< 1e^{-4}$	76.7	17	103.84	8.36	0	5.29	**109.64**	149.84	0	0.00
	20.0	**151.94**	$< 1e^{-4}$	1.03	10	149.17	4.43	0	1.82	**151.94**	9.46	0	0.00
	30.0	**120.02**	$< 1e^{-4}$	0.06	5	119.42	3.63	0	0.50	**120.02**	5.51	0	0.00
a3-36	10.0	**205.54**	$< 1e^{-4}$	2300.64	32	194.33	13.07	0	5.46	**205.54**	618.14	0	0.00
	20.0	**311.95**	$< 1e^{-4}$	18.81	20	308.35	9.38	0	1.15	**311.95**	84.65	0	0.00
	30.0	**221.58**	$< 1e^{-4}$	0.11	9	220.31	7.42	0	0.57	**221.58**	11.49	0	0.00
a4-16	10.0	**87.05**	$< 1e^{-4}$	55.78	14	82.80	4.41	0	4.87	**87.05**	123.24	0	0.00
	20.0	**173.95**	$< 1e^{-4}$	3.75	11	171.25	3.34	0	1.55	**173.95**	12.99	0	0.00
	30.0	**73.93**	$< 1e^{-4}$	0.03	3	71.57	2.75	0	3.19	**73.93**	3.19	0	0.00
a4-32	10.0	192.64	11.45	3600†	29	158.47	43.33	4	17.74	177.74	1268.00	1	7.73
	20.0	**237.22**	$< 1e^{-4}$	32.53	15	224.23	8.50	0	5.48	234.47	201.85	0	1.16
	30.0	**176.06**	$< 1e^{-4}$	0.13	7	143.46	5.20	1	18.52	151.41	6.77	1	14.00
a4-40	10.0	226.60	9.01	3600†	33	205.84	65.38	1	9.16	214.63	1296.70	2	5.03
	20.0	**343.38**	$< 1e^{-4}$	781.50	21	326.23	11.90	0	4.99	**343.38**	935.67	0	0.00
	30.0	**145.88**	$< 1e^{-4}$	0.11	6	140.64	6.20	0	3.60	**145.88**	10.35	0	0.00
a4-48	10.0	272.82	15.78	3600†	40	214.84	141.60	7	21.25	223.35	705.60	8	18.13
	20.0	**349.28**	$< 1e^{-4}$	2451.31	22	319.20	28.36	1	8.61	331.27	643.96	1	5.16
	30.0	**195.49**	$< 1e^{-4}$	0.30	8	195.40	21.80	0	0.05	**195.49**	23.33	0	0.00
a5-50	10.0	279.11	18.76	3600†	41	263.72	95.45	0	5.52	266.01	990.44	0	4.69
	20.0	400.23	5.63	3600†	25	352.02	49.16	3	12.04	367.80	961.77	3	8.10
	30.0	**293.73**	$< 1e^{-4}$	5.06	12	291.93	13.97	0	0.62	**293.73**	26.00	0	0.00
a6-60	10.0	299.81	40.70	3600†	48	**311.10**	94.42	3	-3.76	**311.10**	443.85	3	-3.76
	20.0	607.89	9.01	3600†	37	580.82	85.64	0	4.45	580.82	654.92	0	4.45
	30.0	**323.99**	$< 1e^{-4}$	5.55	13	310.52	20.22	0	4.16	**323.99**	44.50	0	0.00
a6-72	10.0	316.71	30.32	3600†	51	289.98	243.34	2	8.44	299.02	715.82	3	5.59
	20.0	598.61	8.19	3600†	37	544.93	75.99	2	9.02	597.67	1061.76	0	0.16
	30.0	**429.90**	$< 1e^{-4}$	15.77	17	371.73	23.93	2	13.53	378.84	50.40	2	11.87
a7-84	10.0	290.45	101.39	3600†	46	**437.22**	154.30	22	-50.53	**437.22**	488.18	22	-50.53
	20.0	780.98	16.87	3600†	50	**789.75**	199.28	0	-1.22	**789.75**	1141.75	0	-1.12
	30.0	**538.99**	$< 1e^{-4}$	2243.91	21	497.94	64.74	1	7.62	512.75	683.30	1	4.87
a8-96	10.0	250.36	194.28	3600†	41	**556.59**	235.40	44	-122.32	**549.68**	960.94	44	-119.56
	20.0	815.36	18.39	3600†	52	**848.07**	228.91	1	-4.01	**848.07**	511.89	2	-4.01
	30.0	482.47	1.74	3600†	19	408.37	45.52	2	15.36	482.45	1038.62	0	0.04
								PG_{max}:	21.25			PG_{max}:	18.13
								PG_{min}:	-122.32			PG_{min}:	-119.56
								PG_{ave}:	1.54			PG_{ave}:	-1.64

†: CPLEX timed out

Table 4.7: Sensitivity Analysis on f ($p = 0.95, s = 1$)

Sensitivity Analysis on flat fare f

In Table 4.7, the results of the sensitivity analysis on fare f is reported, assuming a flat fare structure is imposed where every passenger pays the same fare for the service. The overall average profit gap for Table 4.7 is 1.54%. From the results it is observed that the increase in fare does not yield a monotonic change in profit, indicating that some fare values result in higher profits with a better balance in the trade-off between higher routing cost and more revenue. The relationship between system profit and fare will be further assessed in Section 4.6.3 under various fare structures.

The results also indicate that LS-H struggles to converge when a cheap fare is charged. When the fare is too low, revenue generated from servicing more requests can barely cover routing costs, which leads to higher computational costs and larger problem size that CPLEX also struggles to cope with. Furthermore, with a cheap fare, chance constraints (4.57) are not binding since every passenger is better off not driving (based on utility functions), which makes the CC-DARP more challenging. This is reflected in both the performance of CPLEX which times out on instance a4-32 for $f = 10$ and in several larger instances too. In these cases, MATH-H helps with narrowing down the profit gap to some extent. In comparison, while the proposed heuristics yield occasionally some profit gaps on low fare instances, they also substantially outperform the best integer solution found by CPLEX on large instances, e.g. a7-84 and a8-96. Note that the $\sum |y_i^\star - y_i|$ columns of Table 4.7 only represents the difference between the values returned by the heuristic and the best lower bound returned by CPLEX within time limit, which is not necessarily the optimal user selection when the instances are too large. It is also worth mentioning that it is more difficult for heuristics to jump out of local optima when tight time windows are imposed (Feillet et al., 2005b).

	LS-H	MATH-H
Average profit gap (PG)	2.59%	0.17%
% of optimal solutions found	17.04%	79.96%

Table 4.8: Comparison between LS-H and MATH-H

Table 4.8 summarizes the performance of LS-H and MATH-H over the benchmarking instances considered. Overall, the sensitivity analysis results (Tables 4.5 - 4.7) show that both LS-H and MATH-H can solve instances up to 96 users within reasonable CPU times with overall average profit gaps of 2.59% and 0.17%, respectively, and are particularly efficient when chance constraints are binding. For medium-sized instances, it is observed that MATH-H is successful in reducing the profit gap. Both LS-H and MATH-H perform well in large instances, generating high quality solutions in a less time-consuming manner comparing to the direct MILP approach (CPLEX). Among all instances solved to optimality by the direct MILP approach MATH-H is able to find nearly 80% of these optimal solutions which is a substantial improvement on LS-H. This suggests that while the route optimization sub-problem is at the heart of the challenge in solving the CC-DARP, the proposed local search on user selection is often good enough to find the optimal selection of users. Hence, the proposed matheuristic could be further improved by replacing the exact classic DARP MILP with a dedicated branch-and-cut-and-price solver for VRPs which has been widely explored in the literature (Molenbruch et al., 2017).

4.6.3 Experiments on NYC Data

In this section, the behaviour of the class-based CC-DARP model is explored on real data extracted from NYC yellow taxi dataset. All 143 requests are categorised into two classes based on their drop-off locations: Class M_1 if the trip destination is in Zone 1 and Class M_2 if trip destination is in Zone 2 (as defined in Figure 3.3a). According to He et al. (2020), the average Value of Time (VoT) in Manhattan is \$29/h. Small (1982) pointed out that the parameter of schedule delay is 1.5-2 times of travel time. Accordingly, Class M_1 is defined as the base class of Manhattan section with $\beta_T^{M_1} = \$29/\text{h}$ and $\beta_S^{M_1} = \$58/\text{h}$. For Class M_2, set $\beta_T^{M_2} = 0.9\beta_T^{M_1}$ and $\beta_S^{M_2} = 0.9\beta_S^{M_1}$, assuming a slightly lower VoT for outside downtown area. Note that this is purely for the purpose of distinguishing two user classes.

Travel time is queried for all 143 requests ($N = 143$) to obtain t_{ij} ($\forall i \in \mathcal{N}, \forall j \in \mathcal{N}$) from Google Maps API during the morning peak (6am-9am) on a weekday. The travel cost c_{ij} ($\forall i \in N, \forall j \in N$)

is then set proportional to t_{ij}: $c_{ij} = 0.1t_{ij}$ ($\forall i \in \mathcal{N}, \forall j \in \mathcal{N}$). The test instance named "NYC" with $n = 143$ and $|\mathcal{N}| = 288$ is obtained by adding two artificial depot nodes $\{0, 2n + 1\}$. Travel cost from depot to all nodes is set to \$2.5 as a remedy for not having a fixed cost penalty for dispatching an additional vehicle in the objective function. For each request $i \in \mathcal{P} = \{1, ..., 143\}$, the trip is determined to be an inbound trip ($i \in \mathcal{IB}$) if i is an odd number, outbound trip ($i \in \mathcal{OB}$) otherwise. For each inbound/outbound trip, a 15-minute time window is set on its departure/arrival time, with its recorded pickup/drop-off time as the midpoint of the time window. For the record, the cost of transit in NYC is \$2.75 per trip. The average taxi fare of morning peak on the sample day is \$12.93.

Flat Fare Structure

This section first compares the proposed heuristics LS-H and MATH-H against the commercial MILP solver CPLEX on flat fare instances. A flat fare structure charges a single fare for all passengers regardless of their travel times or travel distances. When a flat fare structure is applied, the base fare level plays a crucial rule in ridership and system profit. The results are summarized in Table 4.9.

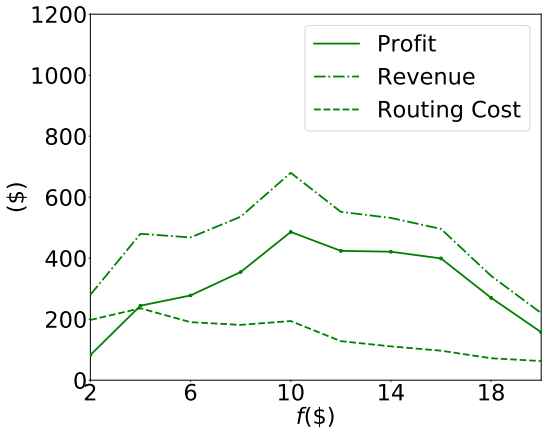
This comparison shows that the proposed heuristics significantly outperfom CPLEX on most large scale instance. Specifically, for all instances with a flat fare ranging from 2 to 12, CPLEX is

f	CPLEX				LS-H				MATH-H			
	Profit	Gap (%)	time(s)	$\sum y_i^\star$	Profit	time(s)	$\sum \lvert y_i^\star - y_i \rvert$	PG (%)	Profit	time(s)	$\sum \lvert y_i^\star - y_i \rvert$	PG (%)
2.0	0	$8.72e^{15}$	3600†	0	82.28	1227.26	91		82.28	1468.36	91	
4.0	0	$1.74e^{16}$	3600†	0	215.50	1493.24	91		215.50	1753.80	91	
6.0	0	$4.76e^{14}$	3600†	0	262.87	885.83	65		262.87	1257.59	65	
8.0	0	$5.35e^{14}$	3600†	0	358.13	784.64	59		358.13	1132.39	59	
10.0	0	$5.87e^{14}$	3600†	0	455.79	527.34	54		455.79	2463.96	54	
12.0	0	$5.48e^{14}$	3600†	0	397.13	365.65	40		397.13	660.36	40	
14.0	390.43	37.82	3600†	31	384.46	192.58	10	1.53	384.46	494.70	10	1.53
16.0	324.21	46.25	3600†	22	345.00	115.60	12	-6.41	345.00	360.64	12	-6.41
18.0	332.83	6.28	3600†	21	280.45	146.00	5	15.74	280.45	443.55	5	15.74
20.0	**174.95**	$< 1e^{-}4$	185.37	10	157.31	33.23	2	10.08	**174.95**	476.05	0	0.00

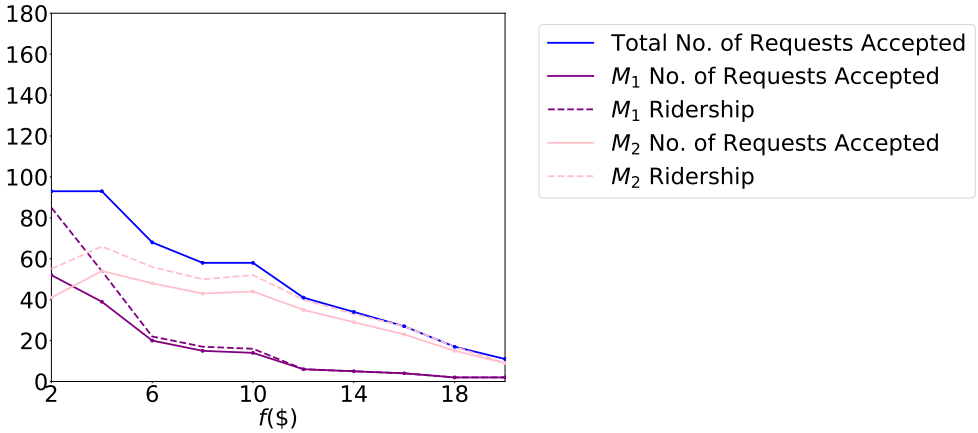
†: CPLEX timed out

Table 4.9: Algorithms comparison with the NYC Dataset instance using a flat fare

unable to find a solution with nonzero profit within a time limit of one hour. In contrast, LS-H and MATH-H find solutions serving up to 91 requests (for low fares). For a fare of 14, CPLEX is able to find a solution serving 31 customers with an optimality gap of 37.8% while LS-H and MATH-H both find a solution only slightly less efficient with a profit gap of 1.53%. Using a fare of 16, the proposed heuristics outperform the solution found by CPLEX by -6.41%. For a fare of 18, the solution found by CPLEX serves 21 requests and outperforms that found by the heuristics by 15.75%. Using a fare of 20, both CPLEX and MATH-H are able to find the optimal solution which only serves 10 user requests while LS-H finds a solution within 10.08% of the optimum. This behavior confirms the trends observed in the benchmarking on instances generated from Cordeau's datasets: for large fares, several user requests are eliminated due to the chance-constraints and the resulting routing problem is of small scale, which is manageable by commercial MILP solvers. In turn, lower fares substantially increase the difficulty of the CC-DARP and this comparison clearly highlights the benefits of the proposed heuristics. Further, this analysis also shows the limits of MATH-H in the face of large scale instances: except for the instance with the largest fare ($f = 20$), both heuristics systematically find the same solution and LS-H exhibits a non-negligible reduction in runtime compared to MATH-H in some cases. For example, using $f = 10$, LS-H is at least four times faster than MATH-H. Additional numerical experiments conducted using MATH-H with varying time limit for CPLEX (up to 15 min) on NYC data-based instances revealed that this did not further improve solution quality. Therefore, LS-H is used for the remaining of the experiments reported in the paper since it is less computationally costly and thus more scalable.



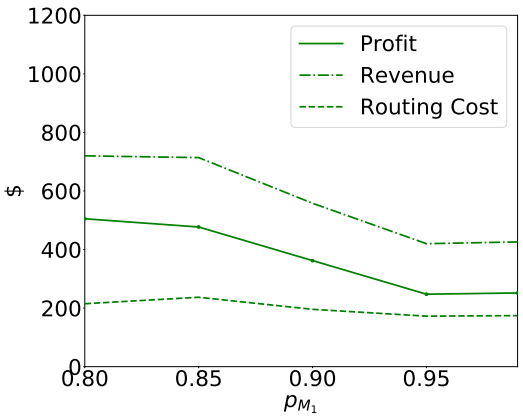
(a) Profit, revenue and routing costs - f.



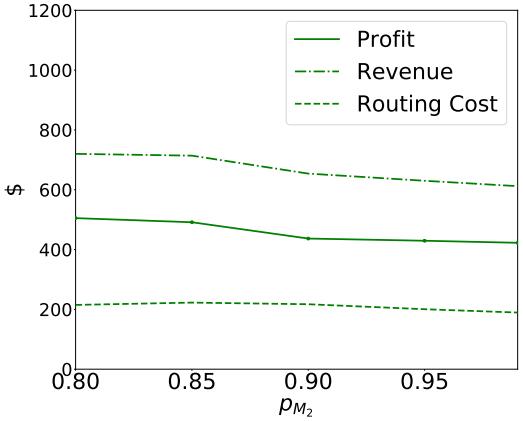
(b) Number of accepted requests - f.

Figure 4.9: Results from the NYC instance under a flat fare structure $f_{im} = f$ for all requests $i \in \mathcal{P}$ and user class $m \in \mathcal{M}$.

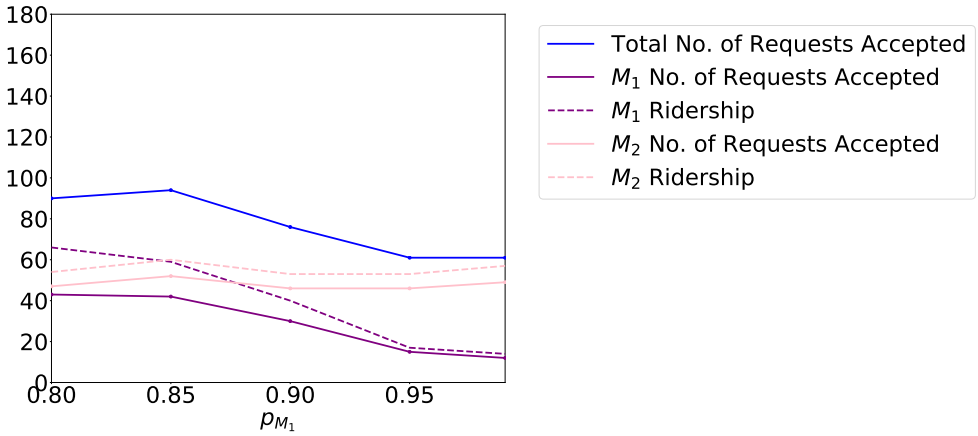
Figure 4.9a shows how profit, revenue and routing cost of the system change with an increasing fare f. As f increases, system profit does not increase or decrease monotonically. The optimal fare level for maximum profit lies around the peak of the profit curve. Figure 4.9b demonstrates how ridership falls in both classes. Note that in this model, the fare is charged per person instead of per request. As observed from Figure 4.9b, passengers with higher VoT (Class M_1) tend to have a higher fare elasticity, as ridership in M_1 drops more sharply than ridership in M_2. This indicates that the bindingness of chance constraints has a more significant effect on the user group with higher VoT under flat fare structures.



