Techniques for large scale deployment of demand-aware bus transit systems

Perera, Talagalage Thilina Dharshana

2020

Perera, T. T. D. (2020). Techniques for large scale deployment of demand-aware bus transit systems. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/144682>

<https://doi.org/10.32657/10356/144682>

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License (CC BY-NC 4.0).

Downloaded on 26 Apr 2023 22:02:27 SGT

TECHNIQUES FOR LARGE SCALE DEPLOYMENT OF DEMAND-AWARE BUS TRANSIT SYSTEMS

TALAGALAGE THILINA DHARSHANA PERERA

School of Computer Science and Engineering

*A thesis submitted to the Nanyang Technological University
in partial fulfilment of the requirement for the degree of
Doctor of Philosophy*

2020

"When the going got tough, she ensured that I reached the end"

This thesis is dedicated to my darling wife, Deshya

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

20.06.2020

.....
Date

.....
Talagalage Thilina Dharshana Perera

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

26/6/2020

.....
Date .....
Dr Thambipillai Srikanthan (Prof.)

Authorship Attribution Statement

This thesis contains material from 7 papers published in the following peer-reviewed journals/conferences in which I am listed as an author.

Chapter 3 is published as T. Perera, A. Prakash and T. Srikanthan, "A Scalable Heuristic Algorithm for Demand Responsive Transportation for First Mile Transit," International Conference on Intelligent Engineering Systems (INES), Larnaca, 2017, pp. 157-162. DOI: <https://doi.org/10.1109/INES.2017.8118547>

The contributions of the co-authors are as follows:

- Prof. Srikanthan provided the initial project direction.
- Dr. Prakash participated in idea generation, brainstorming sessions and revised the manuscript.
- I designed the study, performed all experiments, analyzed the data and prepared the manuscript.

Chapter 3 is published as T. Perera, A. Prakash and T. Srikanthan, "A Hybrid Methodology for Optimal Fleet Management in an Electric Vehicle based Flexible Bus Service," International Conference on Control, Automation, Robotics and Vision (ICARCV), Singapore, 2018, pp. 331-336. DOI: <https://doi.org/10.1109/ICARCV.2018.8581345>

The contributions of the co-authors are as follows:

- Prof. Srikanthan provided the initial project direction.
- Dr. Prakash participated in idea generation, brainstorming sessions and revised the manuscript.
- I designed the study, performed all experiments, analyzed the data and prepared the manuscript.

Chapter 4 is published as T. Perera, A. Prakash, C. N. Gamage and T. Srikanthan, “Hybrid Genetic Algorithm for an On-Demand First Mile Transit System using Electric Vehicles,” In: Shi Y. et al. (eds) Computational Science. ICCS 2018. Lecture Notes in Computer Science, vol 10860. Springer, Cham. DOI: https://doi.org/10.1007/978-3-319-93698-7_8

The contributions of the co-authors are as follows:

- Prof. Srikanthan provided the initial project direction.
- Dr. Prakash participated in idea generation and brainstorming sessions
- I prepared the manuscript drafts. The manuscript was revised by Dr. Prakash.
- I designed the study, performed all experiments and analyzed the data.
- Mr. Gamage assisted in the experiments.

Chapter 4 is published as T. Perera, A. Prakash and T. Srikanthan, “Genetic Algorithm based EV Scheduling for On-Demand Public Transit System,” In: Rodrigues J. et al. (eds) Computational Science. ICCS 2019. Lecture Notes in Computer Science, vol 11540. Springer, Cham. DOI: https://doi.org/10.1007/978-3-030-22750-0_56

The contributions of the co-authors are as follows:

- Prof. Srikanthan provided the initial project direction.
- Dr. Prakash participated in idea generation, brainstorming sessions and revised the manuscript.
- I designed the study, performed all experiments, analyzed the data and prepared the manuscript.

Chapter 5 is published as T. Perera, A. Prakash, D. Wijesundera, T. Srikanthan and C. N. Gamage, "Cluster-First, Route-Second Heuristic for EV Scheduling in On-Demand Public Transit," International Conference on Cybernetics and Intelligent Systems (CIS) and Conference on Robotics, Automation and Mechatronics (RAM), Bangkok, Thailand, 2019, pp. 176-181, DOI: <https://doi.org/10.1109/CIS-RAM47153.2019.9095840>

The contributions of the co-authors are as follows:

- Prof. Srikanthan provided the initial project direction
- Dr. Prakash participated in idea generation and brainstorming sessions
- Dr. Wijesundera participated in idea generation and brainstorming sessions
- I prepared the manuscript drafts. The manuscript was revised by Dr. Wijesundera and Dr. Prakash
- I designed the study performed all experiments and analyzed the data.
- Mr. Gamage assisted in the experiments.

Chapter 5 is published as T. Perera, D. Wijesundera, L. Wijerathna and T. Srikanthan, "Directionality-centric Bus Transit Network Segmentation for On-Demand Public Transit," IET Intelligent Transport Systems Journal (To Appear)

The contributions of the co-authors are as follows:

- Prof. Srikanthan provided the initial project direction
- Dr. Wijesundera participated in idea generation, brainstorming sessions and revised the manuscript.
- I designed the study, performed all experiments, analyzed the data and prepared the manuscript drafts.
- Mr. Wijerathna assisted in the experiments.

Chapter 7 is published as T. Perera, C. N. Gamage, A. Prakash and T. Srikanthan, "A Simulation Framework for a Real-Time Demand Responsive Public Transit System," International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, 2018, pp. 608-613. DOI: <https://doi.org/10.1109/ITSC.2018.8569281>

The contributions of the co-authors are as follows:

- Prof. Srikanthan provided the initial project direction.
- Dr. Prakash participated in idea generation, brainstorming sessions and revised the manuscript.
- Mr. Gamage assisted in the experiments and analyzing the data.
- I designed the study, performed all experiments, analyzed the data and prepared the manuscript.

08.11.2020

.....
Date

.....
Talagalage Thilina Dharshana Perera

Acknowledgements

I would like to thank all the people who have directly and indirectly helped me find my footing in research. First and foremost, I would like to express my heartfelt gratitude to my supervisor and mentor, Prof. Thambipillai Srikanthan, for his support, time, valuable guidance and constant encouragement. His constructive suggestions and ideas have helped me immensely to uncover the contributions presented in this thesis.

My gratitude extends to my colleague and wife Dr. Deshya Wijesundera who has played a vital role at every stage of this thesis. The discussions with her has greatly contributed to this research. More importantly, it was her support and encouragement which made this thesis a reality.

I am very much grateful to Dr Alok Prakash for helping me throughout this work. His support, insights and the discussions have helped me at all stages of the thesis. My good friend and colleague Kalindu helped me immensely throughout this journey. His encouragement, insight on research and the numerous other ways he helped me in Singapore was invaluable.

The technical discussions with Dr. George Jagadeesh have been extremely helpful. My sincere appreciation goes to Chathura and Lahiru who worked with me as interns in HESL. My gratitude extends to Jeremiah for his invaluable administrative support.

I would also like to thank my parents for all their efforts. Last, but not the least, I would like to thank my mother-in-law, who is actually a mother to me. Thank you for all the support and encouragement which saw me through the toughest of times.