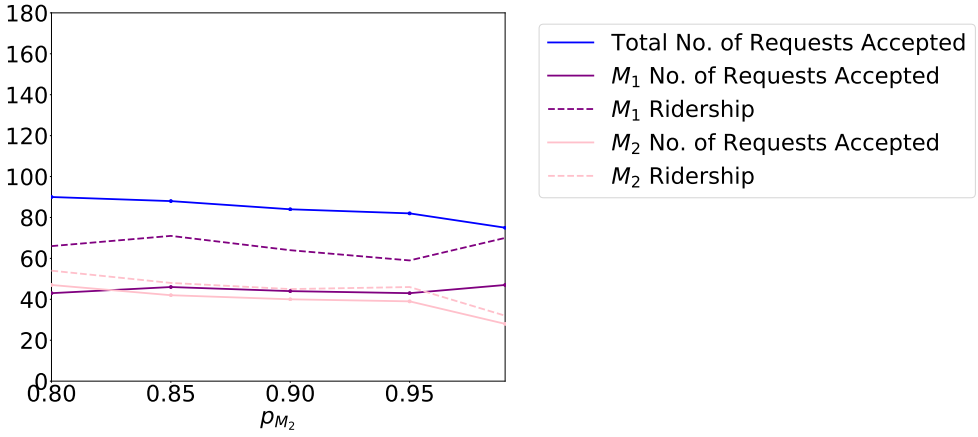
(a) Profit, revenue and routing costs - p_{M1}.



(b) Profit, revenue and routing costs - p_{M2}.



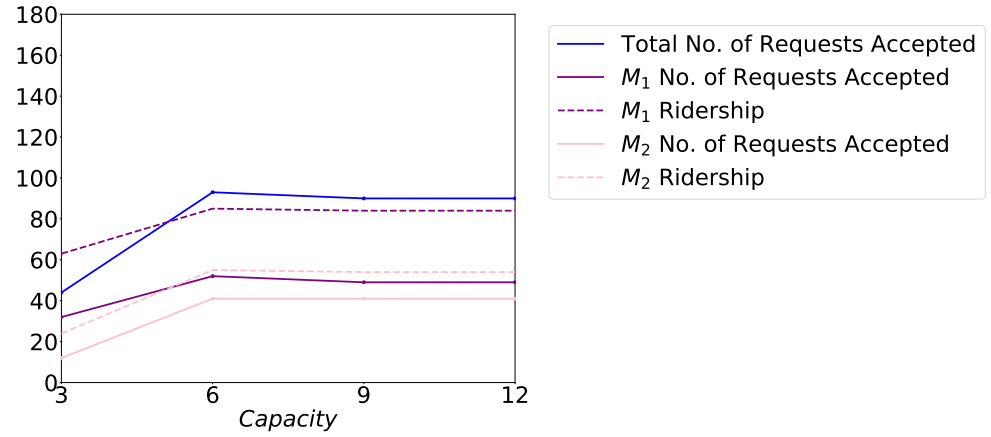
(c) Number of accepted requests - p_{M1}.



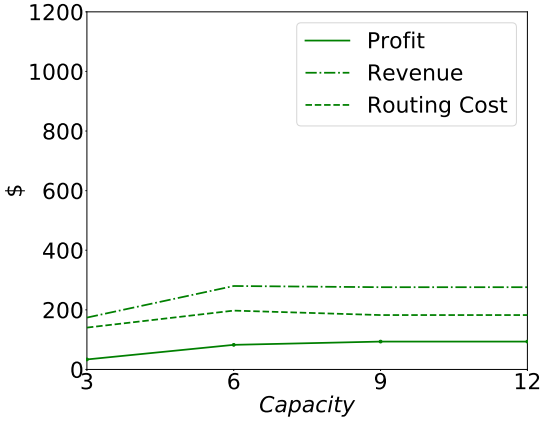
(d) Number of accepted requests - p_{M2}.

Figure 4.10: Results from the NYC instance when changing p_{M1} and p_{M2} under a flat fare structure.

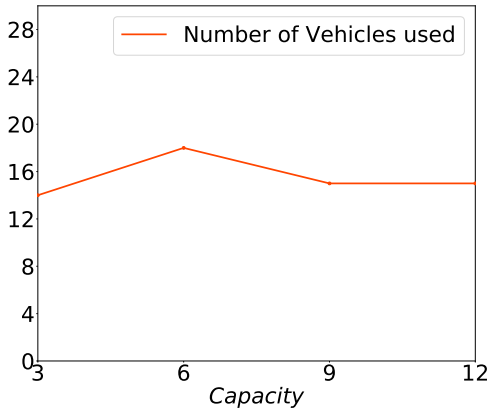
Figure 4.10 displays the effect of varying p_{M_1} and p_{M_2} in the range of $(0.80, 0.99)$ as f is set to \$6.0. While profit falls as p_{M_1} and p_{M_2} being tightened up (Figure 4.10a and 4.10b), Figure 4.10c and 4.10d indicate that an increase in p tend to cause a sharper drop in ridership in the group with higher VoT, as more requests in Class M_1 are filtered out by chance constraints when p_{M_1} increases. Note that when increasing p in one class, the ridership in the other class increases as declining more requests in one class enables capacitated vehicles to accept more requests from the other class.



(a) Number of accepted requests - Capacity.



(b) Profit, revenue and routing costs - Capacity.



(c) Fleet size - Capacity.

Figure 4.11: Results from the NYC instance when changing *Capacity* under a flat fare structure.

Capacity of vehicles could also affect the performance of the system. In this study, it is assumed

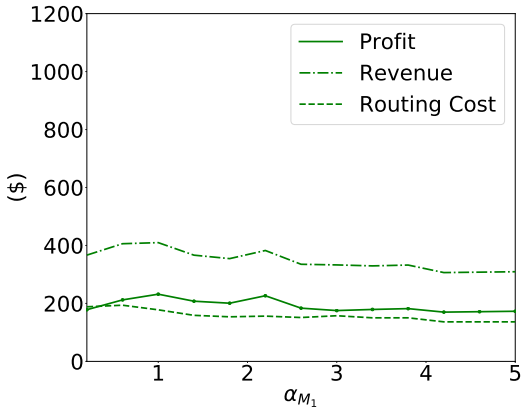
that all vehicles are homogeneous, and passengers from the same request are not allowed to be separated and served by different vehicles. Figure 4.11 reports changes in profit, ridership and number of vehicles used when the capacity of the fleet is increased while holding p and f constant. As shown in Figure 3.3, the size of user requests varies from 1 to 6. Inevitably, using vehicles with capacities smaller than 6 will automatically filter out some requests that exceed the capacity, which leads to lower profit and ridership when capacity is low. If vehicle capacity is greater than 6, all requests are eligible for DRT services. Using vehicles with higher capacity could possibly reduce fleet size. However, when the time windows are tight, fleet size might not vary significantly even with bigger vehicles.

Distance-based Fare Structure

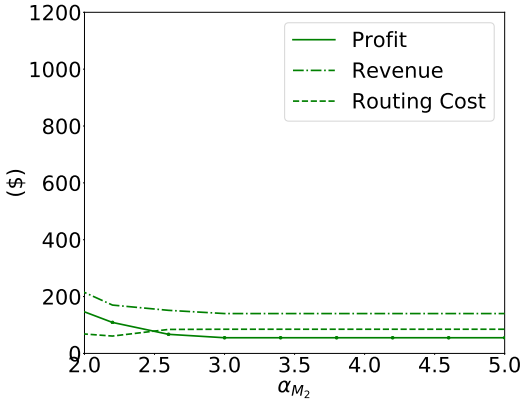
Distance-based fare structure is another widely adapted fare structure in transit systems, where a unique fare is established for each station or stop pair. For the NYC instance, the distance based fare for each class is defined as:

$$f_{im} = \alpha_m t_{i,n+i}, \qquad \forall m \in \mathcal{M}, \forall i \in \mathcal{P}_m.$$

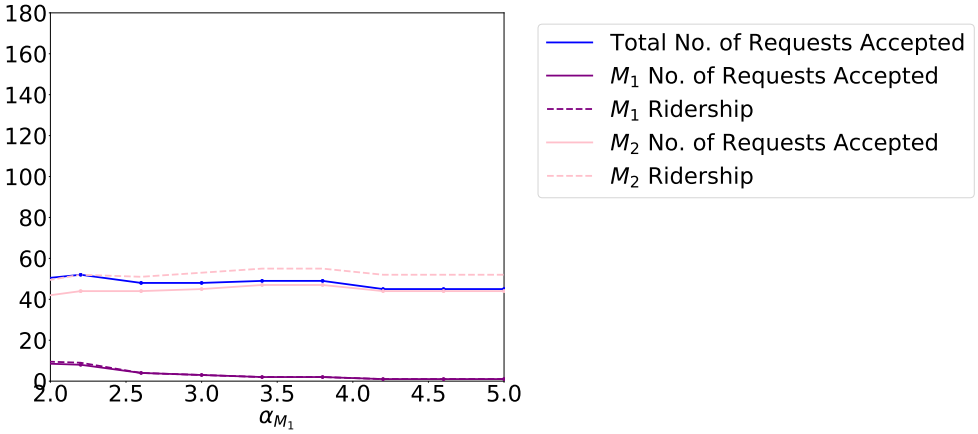
Figure 4.12 shows the behaviours of profit and ridership of the system if increase α_m is increased in one class while holding the other one unchanged. Comparing Figures 4.12a and 4.12c with Figures 4.12b and 4.12d, it is observed that increases in α_{M_2} cause more drastic changes in both profit and ridership, indicating that adjusting the distance-based fare parameter of the lower VoT class has a bigger impact on the system. On the other hand, Figure 4.13 displays the results of the influence of increasing p_{M_1} and p_{M_2}, respectively. Figures 4.13b and 4.13d show that tightening the chance constraints on M_2 may result in an increase in the overall profit and ridership. When the fleet size or the capacity of vehicles is limited, tightening the chance constraints in one class is equivalent to putting more weight on the other class(es) that are potentially more profitable.



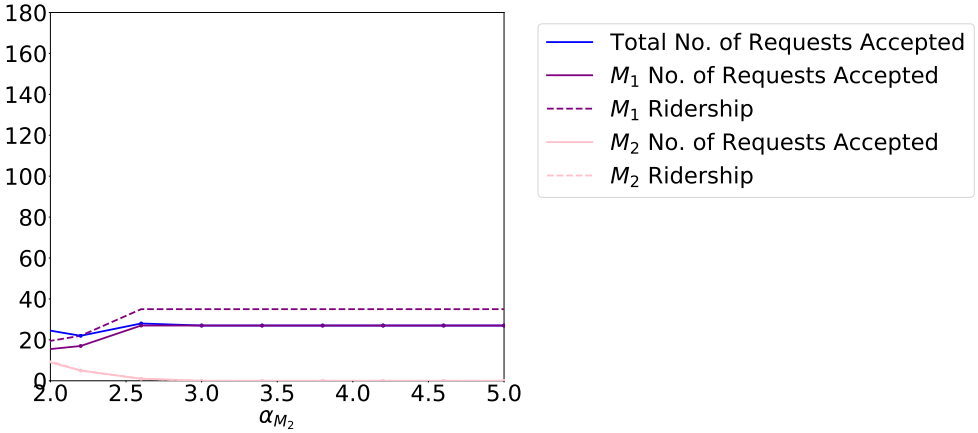
(a) Profit, revenue and routing costs - α_{M1}.



(b) Profit, revenue and routing costs - α_{M2}.

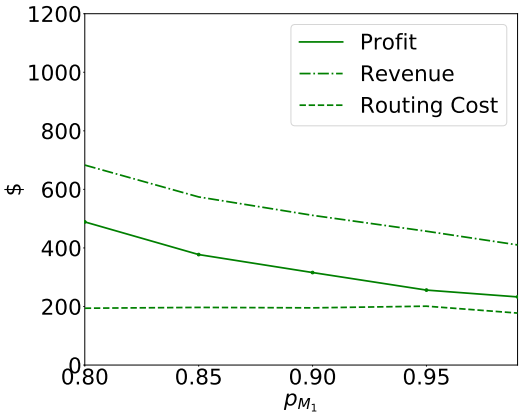


(c) Number of accepted requests - α_{M1}.

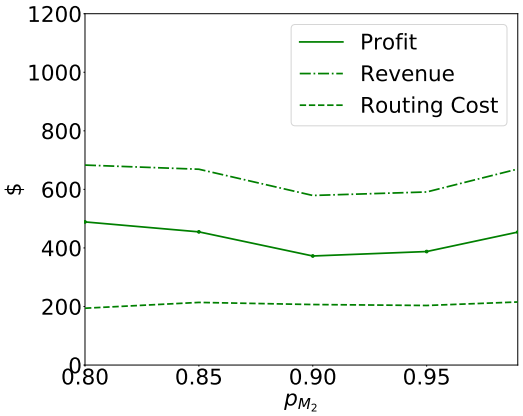


(d) Number of accepted requests - α_{M2}.

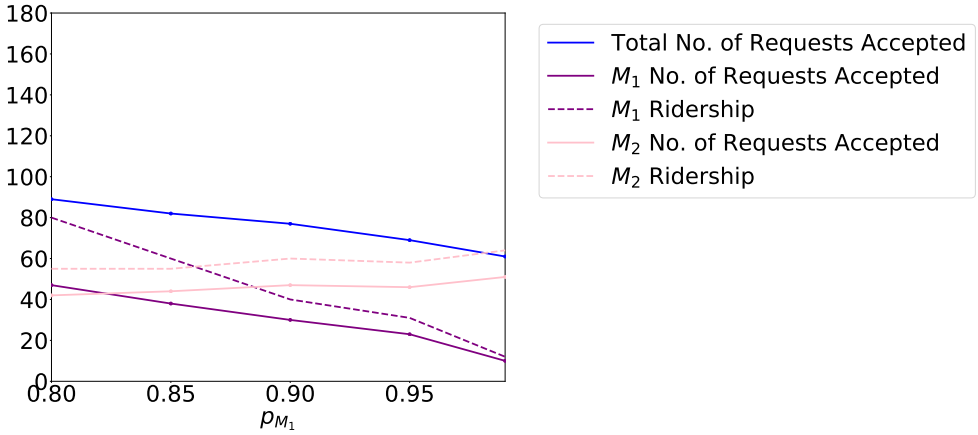
Figure 4.12: Results from the NYC instance when changing α_{M1} and α_{M2} under a distance-based fare structure.



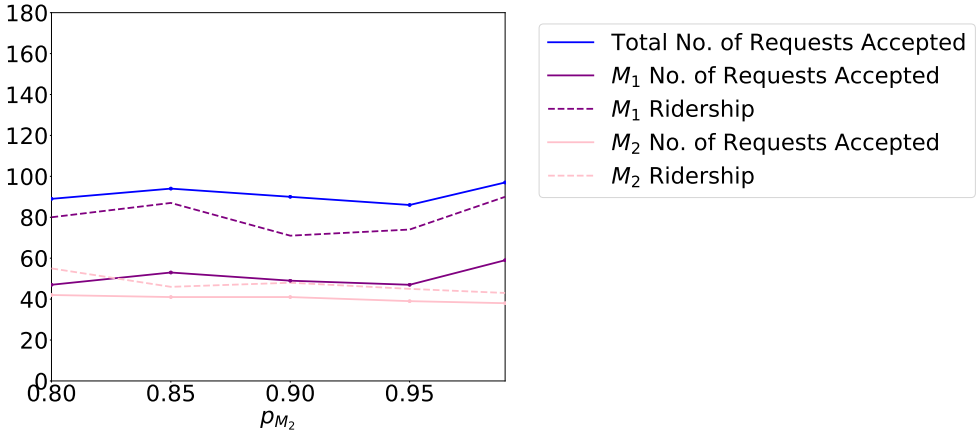
(a) Profit, revenue and routing costs - p_{M1}.



(b) Profit, revenue and routing costs - p_{M2}.

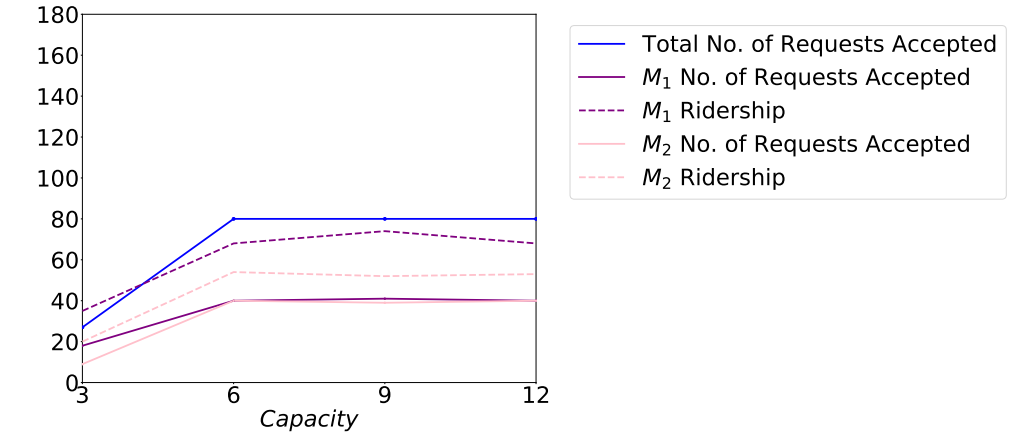


(c) Number of accepted requests - p_{M1}.

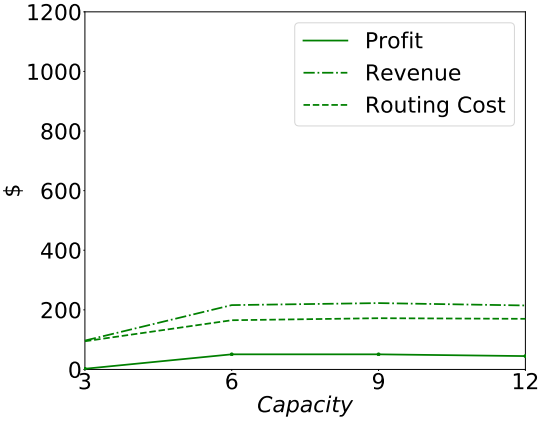


(d) Number of accepted requests - p_{M2}.

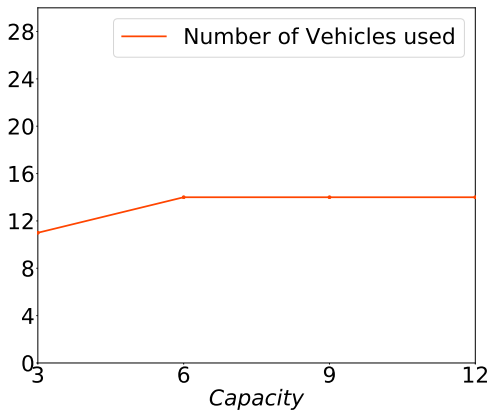
Figure 4.13: Results from the NYC instance when changing p_{M1} and p_{M2} under a distance-based fare structure.



(a) Number of accepted requests - capacity.



(b) Profit, revenue and routing costs - capacity.



(c) Fleet size - capacity.

Figure 4.14: Results from the NYC instance when changing the vehicle capacity Q under a distance-based fare structure.

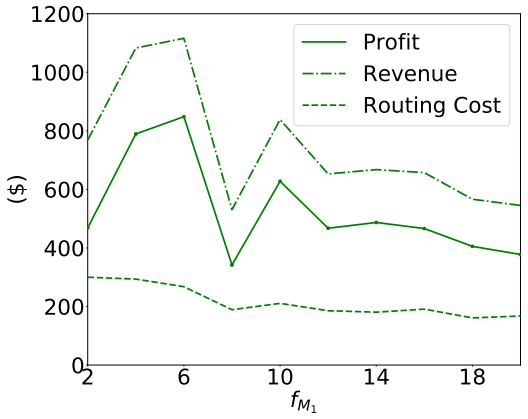
Zone-based Fare Structure

A zone-based or zonal fare structure determine fare based on the number of zones each passenger travels through. In this section, some modifications are made on the zone-based fare structure, and simply define the fare based on the destination zone of the trip, namely f_{M_1} and f_{M_2}. This simplified zone-based pricing rule is inspired by the implementation of congestion pricing policy to charge vehicles entering Manhattan under 60th Street (Zone 2) (**?**).

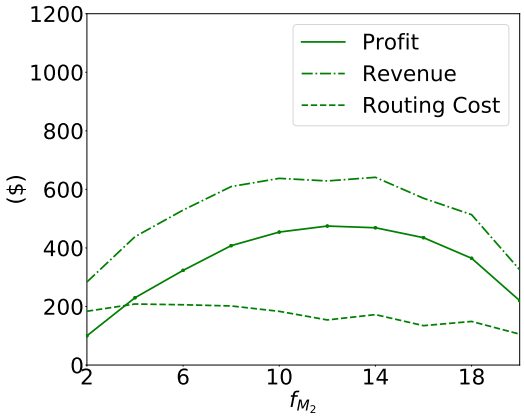
Similarly, this section starts with testing how increasing either f_{M_1} or f_{M_2} (while the other one is set to \$6.0) affects the overall profit and ridership. In Figures 4.15a and 4.15c, it is observed that the increasing f_{M_1} causes more drastic fluctuations in both profit and ridership, since passengers of Class M_1 are more sensitive to changes in fare given their high VoT. Meanwhile, Figures 4.15b and 4.15d depict smoother transitions in curves when f_{M_2} increases.

In both Figures 4.16 and 4.17, f_{M_1} is set to \$3.0 and f_{M_2} to \$2.0. This experiment then explores the impact of using a cheaper fare in Zone 2, how does tightening up chance constraints in each class impact profit and ridership (see in Figure 4.16). With the combination of higher fare f_{M_1} and higher VoT for Class M_1, the overall profit and ridership in M_1 drop sharply to almost zero in Figures 4.16a and 4.16c as p_{M_1} increases. On the contrary, tightening the chance constraints in M_2 has minor impacts on the system, since f_{M_2} is rather cheap and users are mostly better off when not driving.

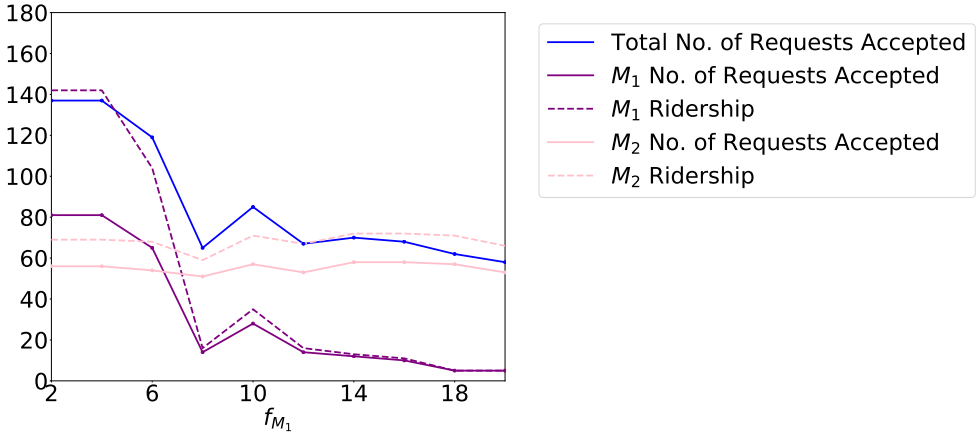
Figures 4.14 and 4.17 illustrate how different vehicle capacities affect profit, ridership and fleet size under distance-based and zone-based fare structures, respectively. Note that even though the impact of capacity appears to be minor in these experiments when capacity is bigger than 6, it is not necessarily always the case. Growing capacity could have a more significant impact on routing optimisation and user selection when requests are more clustered spatially and/or temporally.



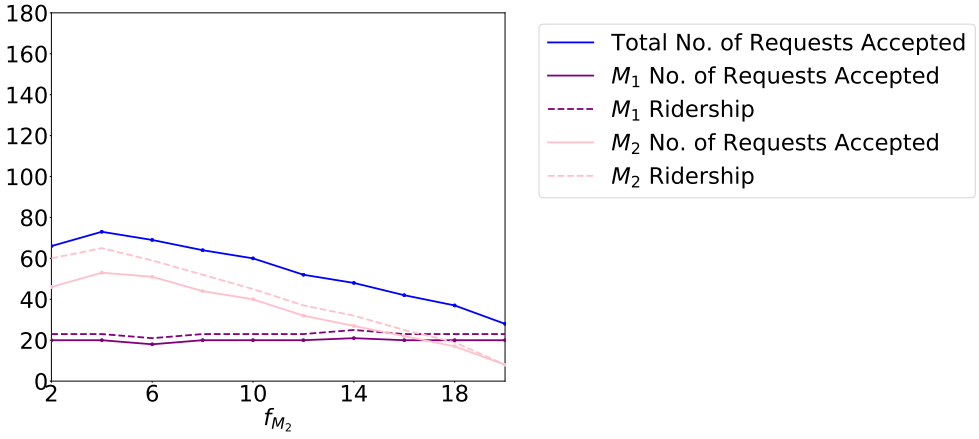
(a) Profit, revenue and routing costs - f_{m1}.



(b) Profit, revenue and routing costs - f_{m2}.

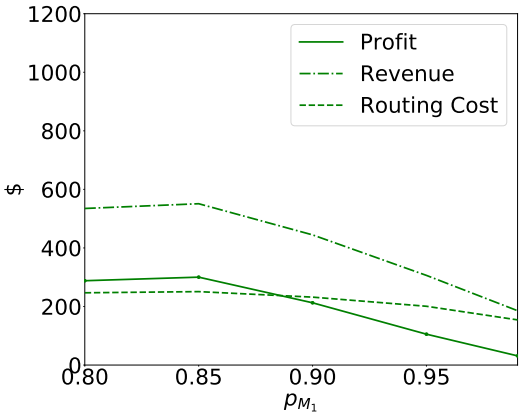


(c) Number of accepted requests - f_{m1}.

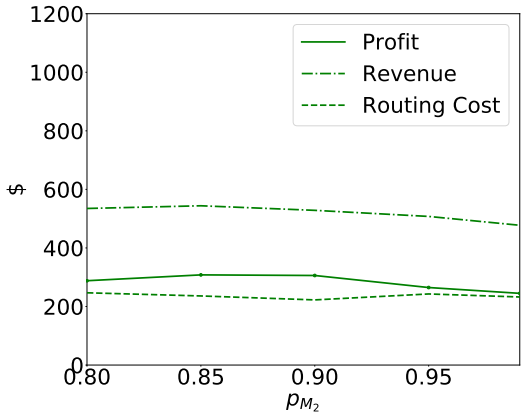


(d) Number of accepted requests - f_{m2}.

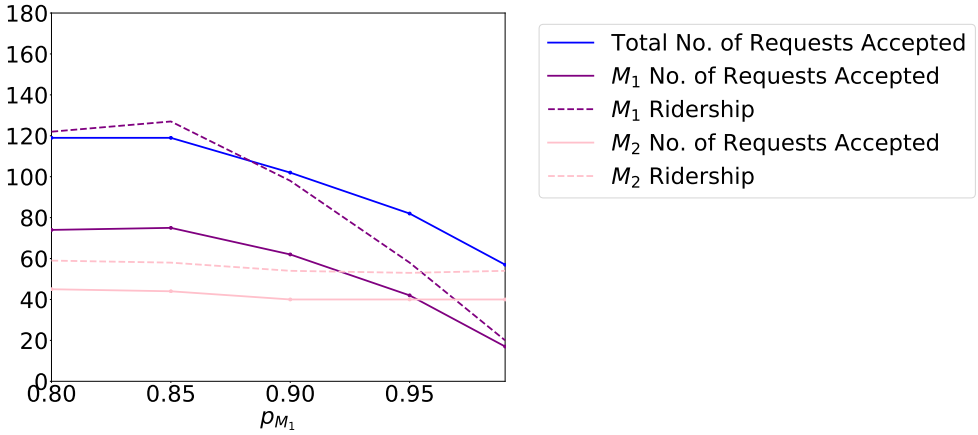
Figure 4.15: Results from the NYC instance when changing f_{m1} and f_{m2} under a zone-based fare structure.



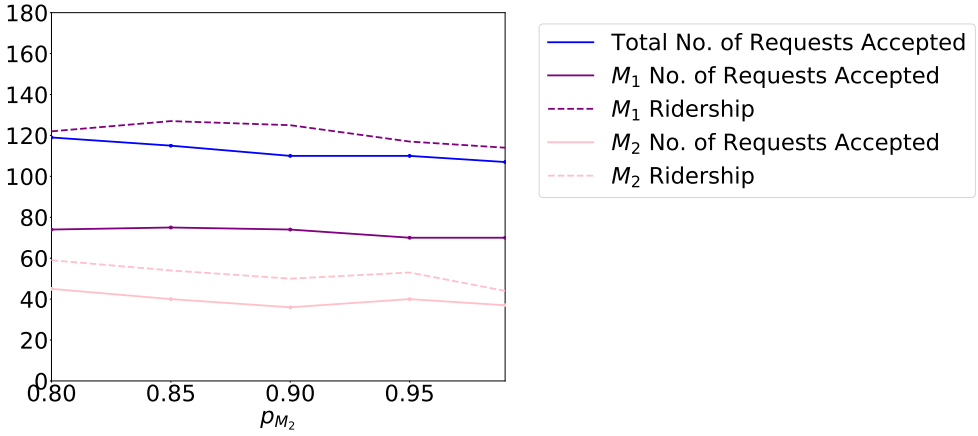
(a) Profit, revenue and routing costs - p_{m1}.



(b) Profit, revenue and routing costs - p_{m2}.

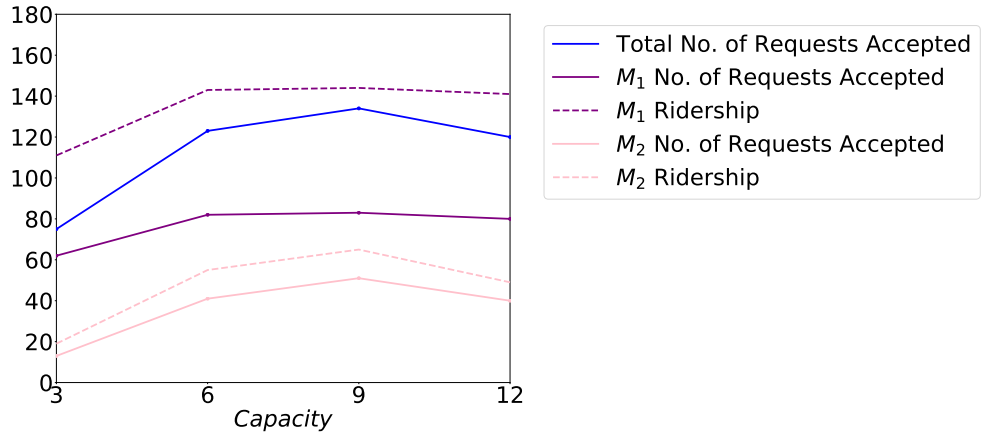


(c) Number of accepted requests - p_{m1}.

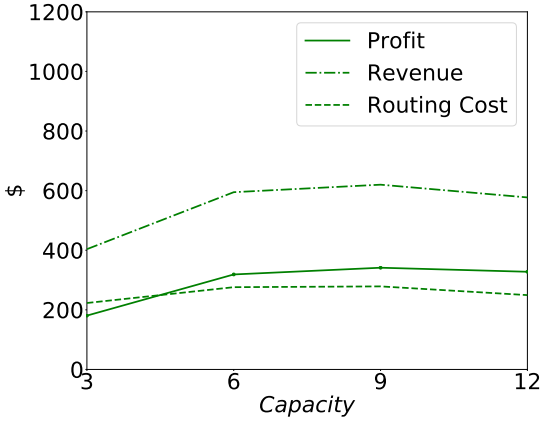


(d) Number of accepted requests - p_{m2}.

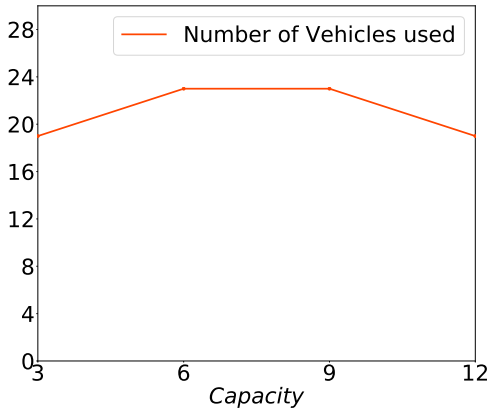
Figure 4.16: Results from the NYC instance when changing p_{m1} and p_{m2} under a zone-based fare structure.



(a) Number of accepted requests - Capacity.



(b) Profit, revenue and routing costs - Capacity.



(c) Fleet size - Capacity.

Figure 4.17: Results from the NYC instance when changing *Capacity* under a zone-based fare structure.

Overall, the accept/reject mechanism of the proposed CC-DARP model reacts more sensitively towards the user class with a higher VoT (M_1), indicating that these users are more elastic when fare policy changes and the profitability of serving such users decreases more rapidly. Across three fare structures, the zone-based fare structure yields the largest profit and ridership, which gives DRT operators insight on revenue management strategies and pricing policies in the context of a class-based system. Furthermore, integrating the demographic attributes of each zone into this class-based model formulation gives zone-based fare structure the edge of designing customized pricing strategies. The flat fare structure, albeit simple, performs quite well. Additionally, its simplicity

of form and low ticket issuing cost makes it more appealing to transit systems. The distance-based fare structure, on the other hand, performs poorly in maximising profit and ridership and its reactions towards variations in conditions are rather mild. Even though the distance-based fare structure is generally perceived as the fairest pricing policy, the findings suggests that it may not be a profit-maximizing strategy for DRT operators.

4.7 Conclusion

In this chapter, two extended DARP models are introduced, capturing users' preferences in a DRT context in the long run. Users' representative utilities for alternative transportation modes are taken into account, including a Demand-Responsive Transportation (DRT) service. This chapter considers utility-maximizing users and proposes a mixed-integer programming formulation for the CC-DARP that captures users' preferences in the long run via a Logit model. The formulation of the multi-class DARP enables the model to explore the behaviours of various user classes under different price structures. This provides insights to DRT operators on pricing policies design as well as revenue management strategies. Two customized heuristic solution methods with layered local search structure are developed to optimise both routing decisions and user selections. Numerical results showed that the customized local search based heuristic algorithm (LS-H) and matheuristic (MATH-H) performed well on DARP benchmarking instances of the literature when compared to the best integer solution returned by an exact MILP solution. The proposed solution methods are further implemented on a realistic case study derived from yellow taxi trip data from New York City (NYC). The results obtained on the realistic case study reveal that a zonal fare structure is a profit-maximizing strategy for DRT operators.

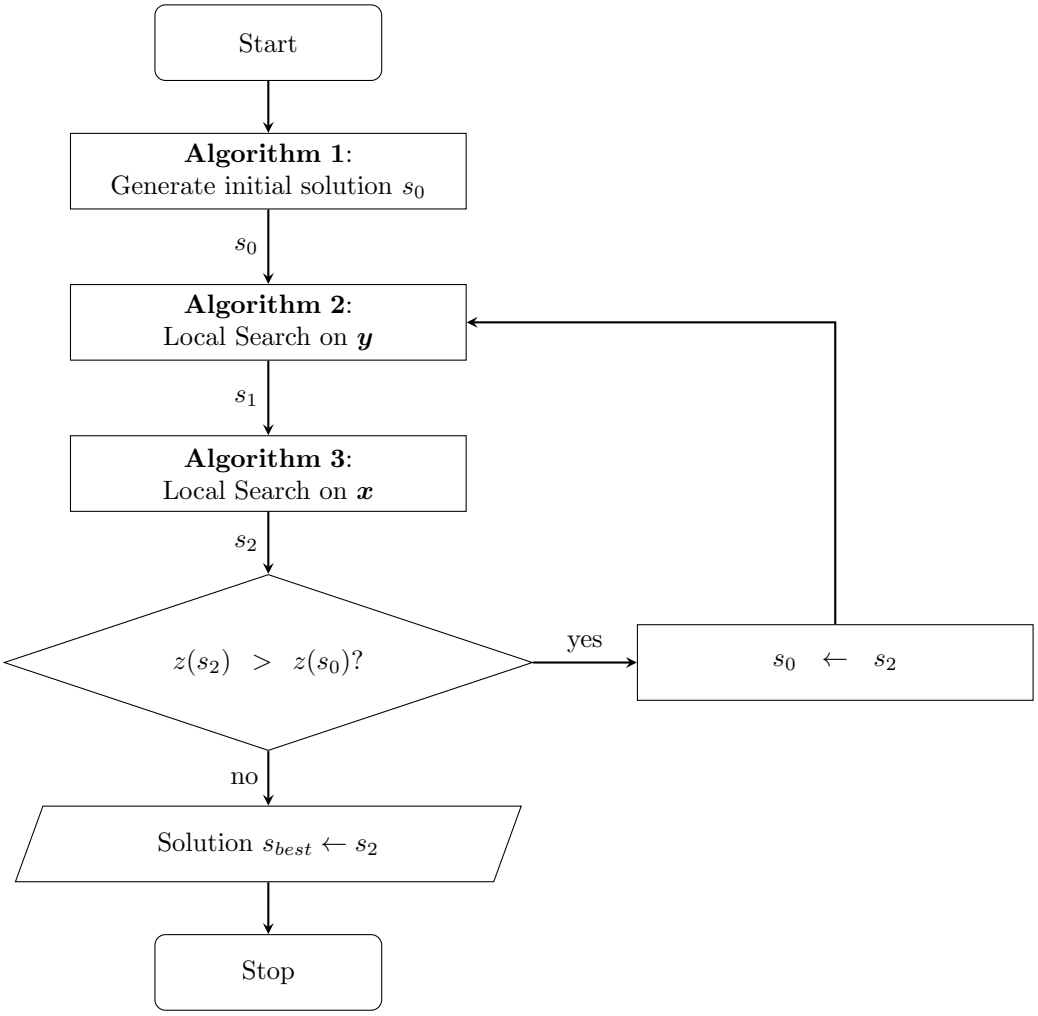


Figure 4.6: LS-H Solution Method Overlook

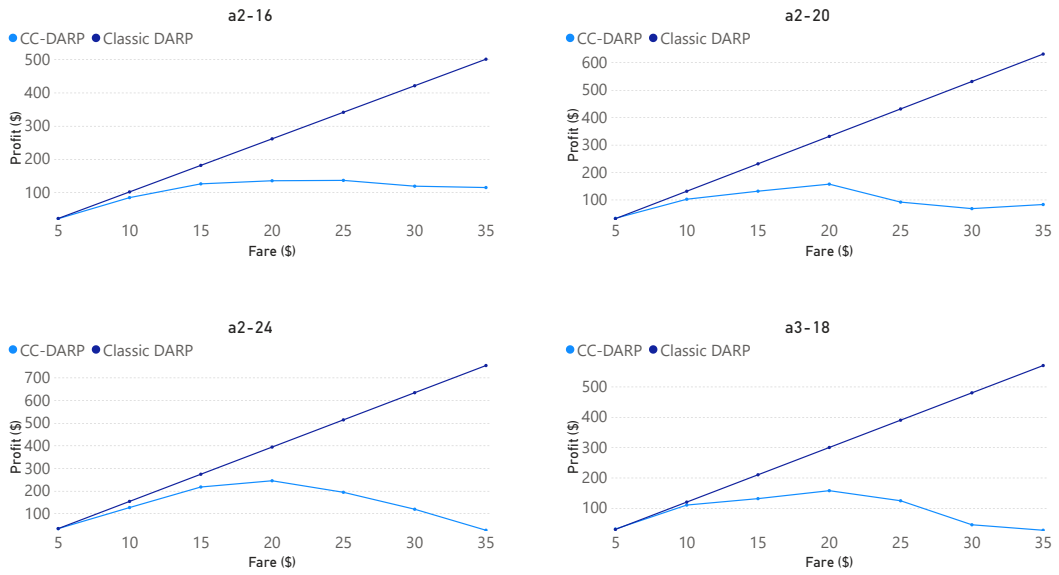


Figure 4.8: Profit trends for CC-DARP and Classic DARP formulations

Chapter 5

Hybrid Metaheuristics and Reinforcement Learning Approaches for the DARP

5.1 Introduction

In this chapter, heuristic algorithms based on column generation are developed and implemented to solve the DARP. In literature, only very few studies conducted to integrate Machine Learning (ML) into solution methods of VRP and its generalisations. Attempts of ML are even rarer when it comes to solving the DARP, which often applies End-to-End learning directly where ML models are directly trained to solve combinatorial optimisation problems on a network level. This thesis steels to another direction and explores the possibility of applying ML alongside optimisation algorithms, where ML is called iteratively to support the global decisions made by optimisation algorithms. In this chapter, hybrid structures integrating elements including Machine Learning, Branch-and-Bound and Metaheuristics are proposed, and their performances are tested with the same set of benchmark instances by Cordeau (2006). As introduced in Chapter 3, the DARP can