Table of contents

List of figures	xix
List of tables	xxv
Abstract	xxix
1 Introduction	1
1.1 Motivation	1
1.2 Research Objective	3
1.3 Organization	4
1.4 Publications	6
2 Literature Review	9
2.1 Introduction	9
2.2 Public Transit Network	10
2.2.1 Connectivity Gap in Public Transit Network	11
2.2.2 Strategies for Bridging Public Transit Connectivity Gap	12
2.3 Demand-aware Transit Systems	18

2.3.1	Vehicle Routing Problem	19
2.3.2	Categories of Demand-aware Transit Systems	23
2.3.3	Simulators	33
2.3.4	Commercial Efforts	34
2.3.5	Scalability Issues	34
2.4	Demand-aware Bus Transit Systems	37
2.5	Summary	41
3	Modelling Demand-aware Bus Transit System	45
3.1	Introduction	45
3.2	Characteristics of Bus Transit System	46
3.3	Design Considerations	51
3.4	Model Generation	60
3.4.1	Mathematical Formulation	66
3.5	Performance Evaluation	72
3.5.1	Experimental Setup	72
3.5.2	Experiments	73
3.6	Summary	81
4	Accelerating Constraint-aware Flexible Route Generation	83
4.1	Introduction	83
4.2	Methodology	84
4.2.1	Dual-Objective Genetic Algorithm	86

4.3	Performance Evaluation	99
4.3.1	Experimental Setup	99
4.3.2	Experiments	100
4.4	Comparing with Existing Bus Transit System	108
4.5	Summary	116
5	Directionality-centric Segmentation of Bus Transit Network	119
5.1	Introduction	119
5.2	Methodology	120
5.2.1	Directionality & Shortest-path Centric Generation of Sub-Zones	121
5.2.2	Allocating EVs to Sub-Zones	135
5.2.3	Flexible Routing for Sub-Zones	138
5.3	Performance Evaluation	139
5.3.1	Experimental Setup	140
5.3.2	Experiments	141
5.4	Summary	150
6	Adopting Demand-aware Bus Transit System for Large Scale Deployment	153
6.1	Introduction	153
6.2	Methodology	154
6.2.1	Classification of Bus Transit System	158
6.2.2	Identifying Transit Hub Region using Feeder Bus Services . .	161

6.2.3	Segmenting Transit Hub Region into Zones	164
6.2.4	Generating Sub-Zones	172
6.2.5	Citywide Adoption	173
6.3	Performance Evaluation	174
6.3.1	Experimental Setup	175
6.3.2	Experiments	176
6.4	Demand-aware Scheduling of Point-to-Point Connections	181
6.5	Summary	185
7	Citywide Deployment of Demand-aware Bus Transit System	187
7.1	Introduction	187
7.2	Real-Time Demand Generation & Estimation	188
7.2.1	Mobile Phone-based Sensors	189
7.2.2	Camera-based Crowd Estimation	191
7.2.3	Public Transit Tap-In & Tap-Out Data	193
7.3	Near-Future Demand Estimation	194
7.4	Proposed Framework for Citywide Deployment	197
7.4.1	Demand Estimation Process	197
7.4.2	Demand-aware Online Scheduling & Routing Process	198
7.4.3	Demand-aware Offline Scheduling Process	201
7.4.4	Proposed Framework	202
7.4.5	Computing Requirements of Sub-Zones	206

7.5	Summary	207
8	Conclusions and Future Directions	209
8.1	Conclusions	209
8.2	Future Research Directions	213
	References	217

List of figures

3.1	Transit Hubs of the Public Transit Network	47
3.2	Densely Distributed Spokes of the Public Transit Network	48
3.3	Illustration of a Typical Public Transit Journey	49
3.4	Travel Times for a Typical Public Transit Journey	50
3.5	Snapshot of the Proposed Demand-aware Bus Transit System	52
3.6	Example of Published Bus Frequencies	53
3.7	Graph Representation of Demand-aware Flexible Routing	62
3.8	Impact of Vehicle Capacity on Passenger Travel Time Optimization .	74
3.9	Waiting & Riding Time Variation with Changing Capacity (for Passenger Travel Time Optimization)	75
3.10	Impact of Vehicle Capacity on VMT Optimization	75
3.11	Waiting & Riding Time Variation with Changing Capacity (for VMT Optimization)	76
3.12	Impact of EV Driving Range	77
3.13	Pareto Front for Simultaneous Optimization of Passenger Travel Time and VMT	80
3.14	Runtime Variation	81

4.1	Proposed Methodology of the Dual-Objective GA for Simultaneous Optimization of Passenger Travel Time and VMT	87
4.2	Example Solution Encoding $V_1 : 3, 8$; $V_2 : 2, 7, 4, 6, 1$; $V_3 : 9, 5, 10$. .	89
4.3	Mutation Operators	92
4.4	Crossover Operators	93
4.5	Illustration of Selecting Parents	95
4.6	Block Diagram of the GA for Generating Nadir Points	98
4.7	Impact of the Local Search Technique	103
4.8	Comparison of the Pareto Fronts	105
4.9	Runtime Comparison with NSGA-II Algorithm	107
4.10	Impact of Vehicle Capacity on Average Waiting Time and VMT . . .	114
5.1	Proposed Methodology for Generating Flexible Routes for a Zone . .	120
5.2	Technique for Generation of Sub-Zones	121
5.3	Shortest Path Route Obtained from OSRM	123
5.4	Route Segments Representing Polylines in Table 5.1	124
5.5	Shortest Path Routes from Bus Stops B1, B2 and B3 to the Destination	125
5.6	Polyline Tree for Example in Figure 5.5	125
5.7	Example of Elementary Clusters	126
5.8	Example Network for Algorithm 3	130
5.9	Technique for Allocating EVs to Sub-Zones	135
5.10	Technique for Flexible Routing for Sub-Zones	138
5.11	Selected Zone for the Experiments	140

5.12 Comparison of VMT of the Fleet with K-Means and K-Medoids Algorithms	142
5.13 Comparison of the Pareto Fronts	145
5.14 Runtime Comparison with the Technique Without Sub-Zoning	146
5.15 Runtime Variation of Flexible Routing for Sub-Zones Without Work-load Balancing	147
5.16 Example of Passenger Demand at Bus Stops	149
6.1 Example City	156
6.2 Locations of Transit Hubs	156
6.3 Distinct Transit Hub Regions with Minimal Overlaps	157
6.4 Segmented Zones in a Transit Hub Region	157
6.5 Example of Bus Services	158
6.6 Feeder Bus Services of Transit Hubs	160
6.7 Example of Transit Hub Regions with the Corresponding Feeder Bus Services	161
6.8 Example for Identifying Transit Hub Region Boundary	162
6.9 Overlapped Boundaries of Adjacent Transit Hub Regions	163
6.10 Selected Bus Stops and the Shortest Path Routes to the Transit Hub . .	165
6.11 Generated Tree of Example in Figure 6.10	166
6.12 Example Generated Tree with 200 Passengers	167
6.13 Methodology for Citywide Adoption	173
6.14 Passenger Distribution in Zones	178

6.15	Trunk Bus Service Routes	182
6.16	Different Scheduling Frequencies of a Trunk Bus Service	183
6.17	Proposed Methodology for Integrating Demand-aware Scheduling of Point-to-Point Connections	184
7.1	Example of BLE-beacon based Real-Time Demand Generation at a Bus Stop. Passengers that have Installed the Mobile Application and Enabled Bluetooth Communication will Connect to the 3 Placed BLE-Beacons to Share Anonymised Data. This Enables to Generate Real-Time Demand at the Bus Stop	190
7.2	Example of using Surveillance Camera for Real-Time Demand Gen- eration. The Surveillance Camera is Mounted at a Pole Opposite the Bus Stop. It can be used Alongside a Computer Vision Software to Count Real-Time Passenger Demand	192
7.3	Output of Computer Vision Software for Counting Passengers. Iden- tified Passengers in the Bus Stop has been Marked using The Green Dots	192
7.4	Proposed Technique for Near-Future Demand Estimation. Technique uses Demand Data from Mobile Phone-based Sensors, Camera-based Crowd Estimation and Fare Card Tap-In & Tap-Out Data to Estimate the Near-Future Demand	195
7.5	Technique for Realizing the Near-Future Demand Estimation Process	198
7.6	Technique for Realizing the Demand-aware Online Scheduling & Routing Process. The Process is Divided into Offline and Online Modules that Expedites Real-Time Flexible Route Generation	199

7.7	Technique for Realizing the Demand-aware Offline Scheduling Process. The Process Generates Demand-aware Offline Schedules for Trunk Bus Services	202
7.8	Proposed Framework for Citywide Deployment	203
7.9	Runtime of Generating Flexible Routes for a Zone	206

List of tables

3.1	Terminology of Passengers and EV	61
3.2	Terminology of Nodes of the Graph	62
3.3	Terminology of Edges of the Graph	63
3.4	Constraints of the Model	63
4.1	Terminology	88
4.2	Values of the Parameters	100
4.3	Test Cases	100
4.4	Results for User Experience Optimization	101
4.5	Results for Operator Profitability Optimization	102
4.6	Auxiliary Data of Feeder Bus Service 243G	109
4.7	Travel Time and Distance Between Bus Stops of Bus Service 243G . .	109
4.8	Comparison of Vehicles	111
4.9	Passenger Distribution at Bus Stops	113
4.10	Comparison of Results	115
5.1	Encoded Polyline Sequence Obtained from OSRM	123

5.2	Variables used in Algorithm 3	127
5.3	Objects and Functions of Algorithm 3	128
5.4	Identified Elementary Clusters of the Example in Figure 5.8	133
5.5	Supplementary Functions for Algorithm 4	134
5.6	Values of the Parameters	141
5.7	Test Cases	141
6.1	Definition of Regions	155
6.2	Values of the Parameters	175
6.3	Test Cases	175
6.4	Analysis of the Passenger Spread in the Zones	179
6.5	Comparison of VMT	179
6.6	Comparison of Runtime	180

List of Abbreviations

AFV	Alternative Fuel Vehicle
BFS	Breadth First Search
BLE	Bluetooth Low Energy
DVRP	Dynamic Vehicle Routing Problem
ETA	Estimated Time of Arrival
EV	Electric Vehicle
E-VRP	Electric-Vehicle Routing Problem
GA	Genetic Algorithm
HOV	High Occupancy Vehicle
LSTM	Long Short Term Memory
LTA	Land Transport Authority
MIQCP	Mixed Integer Quadratically Constrained Program
NTU	Nanyang Technological University
O-D	Origin-Destination
OPL	Optimization Programming Language
OSRM	Open Source Routing Machine
PAM	Partitioning Around Medoids
PMD	Personal Mobility Device
SOV	Single Occupancy Vehicle
SVRP	Stochastic Vehicle Routing Problem

TOD Transit Oriented Development

TSP Travelling Salesman Problem

VMT Vehicle Miles Travelled

VRP Vehicle Routing Problem

VRPTW Vehicle Routing Problem with Time Window

Abstract

Public transit systems have been constantly plagued by the inherent connectivity gap due to fixed routes and schedules of feeder bus services. Demand-aware bus transit systems that rely on real-time scheduling of flexible routes have gained popularity as an alternative to bridge the connectivity gap, thereby enhancing user experience and operator profitability. In this research, scalable techniques have been proposed to realize citywide deployment of a demand-aware bus transit system to replace the conventional fixed-route based feeder bus services.

In Chapter 3, a graph-based representation has been proposed to model demand-aware flexible route generation. The mixed integer programming model for generating the optimal flexible routes incorporates real-life scenarios including actual distances of the road network and asymmetric distance/time matrices that represent the different ‘to and fro’ distance/travelling times between two given points. The proposed model can successfully generate optimal flexible routes to enhance the travel times of passengers (user experience) or vehicle miles travelled by the fleet (operator profitability). In particular, the normalized weighted technique has been introduced to facilitate trade-off analysis based on user requirements to ensure that the flexible routes are sensitive to both the user experience and operator profitability. The proposed model has been successfully employed to prune the design space to speed up route computations without compromising optimality. Experimental results demonstrate the capability of the model in performing diverse what-if analyses by varying different input parameters.

A heuristic routing technique has been proposed in Chapter 4 to accelerate the flexible route generation process by combining both the ϵ -constraint method and a genetic algorithm. The technique incorporates nearest neighbour heuristic to generate superior initial solutions, selection of genetic operators for fast convergence, and a hybrid parent selection algorithm for balancing solution quality and diversity. Experimental results confirm that routes generated by the proposed technique deviate only 3% from the optimal values. The rapid convergence of the proposed technique results in a 26% reduction in runtime when compared to a widely-used baseline algorithm.

A directionality-centric technique has been proposed for the systematic segmentation of bus transit network in Chapter 5. The network segmentation technique generates sub-zones based on the feasible shortest path routes from its bus stops to the destination. Heuristic technique proposed in Chapter 4 has been employed to generate the flexible routes of the size limited sub-zones. This has led to notable speed-up of flexible route computations that can also benefit from parallel computations, thereby paving the way

for a highly scalable technique without compromising the responsiveness demanded by demand-aware bus transit systems. The outlier bus stops of sub-zones are incorporated into the neighbouring sub-zones on-the-fly to minimize vehicle detours. Moreover, dynamic methods for demand-aware allocation of vehicles and workload balancing among sub-zones improves the overall responsiveness of a large-scale deployment. Experimental results confirm that the routes generated using the proposed technique achieve over 7x speed-up when compared to a global, heuristic routing technique without compromising on solution quality. A similar performance improvement was also evident for the case of sporadic demands, highlighting the applicability of the proposed network segmentation technique to real-life scenarios.

In Chapter 6, applicability of the proposed methods to a large scale deployment of demand-aware bus transit system has been demonstrated. This necessitated the systematic segmentation of feeder bus services, into sub-zones and outlier bus stops as well as point-to-point trunk services. Identification of independent transit hub regions of a large scale transit system, as well as segmenting each transit hub into workload balanced zones has notably improved the responsiveness of route computations. Experimental results confirm that, compared to a widely-used unsupervised learning algorithm, the zone-wise runtime has improved by 83% while also improving the quality of routes. A demand-aware scheduling technique to improve the user experience and operator profitability of trunk bus services has also been proposed in this chapter.

In Chapter 7, the various techniques proposed in this thesis have been integrated into a framework to realize the citywide deployment of demand-aware flexible routing. The model for demand prediction was trained using multi-modal sensing inputs from mobile apps and vision-based crowd counting. This has paved the way for estimating near-future demand at each bus stop to schedule flexible routes in real-time. Runtime performance of citywide route generation process has also been improved by the offline processing of a significant component of the workload and by limiting the generation of sub-zones, based on near-future demand estimation, to online. Experiments confirm that the flexible routes can be computed in 20 and 24 seconds, for workloads of 50 and 65 passengers respectively, when implemented on a 4-core Xeon E5-1630 V4 CPU running at 3.70 GHz. The proposed sensing methods also lend well for the periodic generation of offline schedules for trunk bus services.