be formulated as a set partitioning problem as follows:

$$\min \sum_{p \in \Omega} c^p \lambda^p \tag{5.1}$$

subject to:

$$\sum_{p \in \Omega} a_i^p \lambda^p = 1, \qquad \forall i \in \mathcal{P}, \tag{5.2}$$

$$\sum_{p \in \Omega} \lambda^p \leq K, \tag{5.3}$$

$$\lambda^p \in \{0, 1\}, \qquad p \in \Omega. \tag{5.4}$$

In model (5.1)-(5.4), c_p is the routing cost of path p. Binary decision variable $\lambda^p = 1$ if path p is selected in the optimal solution. Ω is the set (pool) that contains all routes in the network. Since the size of Ω is too large to enumerate, column generation based solution algorithms often start off with a small pool of columns Ω' that contains an initial solution. By solving the linear relaxation of the above set partitioning problem (namely the Restricted Master Problem, RMP) iteratively, promising columns are identified and added to the pool. In each column generation iteration, the subproblem (i.e., the pricing problem) takes the values of dual variables d_i ($\forall i \in \mathcal{P}$) that are associated with constraints (5.2) as an input, and returns columns (paths) with negative reduced costs to be added into pool Ω'. The reduced cost of a column p is calculated as follows:

$$\bar{c^p} = c^p - \sum_{i \in \mathcal{P}} a_i^p d_i - d_0 \tag{5.5}$$

where d_0 represents the dual variable associated with constraint (5.3).

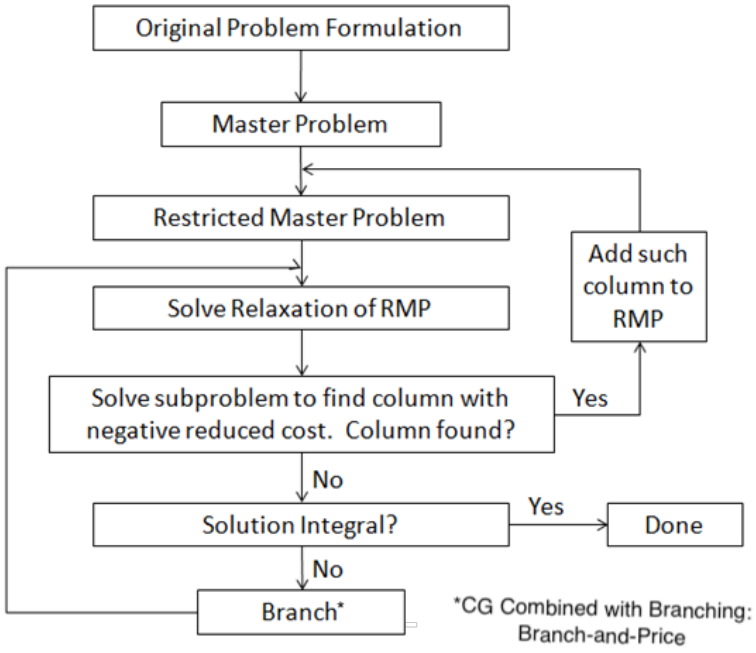


Figure 5.1: Column Generation Outline

Various algorithms can be used to identify eligible columns in the pricing problem, which is evidently a Shortest Path Problem with Resource Constraints (SPPRC). A dynamic programming technique called labelling algorithm, where labels that contains current resource status of the routes are updated dynamically as the partial routes being extended node by node. Labelling algorithms with problem specific dominance rules are often used in CG based exact solution methods (Dumas et al., 1991; Ropke and Cordeau, 2009). In this thesis, the potential of ML techniques to boost the resolution of the pricing problem are investigated. The motivation for this hybrid approach is to avoid solving exactly the pricing problem as a SPPRC, which is known to be NP-hard. Instead, local-search based pricing heuristics such as VNS and LNS combined with ML techniques are used to search for promising columns. While principles and generic frameworks of VNS and LNS have been presented in Chapter 3, DARP-customised applications and hybridisations of these heuristics are presented in detail in this chapter.

5.1.1 Hybrid Metaheuristics and Column Generation Approaches

For large-scale combinatorial optimisation problems, using exact solution (e.g., Branch-and-Price) methods to find the optimal solution could be extremely time-consuming and therefore unrealistic in practical scenarios. As introduced in Chapter 3, great success has been shown in recent VRP studies on algorithms that combine mathematical programming techniques with metaheuristics. More specifically, when it comes to solving VRP and its generalisations, hybrid models that integrate various heuristics within a column generation structure have been proved to be efficient in literature (Prescott-Gagnon et al., 2009; Muter et al., 2010). Note that solution methods that integrate column generation and heuristics can be either exact or approximate. An exact column generation solution method terminates when the optimal solution of the pricing problem has a non-negative reduced cost, where heuristics are used to speed up the pricing process. On the contrary, the heuristic hybrid solution methods discussed in this chapter are by nature approximate methods, given that optimality is often not proven when the algorithms terminate.

Specifically, for solving the DARP, Parragh and Schmid (2013) introduced a hybrid column generation and LNS-VNS algorithm. Solving the pricing problem for DARP is equivalent to solving a Shortest Path Problem (SPP) with pairing, precedence, time-window, maximum ride time and capacity constraints. With the guidance provided by the information (i.e., dual solutions) extracted from CG structure, columns are being populated using various heuristics in a more effective fashion. In Parragh and Schmid (2013)'s hybrid structure, the path-based RMP is solved alternately with its integer constraints enforced (Integer Programming, IP) and relaxed (Linear Programming, LP). When the relaxed RMP is solved as a LP, VNS is applied to each route of the incumbent solution, searching for columns with negative reduced costs with the guidance of dual values. When RMP is solved as an IP, LNS is excuted on the incumbent integer solution as a whole, aiming to make inter-route improvement on the incumbent integer solution while generating and adding more columns to the pool. Temporary deterioration is allowed within LNS to enable the algorithm to jump out of local optima. The process ends when a certain amount of iterations (RMP-solves) have

been executed. Aligning with these late development in hybrid-structured algorithms, this chapter introduces and compares a number of hybridisation schemes involving Metaheuristics, Branch-and-Price and Machine Learning techniques.

5.1.2 Thompson Sampling

Thompson Sampling (TS) (Thompson, 1933), also known as Posterior Sampling or Probability Matching, is a natural Bayesian algorithm originally developed to solve the classic multi-armed bandit problem. In the last few decades, it has been generalised and applied to a broad range of online sequential decision making problems in which the exploration-exploitation trade-off plays an important role. In Thompson Sampling, the agent initialise their uncertainty about the true (unknown) distribution of an arm by setting up a prior distribution. At each step, the idea behind TS is to draw a sample for each option based on its prior distribution, make the decision based on the sampled values, take the action, and update the parameters of the distributions according to the observations afterwards. Thompson Sampling has recently attracted attention within the stochastic optimisation community on a wide range of online decision problems including the SPP.

The Multi-armed Bandit (MAB) problem is a classic online learning problem studied in probability theory, computer science and operations research. Originated from a gambling scenario where a gambler faces a slot machine with multiple levers (or a row of one-armed slot machines) trying to decide which lever to play, the multi-armed bandit problem well represents the dilemma of exploration and exploitation: Should the gambler stick to the lever he/she played in the past that tends to pay high rewards, or should he/she try pulling new levers which might return even higher payoffs, in order to maximise its cumulative reward over time? The MAB framework has been used to model problems where an agent learns about probabilities/rewards of actions by experiments, and tries to optimise its objective over a period of time.

Designed to solve bandit problems, the baseline of Thompson Sampling works as follows: Suppose the agent is learning the unknown expectation of a series of actions θ by taking an action at

each iteration (step) and observing an outcome (reward). The agent initialise a prior distribution p to represent his uncertainty about θ. At each step, the agent draws a random sample from the prior distribution p of each action, and choose the action whose sample reward maximises the objective function. The agent then observes the real reward yield by the applied action, and updates the distribution p accordingly. The structure of generic TS is summarised in Algorithm 6.

Theoretical analyses of Thompson Sampling in recent years also helped to understand better when TS works and why. Russo and Van Roy (2014) established a formal relation between the Upper Confident Bound (UCB) algorithm and Thompson Sampling, which allows the regret bound developed by UCB to be converted into (and therefore compared with) a Bayesian regret bound for TS. The superior performance of TS is presented in Russo and Van Roy (2014)'s simulation results comparing to the UCB algorithm, especially when the action domain is large. Agrawal and Goyal (2012) extended TS for MAB from a discretely distributed reward (e.g. Bernoulli bandit, the reward is either 0 or 1) to continuous distribution (e.g. General stochastic bandit, reward with support [0,1]). This enables the application of various conjugate distributions in TS, including Gaussian priors and the Gaussian likelihood distributions with known variance (Agrawal and Goyal (2013a)).

Algorithm 6: Thompson Sampling

```
for t=1,2,3... do
    /* Sample */
    Sample θ̂ ~ p
    /* Choose action */
    x_t ← argmax_{x∈X} E(r(y_t))|x = x_t
    Observe y_t after applying x_t
    /* Update parameters of p */
    p ← P_{p,q}(θ ∈ ·|x_t, y_t)
```

Back to the set partitioning problem of DARP (5.1)-(5.4) and the structure of column generation based algorithms in Figure 5.1, similarities can be identified between MAB and using CG to solve the RMP: the potential of a column being used in the optimal solution based on its reduced cost

(5.5) is unknown to the algorithm (agent). Information is collected every time the LP relaxation of RMP is solved and dual values d_i are obtained. Dual values d_i do not develop monotonically over iterations. In fact, the oscillation in dual values is especially significant in the first few iterations of RMP-solving because Ω' then contains too few columns to provide valuable dual information (Desrosiers and Lübbecke, 2005). This leads to poor quality of columns being added to Ω' that are not often useful to the RMP. Dual variables oscillation can cause issues in the efficiency of algorithms, especially in approximate algorithms where the stopping criteria involves the number of times RMP being solved and optimality is often not proved at the end.

In this chapter, Thompson Sampling is incorporated within a CG setting for the DARP, where the value (potential) of new columns is unknown at the beginning of solving the RMP. The goal is to use sampling to control the behaviours of dual variables and identify valuable columns sooner in an intelligent way. Therefore, this chapter is attempting to bridge the gap between a competitive machine learning technique and a combinatorial optimisation problem by using TS to iteratively learn the dual values d_i of each request $i \in \mathcal{P}$ associated with constraints (5.2), by observing the updated dual values d_i at each LP Relaxation of RMP-solving iteration.

5.1.3 Chapter Outline

As mentioned before, this chapter explores the integration of TS, Branch-and-Price and hybrid metaheuristic algorithms to scale up DARP. The rest of this chapter is constructed as follows.

Section 5.2 introduces a Branch-and-Price incorporated hybrid metaheurstic solution method. Elements of the hybrid metaheuristics, VNS and LNS, are presented in this section. Their algorithmic specifications as well as how they work together under a CG scheme are explained in detail as they are both integral components of all solution methods presented in this chapter.

In Section 5.3, TS is integrated to the hybrid metaheuristic structure. In this solution method,

integer solutions are obtained by solving RMP as IP (as opposed to use branching in Section 5.2). A revised version of VNS is also introduced in this section to facilitate TS.

In Section 5.4, a comprehensive solution method that integrates branching, TS and hybrid metaheuristics is proposed. Parameter tuning and numerical experiments are conducted and reported in Section 5.5, and the chapter is concluded in Section 5.6.

5.2 Branch-and-Price with Hybrid Metaheuristics

In this section, the hybridisation of Branch-and-price with Metaheuristics is presented. This solution method is refered as B&P-HYB in the following sections. In B&P-HYB, integer solutions are obtained by embedding CG into a branch-and-bound structure (namely the Branch-and-price solution method), as opposed to solving the IP problem of RMP at every other iteration (Parragh and Schmid (2013)). Note that TS is yet to be incorporated in this section.

As presented in Figure 5.2, the B&P-HYB solution method starts off with solving the root node with no branching cuts. Initial solutions are generated using Jaw et al. (1986)'s insertion heuristic as presented in Chapter 3.4.2. At each Branch-and-price iteration, the LP relaxation of RMP is solved, and the global Upper Bound (UB)/Lower bound (LB) are updated whenever better integer/LP solutions are found, respectively. Branching is performed at each node after the RMP is solved and fractional RMP solution(s) exist. Note that branching is executed on a fractional link (i, j), instead of a path, and two children nodes are generated by fixing this link to either 1 or 0. Link and path variables are connected by the following equation:

$$x_{ij} = \sum_{p \in \Omega} b_{ij}^{p} \lambda^{p} \tag{5.6}$$

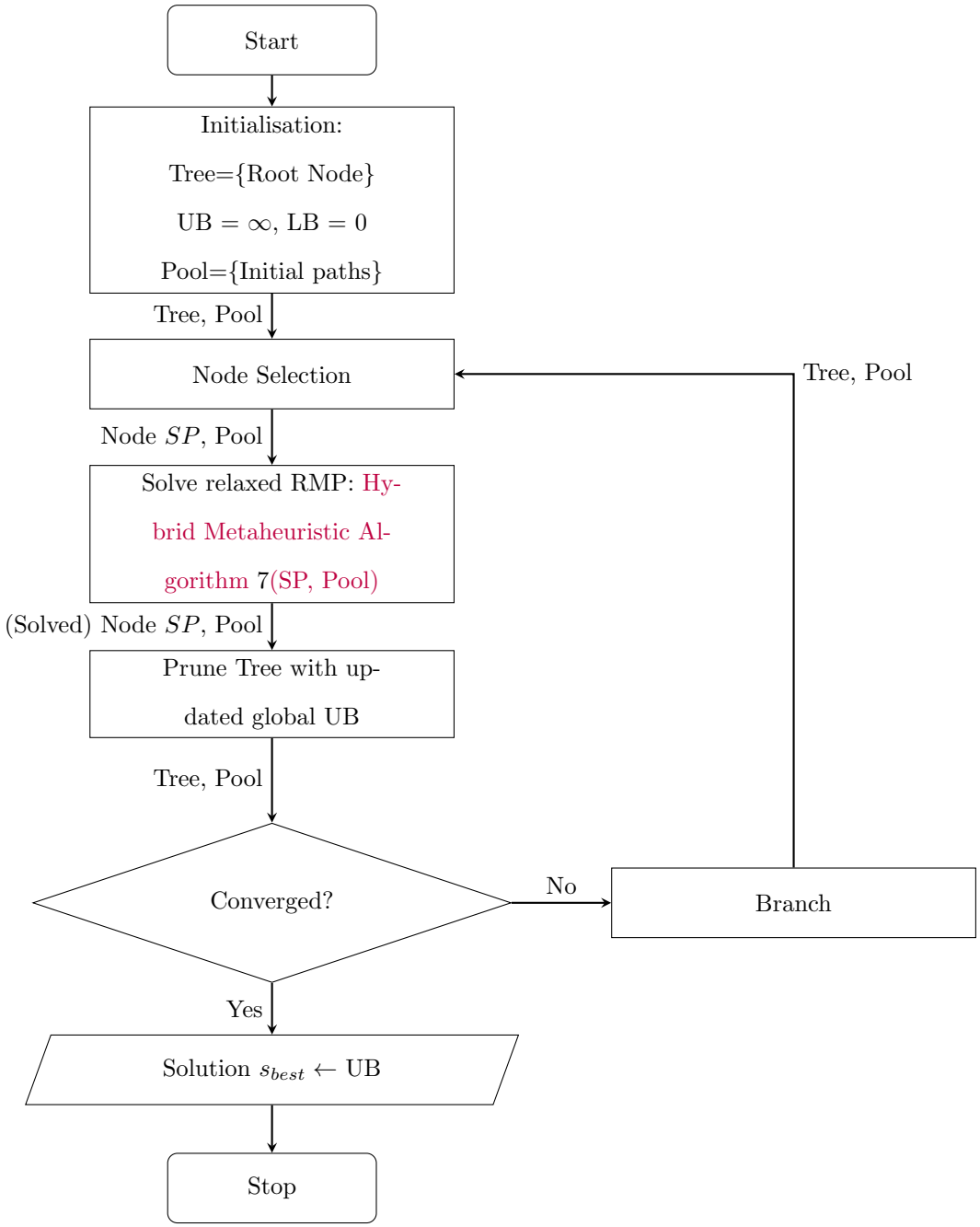


Figure 5.2: B&P-HYB Solution Method Overlook

where $b_{ij}^p = 1$ if link (i, j) is in path p, and 0 otherwise. The branching rule applied in this thesis

is to branch on the variable that is the furthest from the nearest integers. The process terminates when UB and LB converges or all nodes in the tree have been solved.

Algorithm 7: Hybrid Heuristic Framework used to solve RMP in B&P-HYB

Input: Selected Node SP [LB, UB], Pool

Output: Updated SP, Pool

/* Initialisation */

1 *Added Columns* ←{dummy variables}

2 Pool'= Subset of Pool given the branching constraints at this Node

/* Solve RMP */

3 **while** *Added Columns* $\neq \emptyset$ **do**

4 Solve the relaxed RMP at node SP (wrt branching constraints of the node), given the domain of columns in Pool'

5 Obtain Primal solution λ_p and its positive-flow routes $\boldsymbol{s_p}$ ($\forall \lambda_p > 0$), Dual (d_i) solutions and the *Objective value* of relaxed RMP

6 **for** $p_m \in \boldsymbol{s_p}$ **do**

7 *Added Columns*$_{VNS}$ = Algorithm 8: VNS (p_m, d_i)

8 **if** *Objective value*< $SP[LB]$ **then**

9 Update $SP[LB] \leftarrow$ *Objective value*

10 **if** λ_p *is integer* **then**

/* Better integer solution found */

11 Integer Solution routes $\boldsymbol{s_{int}}$ obtained

12 $s_{best} \leftarrow \boldsymbol{s_{int}}$

13 s_{best}, *Added Columns*$_{LNS}$ ←Algorithm 9: LNS($\boldsymbol{s_{int}}$)

14 *Added Columns* ← *Added Columns*$_{VNS} \cup$ *Added Columns*$_{LNS}$

15 Update $SP[UB] \leftarrow c_{s_{best}}$

16 Pool←Pool ∪ *Added Columns*

17 Pool'←Pool' ∪ *Added Columns*

When the gap between LB and UB is not yet converged at the end of a Branch-and-price iteration and fractional solutions exist, we take one of the non-integer paths used in the optimal solution of the relaxed RMP, and branch on one of its links. This results in the generation of two new children nodes in the Branch-and-price tree, with this fractional link set to "0" and "1", respectively. Since the set partitioning formulation (5.1)-(5.4) is path-based, setting a link to "0" or

"1" is implemented by screening all paths in the pool that contains the link using Equation (5.6), and disposing the ones of which b_{ij}^p are not "0" or "1". Further down the exploration of the tree, each node would have a list of links that are fixed to either 0 or 1. It is worth mentioning that the pool needs to be filtered before RMP is solved at each node, leaving only columns satisfying all the fixed links requirements in the (temporary) pool for this node. These fixed-0 and fixed-1 links are also passed on to the subproblem solved by the heuristics: the new columns generated using VNS and LNS at one node must satisfy all the "fixed-0s" and "fixed-1s" of corresponding links at this node. Note that branching could yield truly infeasible nodes with conflicting fixed links (e.g., both link (1,2) and (1,3) are fixed to 1 at a node, which violates the flow conservation constraint of DARP). This infeasibility will be caught by the subproblem: an initial solution is generated using dummy variables if there is no feasible solution found from the pool. When no new column is generated after a certain number of CG iterations, we deem this node to be truly infeasible.

Metaheuristics Large Neighbourhood Search (LNS) and Variable Neighbourhood Search (VNS) applied for generating columns within the subproblem will be explained in detail in this section. The structure of the Hybrid Metaheuristics used for column generation is demonstrated in Algorithm 7.

In each subproblem-solving iteration (Algorithm 7), the linear relaxation of RMP is first solved (in this thesis, using commercial solver CPLEX) and primal (λ) and dual (d_i) solutions are then obtained. VNS is then applied to each individual (positive-flow) route of the solution to the RMP (LP), identifying columns with negative reduced costs by add, removing or exchanging requests. More specifically, within VNS, a number of neighbourhoods (of various structures) of the input column will be explored. A "shaking" phase (Line 5-15 in Algorithm 8) in VNS is where the incumbent column is modified in a series of predefined neighbourhood structures N_k=$(N_1, N_2, ..., N_{k_{max}})$. The neighbourhoods are usually ordered in a nested way. This thesis defines the number of neighbourhoods to be explored in VNS is $k_{max} = 9$. For each neighbourhood N_k, two factors help shape up the neighbourhood from the incumbent path: the 'Operator' applied and the 'Number of requests

involved' Num_k. In this case, the number of requests involved is defined as a certain percentage of the number of existing requests in the incumbent path (Num_p): $Num_k = \lceil Num_p \times (20\lceil k/3 \rceil)\% \rceil$, meaning that when $k = 1, 2, 3$, 20% of the existing route will be affected by the operator; when $k = 4, 5, 6$, 40% of the existing route will be affected by the operator; etc. The operators, on the other hand, are placed in a nested fashion in the neighbourhoods. To sum up, the neighbourhood structure N_k is defined as follows:

- when k=1,4,7: 20%, 40%, 60% of the number of existing requests on the incumbent path are *added* to the incumbent path;

- when k=2,5,8: 20%, 40%, 60% of the number of existing requests on the incumbent path are *exchanged* on the incumbent path;

- when k=3,6,9: 20%, 40%, 60% of the number of existing requests on the incumbent path are *removed* from the incumbent path.

VNS exploration iterates through neighbourhoods from N_1 to $N_{k_{max}}$, then starts over from N_1 again if all neighbourhood structures have been explored. Num_k requests along with an operator are randomly selected in each VNS iteration to define the neighbourhood structure N_k. In each neighbourhood N_k, when applying an operator on the incumbent path, the probability of each request i being added to or removed from the incumbent path p is related to its dual value d_i: the probability of each request i being added to the incumbent path as proportional to d_i, whereas the probability of a request being removed from the incumbent path as inversely proportional to d_i.

Algorithm 8: Generate columns using VNS

Input: Column p_m and Dual (d_i) solutions of RMP

Output: Columns (paths) with negative reduced cost ('Added Columns')

`/* Initialisation */`

Added Columns $\leftarrow \emptyset$

$k \leftarrow 1, k_{max} \leftarrow 9$

`/* VNS generates columns */`

$c_{incumbent} \leftarrow p_m$

while $|Added\ Columns| \leq N^{new}$ **do**

  `/* VNS ‘‘Shaking" */`

  Num_p = Number of requests currently in the incumbent column p, $Num_k = \lceil Num_p \times (20\lceil k/3 \rceil)\% \rceil$

  Select *Operator* (according to k)

  `/* Define the probability of each request being added / removed / exchanged */`

  Rank d_i from highest to lowest.

  For each request that is *not* in $c_{incumbent}$, a probability p_i, proportional to the index of d_i in the ranked list, is introduced.

  For each request that is currently in $c_{incumbent}$, a probability q_i, *inversely* proportional to the the index of d_i in the ranked list, is introduced.

  $m \leftarrow 1$

  while $m \leq Num_k$ **do**

    Select request r based on *Operator* and p_i (or q_i) of all requests

    $c' \leftarrow$ apply *operator* to r on $c_{incumbent}$

    if $\bar{c}_{c'} < 0$ **then**

      Added Columns $\leftarrow$ Added Columns$\cup\{c'\}$

    `/* VNS ‘‘Move or not" */`

    if $\bar{c}_{c'} < \bar{c}_{c(c_{incumbent})}$ **then**

      $c_{incumbent} \leftarrow c'$

      $k \leftarrow 1$

    else

      $k \leftarrow k + 1$

    if $k > k_{max}$ **then**

      $k \leftarrow k - k_{max}$

    $m \leftarrow m + 1$

In each neighbourhood N_k, requests are selected according to their predefined probabilities to be added /removed/exchanged from the incumbent path. In the exploration process, all generated feasible paths with a negative reduced cost, regardless of the percentage of the route affected, are added to the pool from which the RMP will be iteratively solved. When the required number of requests have been modified in the incumbent path, the modified path is then compared with the current incumbent path. If the new path supersedes the incumbent path, it then replaces the incumbent path, k is reset to 1 and VNS restarts from the first neighbourhood structure; Otherwise, the incumbent path remains unchanged, and VNS continues on to explore the $k+1$ neighbourhood. This is called the "Move or not" phase in VNS (Line 22-26 in Algorithm 8). VNS terminates when a certain number of columns have been identified and added to the pool.

In B&P-HYB, LNS is executed *every time an integer solution is found* by solving the relaxed RMP (Line 9 in Algorithm 7). Note that the master problem solved in each pricing iteration is always the linear relaxation of the RMP (LP), and integer solution is marked when the solution happens to be integer, without integer constraints enforced on variable λ^p. The solution, containing several routes, is treated as a whole within LNS, as opposed to how each path in the solution was perturbed separately in VNS.

In each LNS iteration (as presented in Algorithm 9), the algorithm first determine randomly the number of requests q involved in this round of LNS. A removal operator and an insertion operator are also randomly selected in each iteration. Next, q requests are selected and removed using the randomly selected removal operator. These removed requests are then re-inserted back into the solution using the selected insertion operator. After inserting all requests back, if the newly generated solution is better (i.e., yields lower routing cost) than the incumbent solution, it replaces the incumbent solution. Several removal and insertion operators are available to destroy the incumbent integer solution and repair it again, searching for improved solutions.

Algorithm 9: LNS

Input: Integer solution s_{int}

Output: s_{best}, Columns (paths) with negative reduced cost ('Added Columns')

/* Initialisation */

$Added\ Columns \leftarrow \emptyset$

$s_{best} \leftarrow s_{int}$

$s_{incumbent} \leftarrow s_{int}$

$M \leftarrow 1$

/* LNS */

while $M \leq \bar{M}$ **do**

  $q \leftarrow$ random integer within range $(0.1n, 0.5n)$

  $f \leftarrow$ random integer within range $(1, 3)$

  if *f=1* **then**

    Remove q requests from s using "Random Removal"

  if *f=2* **then**

    Remove q requests from s using "Shaw Removal"

  if *f=3* **then**

    Remove q requests from s using "Worst Removal"

  $g \leftarrow$ random integer within range $(1, 2)$

  if *g=1* **then**

    Insert back the q removed requests using "Greedy Insertion",obtain s'

  else

    Insert back the q removed requests from s using "Regret Insertion",obtain s'

  for *each path p in* s' **do**

    Add p to *Added Columns*

  if $c'_s < c_s$ **then**

    $s_{best} \leftarrow LS(s')$

  if *accept(s,s')* **then**

    $s_{incumbent} \leftarrow LS(s')$

Aligning with Parragh and Schmid (2013)'s hybrid model for DARP, the destroy/removal operators applied in LNS in this thesis are:

- Shaw removal: One request is first selected randomly as the seed requests. The selection of the rest $q-1$ requests is inclined towards those that are more related to the seed requests. The definition of relatedness is often problem-specific. In this thesis, relatedness between two requests i and j is deemed to be high if $(|B_i - B_j| + |B_{n+i} - Bn + j| + t_{ij} + t_{n+i,n+j})$ is small.
- Random removal: Randomly remove q requests from incumbent solution.
- Worst removal: Each request is switched off one by one in the incumbent solution, and the routing cost is calculated without this request. All requests are ranked by how much routing cost it can "save" by being removed from the route. Requests that save the most are more likely to be selected and removed.

On the other hand, the repair/re-insertion operators applied in LNS in this thesis are:

- Greedy insertion: The removed requests are ranked (then inserted) by the deterioration it causes at its best insertion position. The request with minimal deterioration is inserted first.
- k-regret insertion: The regret value of a request is defined as the difference in routing cost between when a request is inserted in its best routes and when it is inserted in its second best route (both at best insertion positions). k-regret heuristic insertion prioritises inserting requests that has higher summed-up regret values of second best route, third best route, ..., k^{th} best route(i.e., prioritising inserting requests with higher $\sum_{k_0 \in k}(c_{k_0 th\ \texttt{best route}} - c_{\texttt{best route}})$ first). When this operator is selected by LNS, k is a random number generated in the range of (2, K-1).

In LNS, whenever a new feasible solution is identified, a simple intra-route local search is triggered to locally improve the solution. Furthermore, to avoid being stuck in local optima, temporary deterioration can sometimes be allowed in incumbent solution updates. Specifically, a feasible solution that is at most 3% worse than the incumbent solution could be accepted as the new incumbent

with a probability of 1%.

This proposed hybrid Branch-and-Price structure (5.2) combines two powerful metaheuristic algorithms with column generation embedded in a branching scheme. Numerical results that compare the performance of this hybridisation against others will be presented in Section 5.5.

5.3 Column Generation with Hybrid Metaheuristics and Learning

Buidling on the hybrid structure in Section 5.1.1, this section explores potential ways of integrating Thompson Sampling (TS) into hybrid metaheuristic solution method for DARP. As discussed in Section 5.1.2, the idea is to model the values of dual variables associated with constraints (5.2) to control the oscillation of dual variables and improve the rate of convergence of the CG procedure to the RMP pool Ω'. TS is applied to model dual variables d_i by assuming a prior distribution p_i for each $i \in \mathcal{P}$, and updating the distributions after solving the RMP and observing the dual values in each iteration. The goal is to learn the "potential" of the columns by sampling dual values to find better solutions in a more efficient way. To achieve this, a CG and TS based hybrid solution scheme is proposed to solve the DARP in this section. This solution method is referred as HYB-TS in the rest of the chapter.

As discussed in previous chapters, when using CG techniques to solve the DARP, the subproblem of the RMP in Figure 5.2 is essentially a Shortest Path Problem with Resource Constraints (SPPRC) where the goal is to identify paths (columns) with negative reduced costs from an origin node to a destination node on a directed graph with respect to DARP constraints. The RMP is deemed to be solved to optimality if extensive search is conducted and no column with a negative reduced cost can be found. In this HYB-TS Metaheuristics scheme, we model the values of dual variables d_i related to each request $i \in \mathcal{P}$ (associated with constraints (5.2)). Consider a prior for which each d_i

is Gaussian distributed with parameters $d_i \sim N(\mu_i, \sigma_i^2)$. In each step t, an action x_t is a sequence of links selected in the graph that starts from the origin node and ends at the destination node. After action x_t is applied, the reduced cost of this path $\tilde{c^p}$ is observed by independently sampling the reduced cost of each link in the path:

$$\tilde{c^p} = c^p - \sum_{i \in \mathcal{P}} a_i^p \tilde{d}_i - d_0 \tag{5.7}$$

where $\tilde{d}_i$ is randomly drawn from Normal distribution with parameters μ_i and σ_i^2, and d_0 represents the dual variable associated with constraint (5.3). To integrated TS into the metaheuristic hybrid structure, the VNS procedure described in Algorithm 8) is modified to incorporate sampling: The action set from which actions are applied and new paths are generated is determined by the operators in VNS. *Best insertion* (based on the samples $\tilde{d}_i$) is applied in each perturbation of the incumbent column by action x_t, and the incumbent column is updated if the new column generated by action x_t yields a lower reduced cost. If the observed reduced cost $\tilde{c^p}$ is negative, this column yielded by action x_t is added to the column pool of RMP. Parameters μ_i and σ_i^2 are updated in each iteration after RMP is solved and new dual solutions $\hat{d}_i$ are obtained:

$$\mu_i \leftarrow \frac{\frac{\mu i}{\sigma_i^2} + \frac{\hat{d}_i}{\sigma_L^2}}{\frac{1}{\sigma_i^2} + \frac{1}{\sigma_L^2}} \qquad \forall i \in \mathcal{P} \tag{5.8}$$

$$\sigma_i^2 \leftarrow \frac{1}{\frac{1}{\sigma_i^2} + \frac{1}{\sigma_L^2}} \qquad \forall i \in \mathcal{P} \tag{5.9}$$

Note that since only one of the two parameters could be estimated in conjugate distributions while the other one is assumed known, this thesis assumes that the real variance σ_L^2 of the prior distribution is known for each request i: $\sigma_L^2 = 1$. This section first discusses the integration of TS, CG and metaheuristic hybridisation (without branching). The summary of this HYB-TS algorithm is summarised in Algorithm 10. Aligned with Parragh and Schmid (2013)'s work, an integer solution is obtained by solving the IP of RMP once every N^{int} iteration.

Algorithm 10: TS-Metaheuristics Hybridisation (HYB-TS)

Input: s_0, $\boldsymbol{\mu}$, $\boldsymbol{\sigma^2}$

Output: s_{best}

/* Initialisation */

1 Initialise pool Ω' with all paths in initial solution s_0

2 $m \leftarrow 1, s_{best} \leftarrow s_0$, $d_i \sim N(\mu_i, \sigma_i^2)$

3 Added Columns $\leftarrow \emptyset$

4 **while** $m \leq N^{iterations}$ **do**

5 Solve the LP relaxation of RMP on Ω' yielding s' and dual values $\hat{d}_i$ $\forall i \in \mathcal{P}$

6 Update parameters $\forall i \in \mathcal{P} : \mu_i = \frac{\frac{\mu_i}{\sigma_i^2} + \frac{\hat{d}_i}{\sigma_L^2}}{\frac{1}{\sigma_i^2} + \frac{1}{\sigma_L^2}}$, $\sigma_i^2 = \frac{1}{\frac{1}{\sigma_i^2} + \frac{1}{\sigma_L^2}}$

7 **for** *each path* $p_m \in s'$ **do**

8 Apply Algorithm 11 VNS-TS on p_m and add up to N^{new} columns to Ω'

9 **if** *(i mod* N^{int}*)=0* **then**

10 Solve IP of RMP on Ω' yielding integer solution s^*

11 Apply Algorithm 9 LNS on s^* yielding s'' and add all columns to Ω'

12 $m \leftarrow m + 1$

13 Solve IP of RMP on Ω' yielding s_{best}

In Algorithm 10, in each iteration i, parameters μ_i and σ_i^2 get updated every time the LP of RMP is solved and dual values $\hat{d}_i$ observed by the agent (line 5-6). The updated parameters are then passed on to the VNS-TS Column Generation process (Algorithm 11).

Algorithm 11: Generate columns using VNS-TS (Within HYB-TS)

Input: p_m, $\boldsymbol{\mu}$, $\boldsymbol{\sigma^2}$

Output: Columns (paths) with negative reduced cost ('Added Columns')

/* Initialisation */

Added Columns $\leftarrow \emptyset$

$k \leftarrow 1, k_{max} \leftarrow 9$

/* VNS generates columns */

$c_{incumbent} \leftarrow p_m$

while $|Added\ Columns| \leq N^{new}$ **do**

  /* VNS ''Shaking" */

  $Num_p \leftarrow$ Number of requests currently in the incumbent column p

  $Num_k \leftarrow \lceil Num_p \times (20\lceil k/3 \rceil)\% \rceil$

  Select *Operator* (according to k)

  for $i \in P$ **do**

    Sample dual value : $\tilde{d}_i \sim N(\mu_i, \sigma_i^2)$

  Rank d_i^s from highest to lowest.

  Select list of requests $R = [r_1, ..., r_{Num_k}]$ based on the selected *Operator* and the $\tilde{d}_i$ ranking.