Finally, real-life deployment of the proposed techniques for the city-wide replacement of fixed-route based bus transit systems has been successfully demonstrated to minimise the connectivity gap inherent in current implementations.

Chapter 1

Introduction

1.1 Motivation

The public transit network consisting of rail and road transit systems is constantly challenged to meet the diversified needs of passengers. These systems are not only expected to provide high quality of service (in terms of reliability and efficiency) but also to ensure a high degree of penetration. Furthermore, higher ridership in the public transit network can also lead to easing traffic congestion and pollution in addition to economic gains. As highlighted by the study of Knupfer et al., in 24 global cities, public transit network is almost twice as efficient (64%) as the private transit network (36%), where efficiency is defined by how quickly and reliably a trip can be done [1]. As a result, in Singapore, public transit ridership has grown nearly 50% in the last decade [2] [3]. However, the public transit network has been constantly plagued by the connectivity gap created by the fixed routes and fixed schedules of feeder buses transporting passengers from bus stops to transit hubs and vice versa. For example, in a significant number of cases, passengers must tolerate high waiting times at bus stops due to the fixed schedules. At the same time, the fixed route is often longer than the direct route to the destination due to detours, which increases the riding time as well

as travel distance of the buses. Thus, demand-aware transit systems have emerged as a viable alternative to bridge the connectivity gap.

Demand-aware transit systems react to the ride requests of passengers in real-time by dispatching vehicles based on the demand unlike the fixed schedule counterparts. Therefore, waiting time of passengers at the bus stops can be reduced. In addition, vehicles can follow flexible routes to pick-up and drop-off passengers, which minimizes the riding time. Traditionally, demand-aware transit systems have been operating only in areas, where ride requests are updated a few hours or a day prior to the start of the service. However, in the recent past, developments in mobile communications, smart phone technology, GPS-based real-time location tracking have enabled real-time demand-aware transit systems. This is exemplified by the global ‘mobility on-demand’ market that is expected to grow to 250 billion USD by 2026 [4]. In the mobility-on-demand market, demand-aware bus transit systems, where mini buses/vans operate with flexible schedules and routes to bridge the connectivity gap are considered a segment with significant growth potential.

However, the need to compute schedules and routes in real-time has made large scale deployment of demand-aware bus transit systems a significantly challenging task due to the large geographical areas that needs to be serviced and the high volumes of ride requests. In addition, demand-aware bus transit systems need to ensure that the waiting and riding times (user experience) are enhanced along with reducing the vehicle miles travelled (VMT) (operator profitability) of the fleet. The complexity is exacerbated by policies that have enforced electric vehicles (EVs) with limited driving ranges into bus fleets. This calls for development of robust and highly responsive techniques, which not only optimize user experience of passengers but also operator profitability. However, existing works have not been able to propose scalable techniques that enable large scale deployment of demand-aware bus transit systems. Thus, in this research, robust and highly responsive techniques for large scale deployment of demand-aware bus transit systems are presented.

1.2 Research Objective

To develop a methodology for large scale deployment of a demand-aware bus transit system using EVs for enhancing user experience and operator profitability by proposing novel techniques for:

Modelling: A technique for modelling a user experience and operator profitability centric demand-aware bus transit system surrounding a transit hub. The technique should generate the optimal flexible routes for the proposed demand-aware bus transit system. The technique is intended to be used to perform diverse what-if analyses by varying different input parameters and observe the performance of the system.

Accelerating Computation: A rapid technique for accelerating constraint-aware flexible route generation for a user experience and operator profitability centric demand-aware bus transit system surrounding a transit hub. The technique is intended to have the ability to rapidly generate near-optimal routes for real-time deployment.

Improving Responsiveness: A novel technique for segmenting the bus transit network to leverage parallel computing architectures for route computation for a user experience and operator profitability centric demand-aware bus transit system surrounding a transit hub. The technique is intended to speed-up computation without compromising the quality of the routes.

Citywide Adoption: A technique for systematically decomposing an urban city into independent regions to facilitate parallel route computation in each region surrounding a transit hub. The technique is intended to be scalable, highly responsive and robust, which enables large scale deployment of a demand-aware bus transit system for enhancing user experience and operator profitability.

Large Scale Deployment: Framework for large scale deployment of a demand-aware bus transit system. The framework is intended to generate flexible schedules and routes to facilitate large scale deployment of a demand-aware bus transit system for enhancing user experience and operator profitability by intelligently leveraging, real-time and near-future demand estimation techniques, historical demand-supply patterns, systematic decomposition of urban cities, offline-online division of workloads, and flexible route generation techniques. Further, the demand-aware bus transit system needs to be accessible to all passengers including old, young, persons with disabilities and low income.

1.3 Organization

The rest of the thesis is organized as follows.

- **Chapter 2** presents an extensive discussion on existing literature. The chapter highlights the connectivity gap in public transit network followed by strategies that bridge the connectivity gap. Next, works on demand-aware transit systems that have emerged as a viable alternative in bridging the connectivity gap are presented. A discussion on vehicle routing problems, which forms the basis of demand-aware transit is followed by different categories of demand-aware transit system, simulators, commercial efforts and techniques for addressing scalability. Works on demand-aware bus transit systems are presented next. The chapter concludes with specific objectives and motivations for the rest of this thesis.
- **Chapter 3** proposes a demand-aware bus transit system for enhanced user experience and operator profitability that links a transit hub to the surrounding geographical zone using a fleet EVs that operate over flexible routes and schedules. In order to generate the optimal flexible routes for the fleet of EVs, the proposed transit system is converted to a graph-based representation and subse-

quently modelled using a mixed integer program. The proposed model can be solved to generate optimal flexible routes for the fleet of EVs of the proposed demand-aware bus transit system.

- **Chapter 4** proposes a heuristic routing technique that facilitates real-time, near-optimal flexible route generation in a geographical zone surrounding a transit hub for the proposed demand-aware bus transit system, which simultaneously enhances the user experience and operator profitability. The technique is based on combining the ϵ -constraint method and a genetic algorithm.
- **Chapter 5** proposes a directionality-centric bus transit network segmentation technique to speed-up route computation in a geographical zone surrounding a transit hub for the proposed demand-aware bus transit system. The proposed technique relies on directionality & shortest-path centric generation of sub-zones, demand-aware allocation of EVs for sub-zones and flexible routing for sub-zones to speed-up computation with minimal impact on the quality of the routes.
- **Chapter 6** proposes a methodology for adopting demand-aware bus transit system for large scale deployment. The proposed methodology relies on decomposing a large geographical region such as an urban city into independent regions by leveraging the hub and spoke topology of public transit network. Then, the methodology partitions the regions into workload balanced zones surrounding a transit hub, where the techniques proposed in Chapter 5 are used to generate near-optimal solutions rapidly.
- **Chapter 7** presents a framework for citywide deployment of demand-aware bus transit system. The proposed framework intelligently integrates techniques for real-time and near-future demand generation with all the techniques proposed in the earlier chapters of the thesis.
- **Chapter 8** concludes the thesis by summarizing the important contributions made in this research followed by recommended directions for future work.