  $m \leftarrow 1$

  while $m \leq Num_k$ **do**

    $c' \leftarrow$ apply *Operator* on $c_{incumbent}$ with the m^{th} request in list R

    if $\bar{c}_{c'} < 0$ **then**

      Added Columns $\leftarrow$ Added Columns$\cup\{c'\}$

    /* VNS ''Move or not" */

    if $\bar{c}_{c'} < \bar{c}_{c(c_{incumbent})}$ **then**

      $c_{incumbent} \leftarrow c'$

      $k \leftarrow 1$

    else

      $k \leftarrow k + 1$

    if $k > k_{max}$ **then**

      $k \leftarrow k - k_{max}$

    $m \leftarrow m + 1$

The sampling step of TS is executed in VNS-TS (Algorithm 11): when searching for columns

with negative reduced cost, dual values $\tilde{d}_i$ are drawn in each VNS iteration. The sampled $\tilde{d}_i$ for each request i are then ranked from the highest to lowest. Based on the calculation of the reduced cost for columns, a request with a higher (positive) dual value is more likely to lower the overall reduced cost. Therefore, an ordered list of requests R is selected for the Operators to be applied on in the VNS procedure (Algorithm 11 line 15-16). For example, if the operator selected for the current iteration is ADD, then list R is composed of the first Num_k requests (which are currently not in the incumbent column) with the highest sampled $\tilde{d}_i$; If the operator selected for the current iteration is REMOVE, then list R is composed of the last Num_k requests (which are currently in the incumbent column) with the lowest $\tilde{d}_i$. Note that this is to be distinguished from how Parragh and Schmid (2013) randomised the request selection when perturbing the incumbent column: in this model, TS introduces randomness to VNS which avoids the same column being generated repetitively. Since list R is selected based on the sample value $\tilde{d}_i$ drawn from the prior distribution, the rankings (and requests in R) will vary after each iteration of sampling. The list R, together with the selected operator, facilitate the search for negative reduced cost columns in the VNS procedure (Algorithm 11 line 18-21). The reduced cost of each new column is calculated using (5.7) based on the sampled dual values.

From a TS perspective, it is possible that the agent observes the outcome of the dual values less often than the agent samples the dual values and applies an action that optimises the reward. In this model, the agent observes an outcome in each LP-RMP solving iteration (line 5-6 in Algorithm 10). However, the sampling and action selection can take place more than once every time the LP-RMP is solved, given that the number of actions taken is controlled by N^{new} in VNS-TS. N^{new} could be set to a small value if the agent intends to keep up the observations with sampling. However, this could obstruct the performance of VNS by limiting the exploration of the neighbourhood in its search. Therefore, these parameters need to be tuned or even updated adaptively. Parameter tuning and sensitivity analyses are discussed in Section 5.5.

5.4 Branch-and-Price with Hybrid Metaheuristics and Learning

To set up the control group in the experiments in order to understand the effect of TS and the branching scheme separately, a Branch-and-Price with Hybrid Metaheuristics and Learning Algorithm is also designed and implemented. This algorithm (referred to as B&P-HYB-TS) incorporates the Hybrid Learning algorithm described in Section 5.3 into a Branch-and-Bound structure. This B&P-HYB-TS Solution Method is summarised in Figure 5.3. Compared to B&P-HYB (Figure 5.2), the embedded Hybrid Metaheuristic in this section integrated Thompson Sampling within the VNS Procedure (See Algorithm 12).

Algorithm 12: Hybrid TS-Heuristic Framework used to solve RMP (Within B&P-HYB-TS)

Input: Selected Node SP [LB, UB], Pool
Output: Updated SP, Pool

/* Initialisation */

$Added\ Columns \leftarrow$ {dummy variables}
Pool'= Subset of Pool given the branching constraints at this Node

/* Solve RMP */

while *Added Columns* $\neq \emptyset$ **do**
  Solve the LP relaxation of RMP on Ω' yielding s' and dual values $\hat{d_i}\ \forall i \in \mathcal{P}$
  Update parameters $\forall i \in \mathcal{P} : \mu_i = \frac{\frac{\mu_i}{\sigma_i^2} + \frac{\hat{d_i}}{\sigma_L^2}}{\frac{1}{\sigma_i^2} + \frac{1}{\sigma_L^2}}$, $\sigma_i^2 = \frac{1}{\frac{1}{\sigma_i^2} + \frac{1}{\sigma_L^2}}$
  for *each path* $p_m \in s'$ **do**
    Apply Algorithm 11 VNS-TS on p_m and add up to N^{new} columns to Ω'
  if *Objective value*$<$ *SP[LB]* **then**
    Update $SP[LB] \leftarrow Objective\ value$
  if λ_p *is integer* **then**
    /* Better integer solution found */
    Integer Solution routes $\boldsymbol{s_{int}}$ obtained
    $s_{best} \leftarrow \boldsymbol{s_{int}}$
    s_{best}, $Added\ Columns_{LNS} \leftarrow$Algorithm 9: LNS($\boldsymbol{s_{int}}$)
    $Added\ Columns \leftarrow Added\ Columns_{VNS} \cup Added\ Columns_{LNS}$
    Update $SP[UB] \leftarrow c_{s_{best}}$
  Pool$\leftarrow$Pool $\cup$ *Added Columns*
  Pool'$\leftarrow$Pool' $\cup$ *Added Columns*

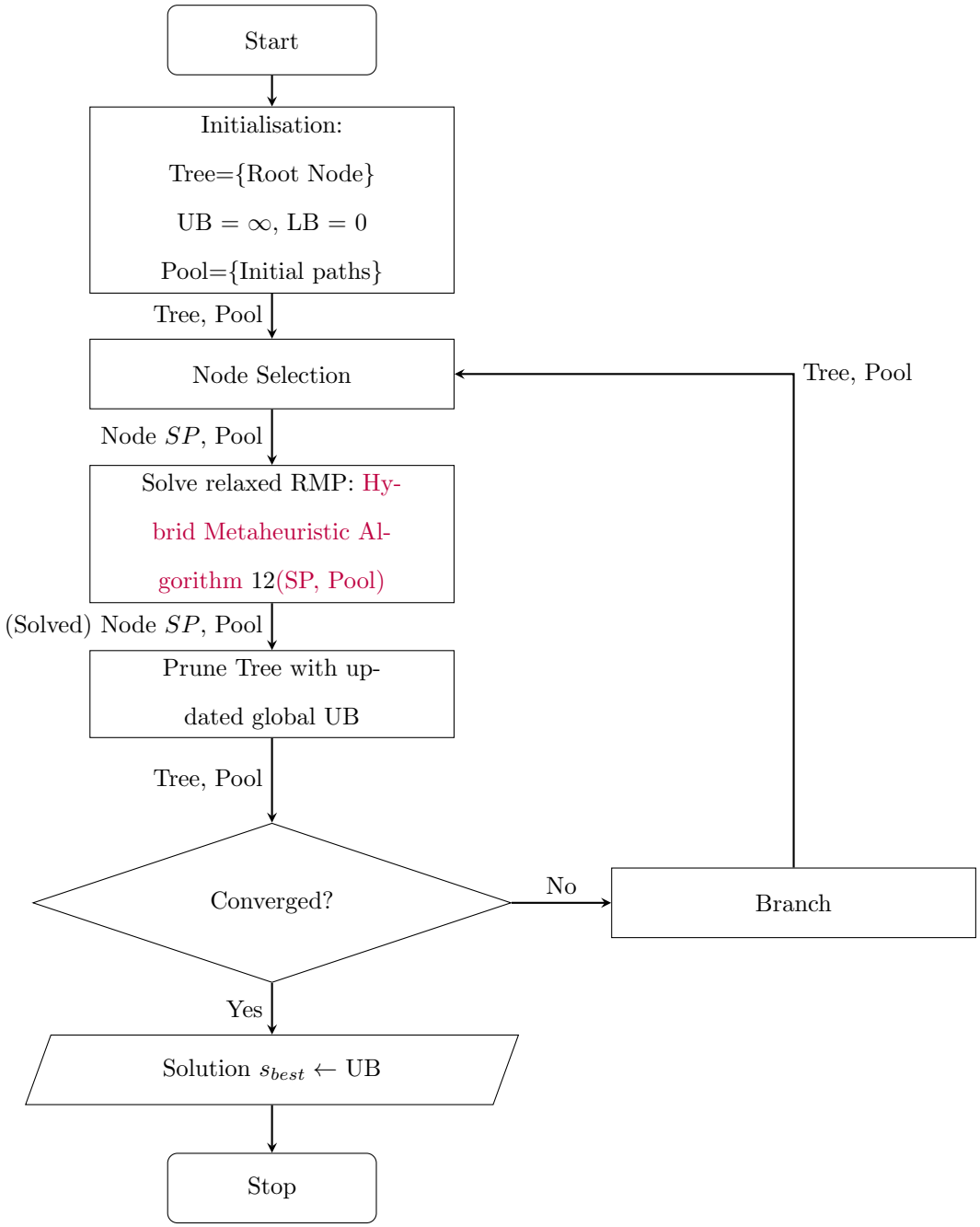


Figure 5.3: Branch-and-Price TS Solution Method Overlook (B&P-HYB-TS)

5.5 Numerical Experiments

5.5.1 Algorithms Summary

In this chapter, Branch-and-Price (B&P) Structure and/or Thompson Sampling (TS) were incorporated to a Column Generation based Hybrid Metaheuristic solution method (Parragh and Schmid, 2013), in search of better solution algorithms to solve the DARP. The algorithms described and implemented in this chapter (followed by where the descriptions are located in this chapter in brackets) are summarised in Table 5.1 below.

	No Learning	Learning
No Branching	HYB (5.1.1)	HYB-TS (5.3)
Branching	B&P-HYB (5.2)	B&P-HYB-TS (5.4)

Table 5.1: Algorithms Summary

This chapter benchmarks Parragh and Schmid (2013)'s Hybrid metaheuristic (referred as "HYB" in Table 5.1) algorithm as the base solution method, and proposes three algorithms aiming to refine the efficiency of solving large-scale DARP problems. In Section 5.2, a Branch-and-Price Hybrid Metaheuristic (B&P-HYB) structure is proposed by incorporating a Branch-and-Bound scheme to Parragh and Schmid (2013)'s CG based hybrid metaheuristic algorithm; In Section 5.3, TS is incorporated within a learning based algorithm (referred to as HYB-TS). Last but not least, a comprehensive algorithm that embeds HYB-TS into a Branch-and-Price structure (B&P-HYB-TS) is introduced in Section 5.4. Key parameters tuning and the performances of each of these four algorithms are presented in the rest of this Section.

5.5.2 Parameters Tuning

There are many parameters across the four algorithms mentioned in Table 5.1, some of which are shared among all algorithms while others only exist in one or two of these algorithms. In this section, tuning of some key parameters is reported. Note that the focus of this chapter is the application of ML techniques in DARP solution algorithms. Therefore, the parameter tuning also revolves around the core algorithm HYB-TS. For Parragh and Schmid (2013)'s Hybrid metaheuristic (HYB) algorithm, this thesis sticks to their selection of parameters, as it has been proved and confirmed to be performing well in their HYB structure.

Hybrid Metaheuristic Parameters

In this section, N^{new} and N^{int} tuning is presented with one of Cordeau (2006)'s benchmark instance ($K = 7$, $N = 84$). N^{new} is the maximum number of columns added in every VNS iteration (Algorithm 11). In Parragh and Schmid (2013)'s work (Algorithm 8), N^{new} is set to 2 to limit the time spent in each VNS iteration. Parragh and Schmid (2013) compensate the fact that the number of columns generated by VNS is quite limited with a high N^{LNS} value to maintain a reasonable column generating rate. The downside of setting $N^{LNS} = 1000$ (Parragh et al., 2009) is that LNS takes up a large proportion of time in the algorithm, especially when a global time limit is enforced. In this thesis, a time limit of an hour is enforced on all experiments, in order to test the performance of each algorithm in Table 5.1 against dual value oscillation. N^{int}, on the other hand, refers to the RMP being solved as an IP at every N^{int} iteration (in Algorithms HYB and HYB-TS) to obtain integer solutions. Note that this parameter does not exist in B&P-structured Algorithms, since integer solutions are obtained by branching. In Parragh and Schmid (2013)'s work (Algorithm 8), N^{int} is set to 2, indicating that the RMP is solved as an IP at every other RMP iteration, exploiting LNS explorations. For HYB-TS, the goal is to exploit the learning process well enough for it to provide guidance to build new paths. This needs a number of RMP (i.e., (μ, σ^2) updates) iterations (the more the better) to enrich the information accumulation for TS before it can boost the performance of column generation. However, every IP solve tends to take up a lot of time, especially when N^{LNS} is set to a high value. In this section, N^{new} and N^{int} are tuned for the

HYB-TS (with N^{LNS} reduced to 500).

Instance	K	N	N^{new} (VNS)	N^{int}	(Avg.) Objective	CPU time	RMP Solves	Pool Size
N^{new} (VNS) Tuning								
a7-84	7	84	2	10	1281.73	3600†	45	4443
a7-84	7	84	5	10	1246.15	3600†	45	8824
a7-84	7	84	10	10	**1216.82**	3600†	39	11564
a7-84	7	84	25	10	1278.25	3600†	36	11775
N^{int} Tuning								
a7-84	7	84	10	2	1349.10	3600†	8	3368
a7-84	7	84	10	5	1345.35	3600†	22	9399
a7-84	7	84	10	10	**1216.82**	3600†	39	11564
a7-84	7	84	10	20	1223.35	3600†	71	11179

†: Timed out

Table 5.2: N^{new} (VNS) and N^{int} Tuning

Results of tuning experiments are presented in Table 5.2, where the average objective value, number of RMP solves and pool size over 5 runs are reported. In Parragh et al. (2009)'s hybrid model, $N^{new} = 2$ (VNS) and $N^{int} = 2$, which means that in each VNS iteration, a maximum of 2 columns can be added, and RMP is solved as an Integer Programming (IP) in every other RMP solve. Increasing N^{new} could increase the pool size by allowing VNS to add more paths to the pool. However, if N^{new} is set too high, VNS takes up a larger proportion of the whole process, which affects the quality of solution negatively since evidently LNS is more efficient in adding more "valuable" columns that constitute better integer solutions. On the other hand, setting N^{new} too high also drives down the total number of RMP solves, which is undesirable from a TS perspective, as a certain amount of explorations and learning is necessary.

Increasing N^{int} entails solving the RMP as an IP less frequently, which means that LNS is excuted less often. One benefit of having a larger N^{int} is that it sets aside more time for LP RMP solving and prior distribution updating. However, this could also lead to missing out on potential good integer solutions and mutation of these promising integer solutions (LNS), which is proved to

be efficient in improving the quality of the solutions in such hybrid structures.

The goal of the tuning experiments is to find parameter configurations that lead to high quality solutions without requiring extensive computational resources (i.e. time). The configuration that gives the best objective value (a.k.a minimal routing cost) on average is marked out in bold in Table 5.2. In the rest of this Section, the parameter setting of $N^{new} = 10$, $N^{int} = 10$, $N^{LNS} = 500$ is used for HYB-TS.

Thompson Sampling Parameters

This thesis considers a prior for which each d_i is Gaussian distributed with parameters $d_i \sim N(\mu_i, \sigma_i^2)$. Since only one of these parameters can be estimated in conjugate distributions, it is assumed that variance $\sigma_L^2 = 1$ is known. In this subsection, variance σ_i^2 is initialised to different values while μ_i is estimated iteratively from observations and used to assist identifying new columns in HYB-TS.

Instance	K	N	μ	σ	(Avg.) Objective	CPU time	RMP Solves	Pool Size
a7-84	7	84	0	1	1286.20	3600†	38	9999
a7-84	7	84	0	10	1247.36	3600†	42	8772
a7-84	7	84	0	100	**1216.82**	3600†	39	11564
a7-84	7	84	0	1000	1250.41	3600†	48	4405
a7-84	7	84	0	10000	1282.07	3600†	48	5126

†: Timed out

Table 5.3: N^{new} (VNS) and N^{int} Tuning

In Figure 5.4, the shapes of the posterior distributions of different initial variances (after the algorithm terminates by reaching the 1-hour time limit) are presented. Evidently, the more data points observed (RMP solves), the higher the precision of the estimated mean. It is worth mentioning that more RMP solves also lead to more steep (lower variance) curves, as observed in Figure 5.4. Lower initial variances of the prior distributions result in steeper bell shapes even only after a few iterations, forcing the algorithm to exploit known experiences rather than explore other pos-

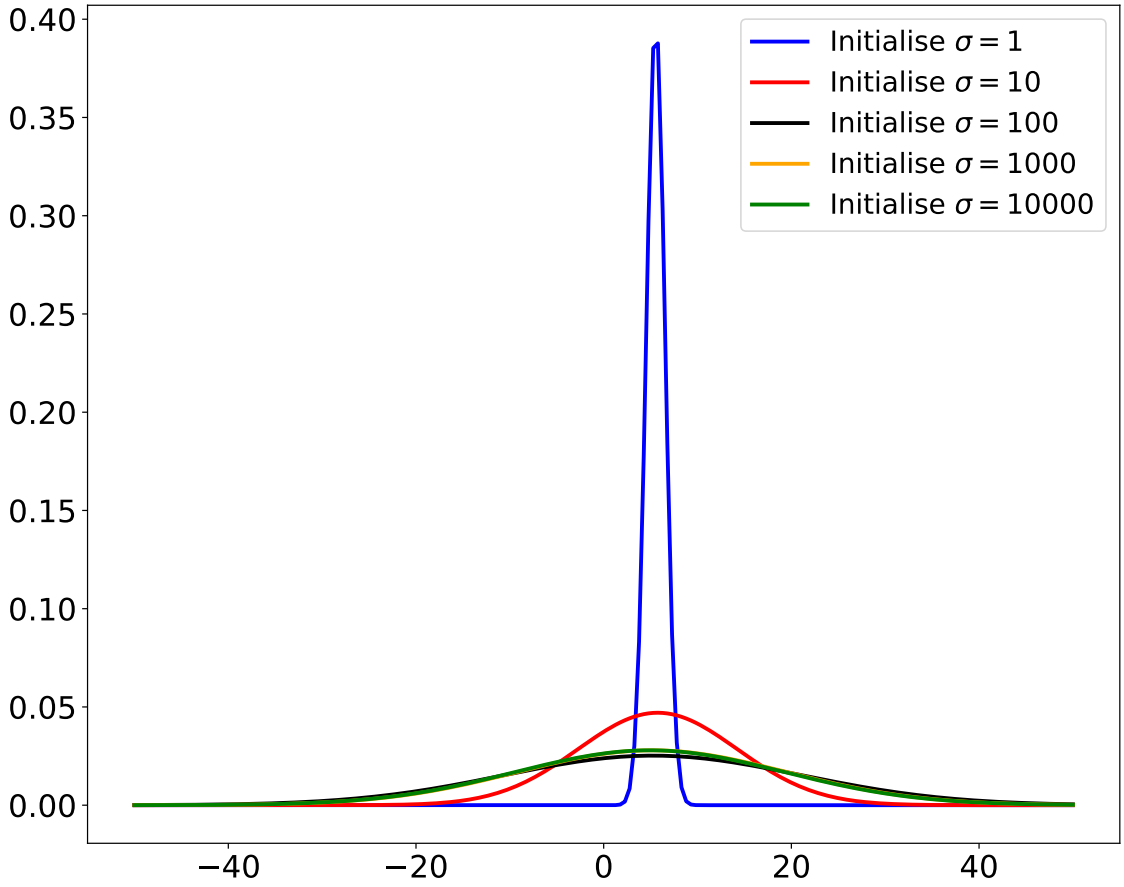


Figure 5.4: Posterior Distribution of different standard deviation σ

sibilities. On the contrary, higher initial variances of prior distributions lead to flatter bell shapes with heavier tails. This brings more randomness in sampling and more likely to find new columns and escaping from local optima. The downside of having a high variance, however, is that the effect of learning is limited. Based on the average objective values returned by different variances in Table 5.3, σ is set to 100 for the rest of this thesis. In other words, dual values of all requests are initialised with a prior distribution $d_i \sim N(0, 10000)$.