1.4 Publications

The work presented in this thesis have been communicated in international journals and conferences as follows:

International Refereed Journal Papers

[J-1] **Thilina Perera**, Deshya Wijesundera, Lahiru Wijerathna and Thambipillai Srikanthan. "Directionality-centric Bus Transit Network Segmentation for On-Demand Public Transit." IET Intelligent Transport Systems (To Appear).

International Refereed Conference Papers

[C-1] **Thilina Perera**, Alok Prakash and Thambipillai Srikanthan. "A Scalable Heuristic Algorithm for Demand Responsive Transportation for First Mile Transit." 21st *International Conference on Intelligent Engineering Systems (INES)*, 2017.

[C-2] **Thilina Perera**, Alok Prakash and Thambipillai Srikanthan. "A Hybrid Methodology for Optimal Fleet Management in an Electric Vehicle Based Flexible Bus Service" 15th *International Conference on Control, Automation, Robotics and Vision (ICARCV)*, 2018.

[C-3] **Thilina Perera**, Alok Prakash, Chathura Nagoda Gamage and Thambipillai Srikanthan. "Hybrid Genetic Algorithm for an On-Demand First Mile Transit System using Electric Vehicles." 18th *International Conference on Computational Science (ICCS)*, 2018.

[C-4] **Thilina Perera**, Chathura Nagoda Gamage, Alok Prakash, and Thambipillai Srikanthan. "A simulation framework for a real-time demand responsive public transit system." 21st *International Conference on Intelligent Transportation Systems (ITSC)*, 2018.

[C-5] **Thilina Perera**, Alok Prakash and Thambipillai Srikanthan. “Genetic Algorithm Based EV Scheduling for On-Demand Public Transit System.” *19th International Conference on Computational Science (ICCS)*, 2019.

[C-6] **Thilina Perera**, Alok Prakash, Deshya Wijesundera, Thambipillai Srikanthan, and Chathura Nagoda Gamage. “Cluster-First, Route-Second Heuristic for EV Scheduling in On-Demand Public Transit.” *9th International Conference on Cybernetics and Intelligent Systems (CIS) and Robotics, Automation and Mechatronics (RAM)*, 2019.

Chapter 2

Literature Review

2.1 Introduction

In urban cities, the public transit network consisting of buses, trains are constantly challenged to meet the diversified needs of passengers such as reliability and penetration. In general, buses provide connectivity to the train stations, which connect to high-speed, long distance trains. However, bottlenecks such as excessive waiting times and riding times resulting from the fixed schedules and routes of buses have constantly plagued the public transit network. Therefore, alternative transit systems have been explored to bridge the connectivity gap. Among many proposed systems, demand-aware transit systems are the most efficient and widely adopted.

Various alternatives of demand-aware transit systems have been studied. Although, real-time performance was not a requirement during the initial generations, growth in mobile communications, GPS-based location tracking and cashless payments have enabled real-time demand-aware transit systems. As a result, the computation speed of routes for the fleet of vehicles that provide demand-aware transit has become a crucial performance metric. At the same time, the routes need to be near-optimal to ensure that operators earn profits while satisfying passenger needs. Also, demand-aware transit

systems should be scalable to handle the high rates of demand and large geographical areas that need to be serviced.

This chapter will present an overview of existing literature on demand-aware transit and discuss the inadequacy of existing methodologies in realizing large scale demand-aware bus transit systems. The rest of the chapter is organized as follows. Section 2.2 presents characteristics of the public transit network and discusses the limitations. Next, mobility solutions that bridge the limitations are presented. Section 2.3 discusses the vehicle routing problem, categories of demand-aware transit systems, commercial efforts and techniques for managing scalability. Section 2.4 presents work on demand-aware bus transit systems. Finally, conclusions are presented in Section 2.5.

2.2 Public Transit Network

Public transit is a critical service in urban cities due to its high impact on quality of life. The public transit network consists of rail transit systems such as trains and road transit systems such as buses. Rail transit systems run at higher average speeds due to dedicated tracks. In contrast, road transit systems share the road network with other vehicles and hence run at lower average speeds. Buses are broadly classified into feeder services, which provide short distance connections and trunk services, which provide long distance point-to-point connections across regions. In general, the public transit network is connected as a hub & spoke model, where the high-speed rail transit system provides the backbone connectivity between hubs and the slow-speed feeder buses act as the spokes by connecting passengers to the hubs. In addition, trunk buses provide connections between hubs and regions that are farther away. However, passengers generally use feeder buses and rail transit for long distance journeys due to reliability issues of trunk buses [5]. In addition, trunk bus service routes are constantly modified to minimize the duplication with the development of new rail transit systems [6]. Therefore, in a public transit journey, feeder buses pick passengers from a neighbourhood and bring them to a hub, typically a train station,

where passengers connect to the high-speed train services and vice versa. As a result, a public transit journey consists of multiple legs that uses different transit services. Thus, works have been focused on developing the connectivity between feeder buses and the rail transit system. However, several features of feeder buses create a connectivity gap in the public transit network. This section highlights the connectivity gap that has led to alternative transit solutions.

2.2.1 Connectivity Gap in Public Transit Network

Based on the hub & spoke model of the public transit network, passengers have to rely on feeder buses to travel to the transit hub, which provides high-speed long distance connectivity. Thus, passengers have to walk from their origin to the nearest bus stop to connect to the feeder bus service. It has been studied that most passengers are comfortable walking up to 400 m (1/4 mile) to the nearest bus stop [7]. Also, the "comfortable distance" depends on several external factors such as weather, demographics and geography among others [8]. For example, walking choice can be influenced by the climate, where people tend to avoid walking under the sun as it can be more tiring. People prefer the walking path to have a roof or trees so that they can be sheltered from the sun rays and rain.

After reaching the bus stop, passengers have to wait until a bus reaches the bus stop. However, feeder buses operate with fixed schedules. Therefore, in some instances passengers have to tolerate long waiting times. At the same time, feeder buses operate on fixed routes. Therefore, the riding time on the feeder buses is generally high compared to the direct riding time from the bus stop and the destination due to large detours. Similarly, the operator has to schedule buses at fixed frequencies even if passengers are not present in the bus stops. Buses always have to ply the fixed routes resulting in increased fleet travel distances. Further, as the operator is not aware of the demand, fixed capacity buses have to be scheduled. Fixed capacities cause under-utilization of feeder buses, which results in low profits for the operator. A study based in Singapore reveals that public transit utilization during off-peak periods

is significantly low compared to the peak periods [9]. Thus, operators lower the fixed frequencies of bus services during off-peak periods. Therefore, passengers are hindered by undesirable additional waiting times at bus stops. Thus, it is identified that fixed routes and schedules of the feeder bus services create a spatial (long travel distances) and temporal (additional waiting and riding time) gap in the public transit network while connecting to the transport hub, which is defined as the connectivity gap.

2.2.2 Strategies for Bridging Public Transit Connectivity Gap

The aforementioned root causes need to be addressed in order to enhance public transit services to provide a better quality of service to passengers. To this end, research and commercial efforts have proposed traditional solutions that enhance existing transit infrastructure or innovative solutions that leverage novel technical developments. Traditional solutions focus on improving the infrastructure for walking and developing facilities around transit hubs or enhancing mobility using taxi services. However, innovative solutions which are economical, greener and comfortable have emerged over the years. Solutions such as bike-sharing, personal mobility devices (PMD) and car-pooling fall into this category. These solutions range in complexity, adding to the enhancement of the overall mobility of passengers and to the effectiveness and utilization of existing transit network. Further, developments in modelling tools and increasing computational power of modern computers have enabled research in this direction helping to optimize the usage of these solutions. It should also be noted that inevitably in almost all public transit trips some form of walking is always included. As a result, some transit users may achieve 30 minutes of physical activity daily solely by walking to and from transit [10]. Therefore, solutions such as bike-sharing and PMD are preferred over taxi services, private vehicles and car pooling due to health benefits.

Transit-Oriented Development

Transit-oriented development (TOD) is the creation of compact, walkable, pedestrian-oriented, mixed-use communities centred around high-quality transit systems [11]. The benefit of TOD is the possibility to live a lower-stress life without complete dependence on a car for mobility. One of the major features of TOD is providing enhanced connectivity by foot to a transit hub surrounded by supermarkets, hospitals and recreational centres. To this end, one of the most common efforts is building sheltered walkways such that passengers can walk more comfortably. This includes connecting the major transit hubs with covered walkways to nearby hospitals and schools under programs such as Walk2Ride [12]. Also, amenities such as benches, shelters, and lighting at transit stops are developed for pedestrian comfort and safety. Thus, more pedestrian oriented urban city planning tasks have been a focus among researchers [13]. However, implementing these solutions in large scale is significantly costly and time consuming. Thus, TOD is generally focused only on major transit hubs in a city.

Taxi Services and Private Vehicles

Taxi services and private vehicles are used as an alternative to bridge the connectivity gap. Formally, taxis and private vehicles are termed as single occupancy vehicles (SOV). It is reported that in the journey to work, about 87.5% of trips in the U.S. happen in automobiles, and 76% of the trips are individuals driving alone [14]. Similarly, a study in Singapore shows that the average car occupancy is 1.7 compared to a public transit bus, which can accommodate 85 to 143 passengers and the six-carriage train with 1800 passengers [15]. Therefore, SOVs have the worst impact on the environment, increasing traffic congestion and greenhouse gas emissions. Therefore, it is not a sustainable solution that can bridge the connectivity gap.

Bike-sharing

One of the earliest methods adopted to bridge the connectivity gap is using shared bicycles, which is formally termed as bike-sharing systems. Bike-sharing systems allow passengers to rent a bicycle on a short-term basis from a place closer to the origin and ride it to the nearest transit hub and continue the journey. Thus, passengers can eliminate unnecessary waiting times at bus stops and excessive riding times on buses by taking the shortest path to the destination.

An extensive study on the history of bike-sharing systems is presented by De-Maio [16]. The pioneering generation of bike-sharing systems in Amsterdam enabled users to hire a public bike to ride to the bus stop and leave the bike at the bus stop. However, a more useful bike-sharing systems with user registration was introduced in England. This system was significantly improved with developments of location tracking and on-board computing units. The latest generation of bike-sharing systems utilize technological developments of mobile communications and GPS tracking to provide enhanced real-time solutions.

Bike-sharing systems have evolved significantly from their inception. Current bike-sharing systems support seamless integration to transit hubs by facilitating dedicated lanes, barrier free access and user-friendly docking stations. As a result, bike-sharing has gained widespread popularity in the recent past. This is confirmed by the estimated 136 bike-sharing programs that existed in 165 cities around the world in 2011 [17]. Further, bike-sharing systems are environmentally friendly, which contributes significantly to the popularity. A study in the U.S. in 2015 looked at bike-sharing systems in five cities, including Washington, D.C., and Minneapolis, and found that users substituted rides via bike-shares for car trips 8 percent of the time in Washington, D.C. and almost 20 percent of the time in Minneapolis [18]. A similar study in 2012 in Denver, Colorado states that 43 percent of bike-sharing trips have successfully replaced car trips [19].

However, a common pitfall of bike-sharing systems is the service coverage. As the operators are concerned about the economic viability of the solution, most of the bike-sharing solutions are limited to major transit hubs. Although this improves the economical prospective of the solution, the goal of bridging the connectivity gap, especially in remote transit hubs is not facilitated. Also, as highlighted by the work of Kumar et al. the number of passengers using bike-sharing systems reduce drastically when the travel distance is beyond a comfortable cycling distance [20]. Also, current generation of bike-sharing solutions require passengers to possess electronic payment options and thus, limits the number of potential users. Further, bike-sharing solutions are inaccessible to the less mobile and elderly community and also hindered by adverse weather conditions.

Personal Mobility Devices

Lately, PMDs have emerged as another attractive alternative to bike-sharing systems. Similar to bike-sharing systems, PMDs provide a flexible and convenient alternative to bridge the connectivity gaps. The compact size of PMDs is a major contributing factor which gives PMDs the edge over the traditional bicycle counterparts. As a result, over the past few years, the group of PMD users have rapidly expanded and has been a popular medium for bridging the connectivity gap among the younger generations, especially over long distances.

Numerous efforts have been undertaken to study the use of PMDs as a service medium for bridging the connectivity gap. Dowling et al. [21] study the impact of different size PMDs in the road infrastructure in a pilot project within Macquarie University, Australia. The study reveals that most of the people prefer using PMDs as it is much convenient than bicycles and faster than walking. However, the authors also highlight the importance of enforcing strict rules when using the devices to maintain safety of all parties, since PMDs share the walkways with pedestrians. However, large scale usage of PMDs as a solution to the connectivity gap is still not viable due to the

restricted urban spaces. Further, PMDs are also inaccessible to the less mobile and elderly community and hindered by adverse weather conditions.

Casual Car-Pooling

Casual car-pooling is another option deployed to bridge the connectivity gap. Casual car-pooling systems can be used for not only improving the connectivity gap but also for long distance public transit journeys. In general, casual car-pooling refers to the sharing of a ride with a driver and one or more passengers, where the ride-sharing between the individuals is not established or prearranged well in advance but coordinated shortly before the trip [19]. However, there are different forms of car-pooling systems which primarily have the same features with minute changes in operational details. For example, in slugging, a passenger walks to the driver's origin, boards at the driver's departure time, alights at the driver's destination, then walks from there to the passenger's own destination [22].

The history of car-pooling systems dates to the 1970's in U.S. As a result of soaring gas prices, the government had decided to curb gasoline consumption. Among many measures taken to mitigate the rising gasoline prices, one measure was to introduce high occupancy vehicle (HOV) lanes. The HOV lanes were for vehicles with more than three occupants. As a result, people started to offer rides to passengers going along the same direction in order to ensure that vehicles met the minimum number of passengers' criteria to occupy the HOV lanes. This practice, which was unofficially a form a casual car-pooling later adopted to car-pooling practices around the world [23]. A similar system was deployed in Singapore as well. As a result of the worsening traffic conditions on roads due to SOVs, the government had introduced a tax system. Vehicles with 4 passengers were excluded from this tax system. Therefore, car-pools had become a popular option, where drivers picked up 3 other passengers from bus stops and offered free rides to popular destinations in the city. However, due to misuse, it was discontinued [15].