5.5.3 Numerical Experiments

In this section, numerical experiments and results are reported on all four algorithms presented in this chapter (summarised in Table 5.1). Instances from Cordeau (2006) are used to test and benchmark the performance of the proposed algorithms. In Table 5.4 and Table 5.5, the performance of HYB, HYB-TS, B&P-HYB and B&P-HYB-TS is reported. All algorithms are implemented in Python on a personal computer with 16GB of RAM and an Intel i7 processor at 2.9GHz.

Instance	K	N	HYB			HYB-TS			
			Avg. Obj	Best Obj	Pool Size	Avg. Obj	Gap	Best Obj	Pool Size
a2-16	2	16	**306.38**	306.12	1783	307.61	0.40%	306.76	1173
a2-20	2	20	352.33	346.59	5049	353.14	0.23%	325.19	3233
a3-18	3	18	**321.55**	314.51	1227	325.11	1.11%	322.61	1257
a3-24	3	24	**358.26**	355.24	3717	363.12	1.36%	362.34	2721
a4-32	4	32	509.08	503.20	4361	**508.20**	-0.17%	507.54	3501
a4-48	4	48	**741.27**	728.37	5342	743.49	0.30%	731.81	7980
a5-40	5	40	**527.47**	525.16	3038	529.64	0.41%	525.89	3965
a5-50	5	50	738.38	723.05	3553	**734.37**	-0.54%	732.51	10629
a5-60	5	60	1071.81	960.81	2881	**998.69**	-6.82%	950.60	19189
a6-48	6	48	**662.48**	653.24	2730	668.28	0.87%	655.68	4668
a6-72	6	72	1218.70	1209.41	2010	**1077.27**	-11.60%	1069.93	11621
a7-56	7	56	901.19	875.95	350	**826.29**	-8.31%	810.71	3383
a7-84	7	84	1404.93	1401.41	1267	**1216.82**	-13.39%	1203.60	7500
a8-64	8	64	982.39	952.39	104	**868.26**	-11.62%	854.28	5324
a8-80	8	80	1227.50	1226.57	810	**1130.60**	-7.89%	1107.47	6635
a8-96	8	96	1560.78	1557.85	1318	**1494.14**	-4.27%	1480.33	2921

†: Timed out

Gap: (Avg. Obj - Avg. $\text{Obj}_{(HYB)}$)/Avg. $\text{Obj}_{(HYB)}$*100%

Table 5.4: HYB and HYB-TS Performance

For each instance in Table 5.4 and Table 5.5, each algorithm is ran 5 times, and the average and best objective value, the average pool size (number of columns found) at the end, and the average relative gaps (comparing to HYB) over the 5 runs are presented. For B&P based algorithms (B&P-HYB and B&P-HYB-TS), the number of nodes solved is also reported in Table 5.5, indicating how deep down in the branch-and-bound tree has the exploration reached before termination. Since Parragh and Schmid (2013)'s hybrid metaheuristics model is the base model (HYB) in this chapter, the gaps in average objective value between HYB and the other three algorithms are also presented. A negative gap value for an algorithm indicates this algorithm outperforms the algorithm

HYB. The best average objective values across four algorithms are marked in bold for each instance.

Instance	K	N	B&P-HYB					B&P-HYB-TS				
			Avg. Obj	Gap	Best Obj	Pool Size	Nodes	Avg. Obj	Gap	Best Obj	Pool Size	Nodes
a2-16	2	16	309.16	0.91%	306.76	657	1	306.95	0.19%	305.18	556	1
a2-20	2	20	367.02	4.17%	346.59	1127	1	**351.53**	-0.23%	347.99	942	1
a3-18	3	18	324.25	0.84%	322.61	743	1	322.61	0.33%	322.61	875	1
a3-24	3	24	376.40	5.07%	372.43	2059	2	371.11	3.59%	367.15	2210	2
a4-32	4	32	543.08	6.68%	521.02	1036	1	532.61	4.62%	513.22	1022	1
a4-48	4	48	784.01	5.77%	772.86	1182	1	770.36	3.92%	750.31	973	1
a5-40	5	40	540.97	2.56%	540.72	1066	1	535.47	1.52%	529.59	1046	1
a5-50	5	50	797.82	8.05%	762.85	1300	1	858.12	16.22%	858.12	9158	4
a5-60	5	60	1101.88	2.80%	1024.55	957	1	1095.64	2.22%	1005.83	19194	5
a6-48	6	48	697.06	5.22%	665.53	1370	2	699.11	5.53%	687.37	4953	3
a6-72	6	72	1234.19	1.27%	1227.11	1428	2	1198.73	-1.64%	1138.13	2837	2
a7-56	7	56	919.58	2.04%	885.78	738	2	860.72	-4.49%	834.04	731	2
a7-84	7	84	1405.37	0.03%	1402.72	1365	1	1362.55	-3.02%	1274.28	1003	1
a8-64	8	64	979.50	-0.29%	926.88	64	1	984.02	0.17%	955.62	24668	4
a8-80	8	80	1227.50	0.00%	1226.57	780	1	1178.26	-4.01%	1145.52	3945	2
a8-96	8	96	1569.61	0.57%	1550.72	1229	1	1576.36	1.00%	1570.37	1016	1

†: Timed out

Gap: (Avg. Obj - Avg. Obj$_{(HYB)}$)/Avg. Obj$_{(HYB)}$*100%

Table 5.5: B&P-HYB and B&P-HYB-TS Performance

From Table 5.4 and Table 5.5, it is observed that while all algorithms perform at similar efficiency when instance size is small (with HYB showing slight advantage at times), HYB-TS performs predominantly better when instance size is large. This aligns with the motivation behind this chapter, which is to incorporate ML techniques to boost hybrid metaheuristic algorithms' capability to scale up. From the "gap" columns, it can be observed that for small instances, the performance gaps among algorithms are rather insignificant. On the contrary, for larger instances, HYB-TS exhibits evident advantages in performance gaps, outperforming HYB for more than 10% in objective values for some instances. Another observation obtained during implementation is that HYB-TS finds better solution sooner in early RMP iterations. This further validates the potential in abating the effect of dual oscillation in early iterations by integrating TS to Column Generation based DARP algorithms. The results also show that the addition of a branching scheme neither in particular benefits or jeopardise the overall performance.

Instance	K	N	HYB					HYB-TS				
			CPU Time (s)	VNS	RMP	LNS	IP	CPU Time (s)	VNS	RMP	LNS	IP
a2-16	2	16	1176.60	0.58%	0.39%	98.60%	0.43%	433.56	52.93%	10.07%	34.94%	2.06%
a2-20	2	20	1636.24	1.00%	1.75%	93.54%	3.71%	900.26	10.31%	8.88%	78.79%	2.02%
a3-18	3	18	1610.94	0.54%	0.37%	98.63%	0.46%	1036.98	10.57%	2.81%	85.98%	0.64%
a3-24	3	24	2105.83	1.51%	0.98%	95.91%	1.60%	1066.73	11.72%	6.47%	79.97%	1.85%
a4-32	4	32	3298.34	0.32%	0.62%	98.21%	0.85%	1646.57	17.27%	8.88%	71.27%	2.59%
a4-48	4	48	3600†	0.56%	1.11%	90.96%	7.37%	3089.52	10.92%	14.19%	67.32%	7.56%
a5-40	5	40	3600†	0.36%	0.29%	99.01%	0.34%	3480.13	4.84%	3.39%	90.89%	0.88%
a5-50	5	50	3600†	0.46%	0.32%	98.85%	0.37%	3600†	14.50%	8.48%	72.91%	4.10%
a5-60	5	60	3600†	0.85%	0.19%	98.77%	0.20%	3600†	21.04%	9.94%	53.80%	15.22%
a6-48	6	48	3600†	0.58%	0.15%	99.08%	0.18%	3600†	4.86%	3.58%	90.60%	0.96%
a6-72	6	72	3600†	0.94%	0.12%	98.82%	0.13%	3600†	11.52%	6.26%	77.69%	4.53%
a7-56	7	56	3600†	0.20%	0.02%	99.75%	0.03%	3600†	4.18%	2.22%	93.12%	0.48%
a7-84	7	84	3600†	0.10%	0.03%	99.84%	0.03%	3600†	6.84%	2.38%	90.21%	0.57%
a8-64	8	64	3600†	0.14%	0.01%	99.84%	0.02%	3600†	6.97%	2.80%	89.50%	0.73%
a8-80	8	80	3600†	0.13%	0.03%	99.80%	0.04%	3600†	6.59%	2.98%	89.85%	0.58%
a8-96	8	96	3600†	0.24%	0.05%	99.66%	0.05%	3600†	3.19%	1.23%	95.32%	0.26%

†: Timed out

Table 5.6: HYB And HYB-TS Performance - Time Split (%)

Instance	K	N	B&P-HYB				B&P-HYB-TS			
			CPU Time (s)	VNS	RMP	LNS	CPU Time (s)	VNS	RMP	LNS
a2-16	2	16	273.27	0.38%	0.14%	99.48%	196.12	1.60%	0.32%	98.08%
a2-20	2	20	252.39	0.22%	0.10%	99.68%	192.69	5.26%	1.34%	93.40%
a3-18	3	18	651.48	0.35%	0.13%	99.51%	447.53	1.56%	0.25%	98.19%
a3-24	3	24	498.47	0.28%	0.09%	99.62%	285.30	5.24%	8.64%	86.12%
a4-32	4	32	549.49	0.14%	0.05%	99.80%	402.62	0.63%	0.13%	99.24%
a4-48	4	48	765.95	0.15%	0.04%	99.81%	638.58	0.12%	0.04%	99.85%
a5-40	5	40	1252.28	0.12%	0.04%	99.84%	1171.49	0.52%	0.09%	99.39%
a5-50	5	50	1439.12	0.11%	0.04%	99.85%	701.17	6.43%	9.49%	84.08%
a5-60	5	60	1844.46	0.10%	0.02%	99.87%	2251.99	0.94%	0.06%	99.00%
a6-48	6	48	2005.95	0.10%	0.03%	99.87%	1331.70	4.33%	11.84%	83.82%
a6-72	6	72	3285.72	0.57%	0.15%	99.28%	2678.86	0.47%	0.04%	99.50%
a7-56	7	56	3107.81	0.07%	0.02%	99.91%	2295.14	1.92%	0.14%	97.94%
a7-84	7	84	3600†	0.06%	0.02%	99.92%	3079.77	0.30%	0.03%	99.67%
a8-64	8	64	3600†	0.05%	0.01%	99.94%	3613.01	0.42%	0.04%	99.54%
a8-80	8	80	3600†	0.10%	0.02%	99.87%	3339.33	0.35%	0.03%	99.62%
a8-96	8	96	3600†	0.03%	0.01%	99.96%	3408.66	0.21%	0.02%	99.76%

†: Timed out

Table 5.7: B&P-HYB And B&P-HYB-TS Performance - Time Split (%)

In Tables 5.6 and 5.7, the overall CPU time (in seconds) as well as the time split (in percentage) in each component of the algorithm is presented for all four algorithms. Note that by the difference in nature of the algorithms, different termination rules are applied. For HYB and HYB-TS (Algorithm 10), a maximum number of iterations $N^{iterations}$ is defined. For Branch-and-Price structured algorithms (B&P-HYB and B&P-HYB-TS), the process terminates when all nodes are solved or UB and LB converged. Note that the CPU time could be decreased for small instances if an adaptive $N^{iterations}$ is set, which would allow HYB and HYB-TS to terminate earlier when no improvement has been seen over a certain amount of iterations. Since this chapter focuses on the performance of algorithms in larger instances, a same $N^{iterations}$ is set across all instances for consistency ($N^{iterations}$=50 for HYB according to Parragh and Schmid (2013), and $N^{iterations} = 200$ for HYB-TS given the need of learning). It is acknowledged that this may have resulted in longer run time for smaller instances. However, this does not matter much in large instances (where lies the interest of this chapter) since when instance size is too large, algorithms terminate due to time limit. It is observed that the time split across all algorithms is rather similar regarding time distribution, especially for larger instances. The fact that HYB-TS outperforms other algorithms in larger instances with similar time distribution in each algorithmic component also proves the contribution of TS and learning.

In this chapter, the focus is on introducing innovative hybrid solution methods (integrating Thompson Sampling within established heuristics) to solve the classic DARP. Performances of the proposed algorithms are tested on one of the classic benchmark datasets proposed by Cordeau (2006). It is acknowledged that the largest instance in the data set, which consists of 96 requests and 194 nodes, can still be relatively small in real-life settings. In this thesis, Cordeau (2006)'s dataset is used as a first step of benchmark testing, as well as to provide consistency among chapters. The implementation of the proposed algorithms on large-size instances with specific practical constraints will be studied in future research work.

5.6 Conclusion

In this chapter, three hybrid metaheuristic algorithms (HYB-TS, B&P-HYB and B&P-HYB-TS) are proposed and implemented, aiming to scale up the computational capability of DARP solution methods. Thompson Sampling and/or a branching scheme are incorporated with various metaheuristic algorithms including VNS and LNS, providing innovative methodologies to solve large-instance DARPs in a more efficient manner. Thompson Sampling (TS) is applied to model dual values of requests under a column generation setting to negate the effect of dual oscillation. The performance of proposed algorithms is tested using Cordeau (2006)'s benchmark instances, of which the instance size varies from 16 to 96 requests. Numerical results show that the application of Thompson Sampling does have a positive impact on solving large-size DARPs by identifying more promising columns in early iterations thus improving solution quality. More specifically, the integration of Thompson Sampling speeds up the algorithm by 31.56% on average. When running under the same time limit (1 hour) on large instances, algorithms with TS integrated found better solutions overall by 7.25% better in objective values (at best 13.4% better).

Needless to say, incorporating TS to hybrid metaheuristics is only one of many potential applications of Machine Learning to solve DARPs. This chapter is expected to kindle further interest in researchers and open the door of applying ML techniques to efficiently solve rich DARPs in practical operations. For example, besides Reinforcement Learning, Supervised Learning and Unsupervised learning are both promising ML fields that can be applied to metaheuristic solution methods for DARPs in order to mutate the incumbent solution in a more efficient manner. Another potential applications involves RL algorithms is the actor-critic framework, where policy and value function are approximated/evaluated simultaneously in the process. Actor-critic algorithms can be applied to solve DARPs by either end-to-end learning on the entire network, or by working alongside optimisation algorithms to support routing and scheduling decisions. The key to these future applications is to work around rich DARP's problem-specific constraints that are related to service quality and/or specific operational needs when integrating applicable ML techniques.

Chapter 6

Conclusion

Aligning with the three research objectives raised in Chapter 1, this thesis advances the study of rich DARPs from the following aspects:

1. This thesis proposes rich DARP formulations that capture users' preferences from a modelling perspective, with intentions of promoting the quality of DRT services and increasing modal share strategically from a DRT operator's perspectives.

2. The proposed user's preferences DARP framework is applied in both deterministic and stochastic setting to explore relevant pricing strategies and revenue management in operational DRT systems.

3. Customised Local Search algorithms and Math-heuristic solution methods are developed to solve the both standard and rich DARPs, with intentions of solving realistic problems with data extracted from NYC Yellow Taxi datasets. Algorithms that leverage advanced Reinforcement Learning techniques are developed to further scale up the computational efficiency of DARP solution methods.

4. This thesis creates a realistic dataset that reflects real-life travel demand and origin-destination distribution in Manhattan. This dataset is made public online to the OR community, sup-

plementing the existing benchmark (artificial) datasets that are often used to test the performance of DARP solution methods.

This chapter will summarise the work of this thesis, identifying findings and contributions as well as pointing out limitations and potential future research directions.

6.1 Summary

This thesis advances the modelling and solution methods of the Dial-a-Ride Problem from a number of aspects. Aiming to extend DARP and enrich its practical application in operations, various new formulations of the DARP and solution methods are proposed and implemented in this thesis. A chapter by chapter summary will be presented in this section, summarising the contribution of the core chapters of this thesis.

In Chapter 3, two standard Mixed-Integer Linear Programming (MILP) DARP formulations are presented: a link-based DARP model and a path-based DARP model. The link-based formulation uses the flow of each arc (link) of the network as a base unit, presenting constraints at each node and each arc of the system explicitly. On the other hand, the path-based formulation (often acts as the Master Problem in Column Generation related algorithms) deals with complete paths that start from the origin depot and end at the destination depot, leaving the constraints implicit. Different modelling approaches of DARP affect the compatibility of certain solution methods when it comes to solving the problem effectively. Link-based formulations are in particular suitable for Branch-and-Cut approaches, whereas Path-based formulations are more tailored for Column Generation and thus Branch-and-Price based solution methods, where the algorithms start with a small set of known feasible paths, and gradually finding better paths to improve the objective value. Algorithms that are widely used for solving DARP, including Column Generation, Local Search, and some Metaheuristic algorithms, are also presented in this chapter, which act as the backbone of the sophisticated algorithms developed in the following chapters. Mathematical notations and benchmark datasets that are frequently used in DARPs are explained and summarised, for the con-

venience of further references in the following chapters. Furthermore, data cleaning and processing of the datasets used in this thesis and implementations of Classic DARP using commercial solver CPLEX are documented as a benchmark for the rest of this thesis.

In Chapter 4, two extended DARP models (one deterministic, one stochastic) are proposed, capturing users' preferences in a DRT context from a strategic planning aspect. This chapter bridges the research gap in literature where users' preferences and revenue management in DARPs are often overlooked from a DRT operator's perspective. Users' representative utilities for alternative transportation modes are taken into account, including DRT services and driving. This chapter assumes passengers are utility-maximizing and proposes two Mixed-Integer Linear Programming (MILP) formulations for the extended DARP that captures users' preferences using a Logit model. This chapter also explores the design of revenue/fleet management and pricing differentiation: The formulation of the multi-class stochastic Chance-Constrained DARP (CC-DARP) enables the model to explore the behaviours of various user classes under different pricing rules. This provides a new decision-support tool to inform on revenue and fleet management, including fleet sizing, for DRT systems at a strategic planning level. Two customized heuristic solution methods (LS-H and MATH-H) with layered local search structure are developed to solve the stochastic CC-DARP, optimising both routing decision and user selection. Numerical results showed that the customized local search based heuristic algorithm (LS-H) and matheuristic (MATH-H) performed well on DARPs benchmarking instances of the literature when compared to the best integer solution returned by an exact MILP solution. The proposed solution methods are further implemented on a realistic case study derived from yellow taxi trip data from New York City (NYC). The results obtained on the realistic case study reveal that a zonal fare structure is a profit-maximizing strategy for DRT operators.