Car-pooling systems have several common features. Car-pooling occurs between drivers and passengers going along the same direction. Thus, common transit hubs are popular origins/destinations for car-pooling systems such as the Pentagon, which is a major community hub in U.S. [23]. Similarly, rides are shared to and from residential neighbourhoods to downtown business districts. Another feature is the instant matching of drivers and passengers. Car-pooling drivers and passengers can find their carpools "on the fly" instead of formalized booking systems. This is an added benefit as participants do not have to meet at a fixed time. This increases the travel flexibility and better accommodates unscheduled needs to share a ride. However, the reliability of casual car-pools is significantly low. Thus, passengers can often encounter large delays which limits the usage in bridging the connectivity gap.

Although, the above mentioned strategies have been deployed to bridge the connectivity gap in public transit network, limitations such as significant costs, traffic congestion, increased greenhouse gas emissions, accessibility to less mobile & elderly communities and less reliability have hindered the widespread popularity. However, demand-aware transit systems, which are characterised by flexible routes and schedules and smaller vehicles operating in shared-ride mode between pick-up and drop-off locations according to passenger needs [24] that arrange one-time shared rides have become increasingly popular. Demand-aware transit systems have also been used to bridge the connectivity gap. For example, Uber [25], in Altamonte Springs, Florida provides connectivity from a transit hub using a real-time demand-aware transit system [26]. Logan et al. mention that demand-aware transit is the hottest and fastest evolving mobility option on the market [27]. Further, authors claim that it can be used to conveniently bridge connectivity gap for users by providing quick and easy connections to and from bus stops/transit hubs. Therefore, demand-aware transit is considered as an attractive alternative to bridge the connectivity gap. In the next section, literature on demand-aware transit systems are presented.

2.3 Demand-aware Transit Systems

Demand-aware transit systems require designing vehicle routes for users who specify pick-up and delivery requests between origins and destinations either in advance or in real-time. Systems that handle ride requests received in advance are known as static demand-aware transit. On the other hand, dynamic demand-aware transit systems satisfy real-time requests. Therefore, in static systems, all passenger requests are known at the time of route computation as opposed to dynamic systems, where routes are computed instantaneously when passenger requests arrive. Although, static demand-aware transit systems have been a common practice, especially to facilitate para-transit services, dynamic demand-aware transit has been a viable option only in recent years due to the technological advancements in GPS-based location tracking, smart phones, wireless communications [28]. As a result, passengers with access to a smart phone and registered with a service provider can request the service in an instantaneous manner. The demand-aware transit platform would then match the request of the riders with the drivers providing the service in real-time. Thus, dynamic demand-aware transit systems have gained widespread popularity.

This problem in the most elementary form is known as the vehicle routing problem (VRP). It has been one of the most studied problems in mathematical optimization and operations research. This section presents the different types of VRPs followed by the categories of demand-aware transit systems. Next, commercial efforts and techniques for managing scalability of demand-aware transit systems are presented. Earlier literature on demand-aware transit has been focussed on goods transportation, where requests are mostly received in advance. Although, this has limited applications in the context of public transportation, the concepts presented in these works can be applied for the dynamic case. Therefore, some works that are relevant are highlighted in this section.

2.3.1 Vehicle Routing Problem

The VRP consists of designing m vehicle routes for n passengers. A VRP with a single vehicle is termed as the travelling salesman problem (TSP). The objective of the TSP is to find the shortest route for a travelling salesman who, starting from his home city, must visit every city on a given list precisely once and then return to his home city. This is similar to a demand-aware transit system, where a vehicle must pick-up all the passengers that have requested the service and drop-off them at the destination, which can be modelled as another city. However, as demand-aware transit systems are provided by a fleet of vehicles, the applicability of TSP is limited to cases where the subproblems are solved as a TSP in the context of a VRP.

The main difficulty of the TSP is the immense number of possible tours: $(n-1)!/2$ for n cities [29]. In a mathematical form, this is defined as, given an integer $n > 3$ and an $n \times n$ matrix $C = (c_{ij})$, where each c_{ij} is a non-negative integer, which cyclic permutation π of the integers from 1 to n minimize the $\sum_{i=1}^n c_{i\pi(i)}$. Even though, TSP can be easily defined in a mathematical form, solving the problem optimally is difficult. This falls under the class of NP-hard problems, where no polynomial time algorithm is known with which the routes can be verified to be optimal. However, the problem has been studied a lot due to the widespread implications of routing problems in many domains, such as transportation, cryptanalysis and statistics.

Broadly, there are two types of solvers used for finding solutions of the TSP, namely exact solvers and non-exact solvers. Further, the group of exact solvers, are classified into solvers which solve relaxations of the TSP linear programming formulation and dynamic programming methods. TSP linear programming formulations are solved using methods such as cutting plane, branch-and-bound and branch-and-cut. However, as the TSP is an NP-hard problem, research efforts have focused on finding efficient heuristic solutions, comprising the class of non-exact solvers. Kirkpatrick et al. tried to solve the TSP using simulated annealing [30]. Brady applied the principals of genetic algorithms to solve the TSP [31]. A neural network-based approach to the TSP was studied by Hopfield [32]. Further, heuristic algorithms such as tabu search, ant colony

optimization, insertion heuristic, nearest neighbourhood search and clarke-wright heuristic have also been used to solve the TSP.

VRP was first introduced by Dantzig and Ramser in the form of "Truck Dispatching Problem", where a fleet of homogeneous trucks could serve the demand for oil of a number of gas stations from a central hub with a minimum travelled distance [33]. The classical VRP known as the homogeneous fleet VRP, has been extended in many ways by introducing additional real-life constraints or characteristics, resulting in many variants of the VRP.

One of the most common methods of extending the classical VRP has been to vary the size of the vehicles resulting in the heterogeneous fleet VRP or mixed fleet VRP. This has been the basis for Clarke and Wright, where the authors generalized the VRP to a linear optimization problem that is commonly encountered in the domain of logistics and transport: i.e., how to serve a set of customers, geographically dispersed around the central depot, using a fleet of trucks with varying capacities [34].

The fleet routing and scheduling problem was introduced and solved using an integer linear program by Levin [35]. Here, the author formulates the problem as a 0-1 integer linear program. Marks et al. present an algorithm for the Chinese postman problem, which is practically used in public services such as trash collection, snow ploughing and street cleaning [36]. Eilon et al. discuss the problems of depot location, vehicle scheduling, loading and the determination of fleet size in distribution management [37].

The first research literature bearing the term "vehicle routing" in its title is attributed to Golden et al. [38], where the authors study the multi-depot VRP in the context of urban newspaper distribution. The multi-depot VRP assumes that multiple depots are geographically spread among the customers [39]. This is specifically an important scenario in realizing demand-aware transit systems.

The stochastic VRP (SVRP) was proposed by Cook and Russell, where the authors examine a generalization of the vehicle scheduling problem in which time constraints

as well as stochastic demand and travel times are considered [40]. The objective is to minimize total travel time subject to vehicle load capacities and sequencing restrictions, meaning that certain points may be serviced only in specified time intervals. A similar study from Golden and Stewart, where the authors introduced stochastic demands compared to fix demands is presented in [41].

An important variant of the classical VRP was introduced by Solomon, where the author considers the design and analysis of algorithms for VRP with time window (VRPTW) constraints [42]. The VRPTW assumes that deliveries to a given customer must occur in a certain time interval, which varies from customer to customer.