Chapter 5 tackles the NP-hardness of the DARP and the challenges faced by scaling up solution methods in literature. Following the lead of incorporating Machine Learning (ML) techniques into solving large-scale combinatorial optimisation problems, one of the solution method this chapter proposes (HYB-TS) is an integration of Thompson Sampling and metaheuristics within a Column

Generation algorithmic structure. With a set partitioning DARP formulation, Thompson Sampling (TS) is applied to model dual values of requests under a column generation setting to negate the effect of dual oscillation. Details of how metaheuristics LNS and VNS are modified and integrated to adapt to the machine learning setting are also presented in detail in this chapter. Furthermore, this chapter also explores the effect of Branch-and-bound scheme incorporated into the column generation structured hybrid metaheuristic solution method (B&P-HYB), as an alternative way to obtain integer solutions. Last but not least, Thompson Sampling was also applied to the Branch-and-Price hybrid metaheuristics (B&P-HYB-TS) in implementation to test the effect of TS and Branching, respectively and combined. the performance of proposed algorithms is tested using Cordeau (2006)'s benchmark instances, of which the instance size varies from 16 to 96 requests. Numerical results show that the application of Thompson Sampling does have a positive impact on solving large-size DARPs by identifying more promising columns in early iterations thus improving solution quality.

6.2 Limitations and Future Work

The limitations of this work and potential research directions are outlined next:

- For CC-DARP in Chapter 4, a Logit model is applied to analyse the behaviour of users' preferences between two travel modes (DRT and private trips). It is a starting point for taking users' choice into account from an operator's perspective when modelling DARPs. Future research effort is needed to expand CC-DARP to be multi-modal, considering other alternative travel modes including transit, cycling, etc. Regression models other than Logit can also be considered when it comes to choice modelling in order to model users' preferences.
- For the CC-DARP model, a customised (layered) local search based solution method and a matheuristic solution method are proposed to solve the problem. Although the performance of those two algorithms have been proved to be fairly competitive, more sophisticated algorithms

can be customised and implemented to further advance the computational efficiency of solving CC-DARP with users' preferences. In future research, metaheuristic algorithms such as VNS, ALNS, Tabu Search and HYB-TS (proposed in Chapter 5) can be incorporated into the customised solution methods to solve CC-DARP, which will compensate the weakness of descent local search algorithms (sometimes stuck in local optima).

- In Chapter 5, the proposed solution algorithms can be further modified to be able to solve numerous rich DARPs that are proposed in this work and in literature. The performance of the TS solution methods can be further tested by solving the realistc NYC instance generated in Chapter 4.
- The application of Machine Learning algorithms and techniques in solving combinatorial optimisation problems is a field full of potential. In Chapter 5, Thompson Sampling with a Gaussian prior is integrated into the Column Generation structure, learning the dual values of the MP. This is merely one possible way of integrating Thompson Sampling. Future research may explore the potential of applying TS elsewhere in algorithms and take advantage of learning in more Metaheuristic solution methods, instead of randomly mutating a proportion of the incumbent solution. From a even broader view, the integration of other ML methods, including supervised or unsupervised learning, and more advanced reinforcement learning algorithms, into exact or heuristic approaches for DARPs should be explored.

6.3 Final Remarks

The last few decades witnessed a spike in the public's interest in shared mobility systems and demand-responsive transport services. With the development of ICT and the popularisation of smart devices, DRT services are more accessible to and accepted by the public than ever. This thesis is dedicated to propose novel modelling frameworks and solution methods for the DARP, which ideally will benefit the applications in operational DRT world. Meanwhile, Artificial Intelligence is expanding in every aspect of the modern society Hopefully, this work will also kindle further interest in researchers to further connect the worlds of Machine Learning and combinatory optimisation,

and develop more efficient algorithms to solve the complex routing and scheduling problem faced by public transport and DRT operators. Ultimately, all the effort will bring us closer to more accessible, affordable, reliable and customised public transport.

Bibliography

Agrawal, S., Goyal, N., 2012. Analysis of thompson sampling for the multi-armed bandit problem, in: Conference on learning theory, JMLR Workshop and Conference Proceedings. pp. 39–1.

Agrawal, S., Goyal, N., 2013a. Further optimal regret bounds for thompson sampling, in: Artificial intelligence and statistics, PMLR. pp. 99–107.

Agrawal, S., Goyal, N., 2013b. Thompson sampling for contextual bandits with linear payoffs, in: International Conference on Machine Learning, PMLR. pp. 127–135.

Åkerblom, N., Chen, Y., Chehreghani, M.H., 2020. An online learning framework for energy-efficient navigation of electric vehicles. arXiv preprint arXiv:2003.01416 .

Archetti, C., Feillet, D., Hertz, A., Speranza, M.G., 2009. The capacitated team orienteering and profitable tour problems. Journal of the Operational Research Society 60, 831–842.

Archetti, C., Speranza, M.G., Hertz, A., 2006. A tabu search algorithm for the split delivery vehicle routing problem. Transportation Science 40, 64–73.

Athira, I., Muneera, C., Krishnamurthy, K., Anjaneyulu, M., 2016. Estimation of value of travel time for work trips. Transportation Research Procedia 17, 116–123.

Baugh Jr, J.W., Kakivaya, G.K.R., Stone, J.R., 1998. Intractability of the dial-a-ride problem and a multiobjective solution using simulated annealing. Engineering Optimization 30, 91–123.

Bello, I., Pham, H., Le, Q.V., Norouzi, M., Bengio, S., 2016. Neural combinatorial optimization with reinforcement learning. arXiv preprint arXiv:1611.09940 .

Bengio, Y., Lodi, A., Prouvost, A., 2021. Machine learning for combinatorial optimization: a methodological tour d'horizon. European Journal of Operational Research 290, 405–421.

Bongiovanni, C., Kaspi, M., Cordeau, J.F., Geroliminis, N., 2020. A predictive large neighborhood search for the dynamic electric autonomous dial-a-ride problem. Technical Report.

Calvo, R.W., Touati-Moungla, N., 2011. A matheuristic for the dial-a-ride problem, in: International Conference on Network Optimization, Springer. pp. 450–463.

Cervero, R., 1990. Transit pricing research. Transportation 17, 117–139.

Chao, I.M., Golden, B.L., Wasil, E.A., 1996. The team orienteering problem. European journal of operational research 88, 464–474.

Charnes, A., Cooper, W.W., 1959. Chance-constrained programming. Management Science 6, 73–79.

Chin, A., Lai, A., Chow, J.Y., 2016. Nonadditive public transit fare pricing under congestion with policy lessons from a case study in toronto, ontario, canada. Transportation Research Record 2544, 28–37.

Chow, J.Y., Liu, H., 2012. Generalized profitable tour problems for online activity routing system. Transportation Research Record 2284, 1–9.

Cordeau, J.F., 2006. A branch-and-cut algorithm for the dial-a-ride problem. Operations Research 54, 573–586.

Cordeau, J.F., Laporte, G., 2003a. The dial-a-ride problem (darp): Variants, modeling issues and algorithms. Quarterly Journal of the Belgian, French and Italian Operations Research Societies 1, 89–101.

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

Dell'Amico, M., Maffioli, F., Värbrand, P., 1995. On prize-collecting tours and the asymmetric travelling salesman problem. International Transactions in Operational Research 2, 297–308.

Dell'Olio, L., Ibeas, A., Cecin, P., 2011. The quality of service desired by public transport users. Transport Policy 18, 217–227.

Desrosiers, J., Lübbecke, M.E., 2005. A primer in column generation, in: Column generation. Springer, pp. 1–32.

Di, X., Ma, R., Liu, H.X., Ban, X.J., 2018. A link-node reformulation of ridesharing user equilibrium with network design. Transportation Research Part B: Methodological 112, 230–255.

Diana, M., Dessouky, M.M., 2004. A new regret insertion heuristic for solving large-scale dial-a-ride problems with time windows. Transportation Research Part B: Methodological 38, 539–557.

Diana, M., Dessouky, M.M., Xia, N., 2006. A model for the fleet sizing of demand responsive transportation services with time windows. Transportation Research Part B: Methodological 40, 651–666.

Dong, X., Chow, J.Y., Waller, S.T., Rey, D., 2022. A chance-constrained dial-a-ride problem with utility-maximising demand and multiple pricing structures. Transportation Research Part E: Logistics and Transportation Review 158, 102601.

Dong, X., Rey, D., Waller, S.T., 2020. Dial-a-ride problem with users' accept/reject decisions based on service utilities. Transportation Research Record , 0361198120940307.

Dumas, Y., Desrosiers, J., Soumis, F., 1991. The pickup and delivery problem with time windows. European journal of Operational Research 54, 7–22.

Falcocchio, J., 1979. A Methodology for Evaluating the Effectiveness of Transportation Improvements for the Elderly and Handicapped. Technical Report.

Feillet, D., Dejax, P., Gendreau, M., 2005a. The profitable arc tour problem: solution with a branch-and-price algorithm. Transportation Science 39, 539–552.

Feillet, D., Dejax, P., Gendreau, M., 2005b. Traveling salesman problems with profits. Transportation Science 39, 188–205.

Ferreira, K.J., Simchi-Levi, D., Wang, H., 2018. Online network revenue management using thompson sampling. Operations Research 66, 1586–1602.

Fortet, R., 1960. Applications de l'algebre de boole en recherche opérationelle. Revue Française de Recherche Opérationelle 4, 17–26.

Garaix, T., Artigues, C., Feillet, D., Josselin, D., 2011. Optimization of occupancy rate in dial-a-ride problems via linear fractional column generation. Computers & Operations Research 38, 1435–1442.

Gentile, G., Papola, N., Persia, L., 2005. Advanced pricing and rationing policies for large scale multimodal networks. Transportation Research Part A: Policy and Practice 39, 612–631.

Gunay, B., Akgol, K., Andreasson, I., Terzi, S., 2016. The estimation of modal shift potential for a new form of dial-a-ride service. Journal of Public Transportation 19, 5.

He, B.Y., Zhou, J., Ma, Z., Chow, J.Y., Ozbay, K., 2020. Evaluation of city-scale built environment policies in new york city with an emerging-mobility-accessible synthetic population. Transportation Research Part A: Policy and Practice 141, 444–467.

Ho, S.C., Haugland, D., 2011. Local search heuristics for the probabilistic dial-a-ride problem. Or Spectrum 33, 961–988.

Ho, S.C., Szeto, W.Y., Kuo, Y.H., Leung, J.M., Petering, M., Tou, T.W., 2018. A survey of dial-a-ride problems: Literature review and recent developments. Transportation Research Part B: Methodological 111, 395–421.

Iommazzo, G., D'Ambrosio, C., Frangioni, A., Liberti, L., 2020. A learning-based mathematical programming formulation for the automatic configuration of optimization solvers, in: International Conference on Machine Learning, Optimization, and Data Science, Springer. pp. 700–712.

Jaw, J.J., 1984. Solving large-scale dial-a-ride vehicle routing and scheduling problems. Ph.D. thesis. Massachusetts Institute of Technology.

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

Jepsen, M.K., Petersen, B., Spoorendonk, S., Pisinger, D., 2014. A branch-and-cut algorithm for the capacitated profitable tour problem. Discrete Optimization 14, 78–96.

Jorgensen, R.M., Larsen, J., Bergvinsdottir, K.B., 2007. Solving the dial-a-ride problem using genetic algorithms. Journal of the Operational Research Society 58, 1321–1331.

Kirchler, D., Calvo, R.W., 2013. A granular tabu search algorithm for the dial-a-ride problem. Transportation Research Part B: Methodological 56, 120–135.

Lisco, T.E., 1968. Value of commuters travel time-a study in urban transportation. Technical Report.

Litman, T., 2009. Transportation cost and benefit analysis ii–travel time costs. Victoria Transport Policy Institute, Victoria, Canada .

Liu, Y., Wang, S., Xie, B., 2019. Evaluating the effects of public transport fare policy change together with built and non-built environment features on ridership: The case in south east queensland, australia. Transport Policy 76, 78–89.

Lodi, A., Zarpellon, G., 2017. On learning and branching: a survey. Top 25, 207–236.

Masmoudi, M.A., Hosny, M., Braekers, K., Dammak, A., 2016. Three effective metaheuristics to solve the multi-depot multi-trip heterogeneous dial-a-ride problem. Transportation Research Part E: Logistics and Transportation Review 96, 60–80.

Masmoudi, M.A., Hosny, M., Demir, E., Genikomsakis, K.N., Cheikhrouhou, N., 2018. The dial-a-ride problem with electric vehicles and battery swapping stations. Transportation Research Part E: Logistics and Transportation Review 118, 392–420.

Mauri, G., Lorena, L.N., 2006. A multiobjective model and simulated annealing approach for a dial-a-ride problem, in: Workshop dos cursos de computaçao.

Migdalas, A., 1995. Bilevel programming in traffic planning: models, methods and challenge. Journal of Global Optimization 7, 381–405.

Mohring, H., Schroeter, J., Wiboonchutikula, P., 1987. The values of waiting time, travel time, and a seat on a bus. The RAND Journal of Economics , 40–56.

Molenbruch, Y., Braekers, K., Caris, A., 2017. Typology and literature review for dial-a-ride problems. Annals of Operations Research 259, 295–325.

Montoya, A., Guéret, C., Mendoza, J.E., Villegas, J.G., 2016. A multi-space sampling heuristic for the green vehicle routing problem. Transportation Research Part C: Emerging Technologies 70, 113–128.

Muter, İ., Birbil, Ş.İ., Şahin, G., 2010. Combination of metaheuristic and exact algorithms for solving set covering-type optimization problems. INFORMS Journal on Computing 22, 603–619.

Nair, R., Miller-Hooks, E., 2014. Equilibrium network design of shared-vehicle systems. European Journal of Operational Research 235, 47–61.

Nazari, M., Oroojlooy, A., Snyder, L., Takác, M., 2018. Reinforcement learning for solving the vehicle routing problem. Advances in neural information processing systems 31.

Parragh, S.N., 2011. Introducing heterogeneous users and vehicles into models and algorithms for the dial-a-ride problem. Transportation Research Part C: Emerging Technologies 19, 912–930.

Parragh, S.N., Cordeau, J.F., Doerner, K.F., Hartl, R.F., 2012. Models and algorithms for the heterogeneous dial-a-ride problem with driver-related constraints. OR spectrum 34, 593–633.

Parragh, S.N., Doerner, K.F., Hartl, R.F., 2010. Variable neighborhood search for the dial-a-ride problem. Computers & Operations Research 37, 1129–1138.

Parragh, S.N., Doerner, K.F., Hartl, R.F., Gandibleux, X., 2009. A heuristic two-phase solution approach for the multi-objective dial-a-ride problem. Networks: An International Journal 54, 227–242.

Parragh, S.N., Schmid, V., 2013. Hybrid column generation and large neighborhood search for the dial-a-ride problem. Computers & Operations Research 40, 490–497.

Parragh, S.N., Pinho de Sousa, J., Almada-Lobo, B., 2014. The dial-a-ride problem with split requests and profits. Transportation Science 49, 311–334.

Parragh, S.N., Pinho de Sousa, J., Almada-Lobo, B., 2015. The dial-a-ride problem with split requests and profits. Transportation Science 49, 311–334.

Paulley, N., Balcombe, R., Mackett, R., Titheridge, H., Preston, J., Wardman, M., Shires, J., White, P., 2006. The demand for public transport: The effects of fares, quality of service, income and car ownership. Transport policy 13, 295–306.

Prescott-Gagnon, E., Desaulniers, G., Rousseau, L.M., 2009. A branch-and-price-based large neighborhood search algorithm for the vehicle routing problem with time windows. Networks: An International Journal 54, 190–204.

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

Redman, L., Friman, M., Gärling, T., Hartig, T., 2013. Quality attributes of public transport that attract car users: A research review. Transport policy 25, 119–127.

Rey, D., Dixit, V.V., Ygnace, J.L., Waller, S.T., 2016. An endogenous lottery-based incentive mechanism to promote off-peak usage in congested transit systems. Transport Policy 46, 46–55.

Ropke, S., Cordeau, J.F., 2009. Branch and cut and price for the pickup and delivery problem with time windows. Transportation Science 43, 267–286.

Ropke, S., Cordeau, J.F., Laporte, G., 2007. Models and branch-and-cut algorithms for pickup and delivery problems with time windows. Networks: An International Journal 49, 258–272.

Ropke, S., Pisinger, D., 2006. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. Transportation science 40, 455–472.

Russo, D., Van Roy, B., 2014. Learning to optimize via posterior sampling. Mathematics of Operations Research 39, 1221–1243.

Russo, D.J., Van Roy, B., Kazerouni, A., Osband, I., Wen, Z., et al., 2018. A tutorial on thompson sampling. Foundations and Trends® in Machine Learning 11, 1–96.

Sayarshad, H.R., Chow, J.Y., 2015. A scalable non-myopic dynamic dial-a-ride and pricing problem. Transportation Research Part B: Methodological 81, 539–554.

Schönberger, J., Kopfer, H., Mattfeld, D.C., 2003. A combined approach to solve the pickup and delivery selection problem, in: Operations Research Proceedings 2002. Springer, pp. 150–155.

Shaw, P., 1998. Using constraint programming and local search methods to solve vehicle routing problems, in: International conference on principles and practice of constraint programming, Springer. pp. 417–431.

Shi, J., Gao, Y., Wang, W., Yu, N., Ioannou, P.A., 2019. Operating electric vehicle fleet for ride-hailing services with reinforcement learning. IEEE Transactions on Intelligent Transportation Systems 21, 4822–4834.

Small, K.A., 1982. The scheduling of consumer activities: work trips. The American Economic Review 72, 467–479.

Streeting, M., Charles, P., 2006. Developments in transit fare policy reform, The Australasian Transport Research Forum.

Talluri, K., Van Ryzin, G., 2004. Revenue management under a general discrete choice model of consumer behavior. Management Science 50, 15–33.

Tang, H., Miller-Hooks, E., 2005. Algorithms for a stochastic selective travelling salesperson problem. Journal of the Operational Research Society 56, 439–452.

Thompson, W.R., 1933. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. Biometrika 25, 285–294.

Tirachini, A., Hensher, D.A., 2012. Multimodal transport pricing: first best, second best and extensions to non-motorized transport. Transport Reviews 32, 181–202.

Tirachini, A., Hensher, D.A., Rose, J.M., 2014. Multimodal pricing and optimal design of urban public transport: The interplay between traffic congestion and bus crowding. Transportation Research Part B: Methodological 61, 33–54.

Toth, P., Vigo, D., 1997. An exact algorithm for the vehicle routing problem with backhauls. Transportation Science 31, 372–385.

Train, K.E., 2009. Discrete choice methods with simulation. Cambridge university press.

Vinyals, O., Fortunato, M., Jaitly, N., 2015. Pointer networks. arXiv preprint arXiv:1506.03134 .

Xiang, Z., Chu, C., Chen, H., 2006. A fast heuristic for solving a large-scale static dial-a-ride problem under complex constraints. European journal of Operational Research 174, 1117–1139.

Xu, H., Pang, J.S., Ordónez, F., Dessouky, M., 2015. Complementarity models for traffic equilibrium with ridesharing. Transportation Research Part B: Methodological 81, 161–182.

Yoon, G., Chow, J.Y., Dmitriyeva, A., Fay, D., 2020. Effect of routing constraints on learning efficiency of destination recommender systems in mobility-on-demand services. IEEE Transactions on Intelligent Transportation Systems .

Zhao, J., Mao, M., Zhao, X., Zou, J., 2020. A hybrid of deep reinforcement learning and local search for the vehicle routing problems. IEEE Transactions on Intelligent Transportation Systems 22, 7208–7218.