The first instance of dynamic VRP (DVRP) in research literature is credited to Powell [43]. The authors study the multiple time period vehicle allocation problem under random demands. Larsen et al. introduce the partially dynamic travelling repairmen problem and present several dynamic policies to minimize routing costs [44]. Fleischmann et al. present a study on using dynamic travel time information, which is used for a dynamic routing system that dispatches a fleet of vehicles according to customer orders arriving at random during the planning period [45].

With the addition of real-life constraints to the VRP, a new set of routing problems known as the rich VRPs have been introduced in the recent past. Here, some of the standard problems have been revised with addition of novel constraints such as, VRP with pick-up and delivery, VRP with backhaul and periodic VRP. In the VRP with pick-up and delivery, goods need to be picked up from a certain location and dropped off at their destination. Also, as the pick-up and drop-off must be done by the same vehicle, the pick-up location and drop-off location must be included in the same route [46]. VRP with backhaul is when a vehicle does deliveries as well as pick-ups in one route. Some customers require deliveries (referred to as linehaul) and others require pick-ups (referred to as backhaul). The combination of linehaul and backhaul has been proven very valuable to the industry. The periodic VRP is used when planning is made over a certain period and deliveries to the customer can be

made in different days. For the periodic VRP, customers can be visited more than once, though often with limited frequency [47].

Also, with the advent of EVs and the usage in public transportation, more works have been attracted to the electric vehicle routing problem (E-VRP). The E-VRP adds constraints to the traditional VRP in terms of limitation in the driving range of the vehicles and also the high charging time. These limitations result in 'range anxiety', which is the fear of running out of battery and the concern of making an unplanned trip [48].

Erdogan and Miller-Hooks is credited for introducing the green VRP problem, which considers not only EVs but also alternative fuel vehicles (AFV) in the fleet [49]. Similar to the classical VRP many variants of the E-VRP have been proposed such as partial charging [50], time windows and recharging [51] and mixed fleets [52]. A comprehensive review of literature on E-VRP and the hybrid vehicle routing problem is presented in [53] and [54] respectively.

Similar to the TSP, the VRP is also a NP-hard problem [55] thus, exact solvers are only able to find optimal solutions for small problem instances. Heuristics and meta-heuristics are often more suitable for practical applications, because real-life problems are considerably larger in scale. However, VRPs are in general much more difficult to solve than TSPs of the same size. Effort has been devoted to the solution of the TSP by exact methods, where most algorithms for the VRP are heuristics, reflecting the relative difficulty of the problem. The works by Laporte et al. [56] [57] present a comprehensive study on exact and heuristic algorithms used for the VRP respectively.

As microcomputer capability was improved, researchers managed to develop and implement more complex search algorithms. More powerful computers and enhanced coding capability helped researchers model complex problems much easily. As a result, meta-heuristics was introduced for solving the VRP as well as other combinatorial optimization problems. Wasserman applied neural networks to the domain of VRP [58]. Dorigo et al. applied the ant colony optimization technique to the VRP [59]. Aarts and

Lenstra studied local search heuristics in the context of VRP [60]. Glover and Lagune introduced the tabu search meta-heuristic approach [61]. Gendreau et al. presents a comprehensive study on the application of meta-heuristics to the domain of VRP [62].

The TSP and VRP have evolved significantly from its' initial versions with the latest developments in GPS-based location tracking, mobile communications and cashless payment. Latest works consider more complex problems and also impose real-life constraints. Similarly, solutions methods have evolved from exact solutions that focus on the TSP to efficient heuristic and meta-heuristic solutions for the VRP. Applicability of these problems and constraints to the context of demand-aware transit for bridging public transit connectivity gap can be identified. For example, multi-depot VRP resembles different origins of the vehicles in the fleet, the stochastic VRP can be used to anticipate the future demands in order to make better routing decisions, VRP with time windows is similar to scenarios where passengers request for advance rides indicating a pick-up time. Thus, the next sub section presents the different categories of demand-aware transit systems and applicable literature for each category.

2.3.2 Categories of Demand-aware Transit Systems

Section 2.3.1 discussed the variants of VRP, which forms the basis of demand-aware transit systems. However, when deploying demand-aware transit systems, different features of the vehicles & requests and objectives of the systems need to be considered. For example, an important factor that differentiates the traditional VRP from demand-aware transit is the human perspective to the problem. Therefore, it needs to balance the user-inconvenience with the objective of minimizing operating costs of the service provider. As a result, the optimizing routes in demand-aware transit system has to be studied as a multi-objective optimization problem. This section presents the categories of demand-aware transit literature in the context of passenger transportation and highlights works based on the provided categorization. A comprehensive classification of demand-aware transit literature can be found in [63] [64] [65].

Demand-aware transit systems are categorized based on multiple criteria. For example, based on the mode of operation, it is classified into static and dynamic modes. In the static mode, all passenger requests are known beforehand, while in the dynamic mode requests are served immediately when received. However, it should be noted that in practice, pure dynamic demand-aware transit systems rarely exist since a subset of requests is often known in advance [66]. Further, it is categorized based on the type of vehicles in the fleet into homogeneous [33] and heterogeneous fleets [67] [68], the number of depots into single [69] and multiple depots [39]. Based on the problem type, it is classified into problems which minimize costs subject to full demand satisfaction and side constraints or maximize satisfied demand subject to vehicle availability and side constraints. An important side constraint is the quality of service parameter. The quality of service parameter is measured by route duration, route length, customer waiting time, customer ride time, difference between actual and desired delivery times. Some of these parameters may be treated as constraints or as part of the objective function based on the goal of the problem. Also, based on the objective, problems are classified as single-objective optimization and multi-objective optimization problems.

Static Request Based

Psaraftis presented a dynamic programming-based solution to a static, single vehicle demand-aware transit problem [70]. The objective of the problem is to reduce a generalized weighted linear function comprising of the time to service all customers and the total degree of dissatisfaction experienced by customers while waiting for service. Time-window constraints are integrated to the algorithm in [71]. However, as both cases formulate an exact solution, it is only applicable to instances with relatively low demand.

Sexton and Bodin study a static, single vehicle demand-aware transit problem as a sub problem for a single cluster in a multi-vehicle demand-aware transit system [72] [73]. The presented algorithm iterates between solving a routing problem by means of an insertion heuristic and solving the associated scheduling problem.

The single vehicle, static demand-aware transit problem is reformulated as an integer program by Desrosiers et al. [74]. The formulation includes time windows, vehicle capacity and precedence constraints. Similar to [71] the algorithms work only for relatively small cases of the problem.

Jaw et al. [75] proposed a heuristic algorithm for a static, multi vehicle demand-aware transit problem. The algorithm uses a sequential insertion procedure to assign customers to vehicles and to determine a time schedule of pick-ups and deliveries for each vehicle. The authors present computational results for a simulation with a real database of 2600 customers and 20 active vehicles.

Cordeau and Laporte presented a tabu search heuristic for a static, multi vehicle, demand-aware transit problem [69]. In this work, users provide a time window on their desired departure or arrival time and transportation is supplied by a fleet of vehicles based at a common depot. Side constraints such as vehicle capacity, route duration and the maximum ride time of a user are considered in the formulation. Authors claim that with respect to alternative algorithms such as column generation and branch-and-cut, tabu search can easily accommodate a large variety of constraints and objectives, even if these are non-linear.

Cordeau [76] and Ropke et al. [77] presented two exact branch-and-cut algorithms to the static, multi vehicle, demand-aware transit problem studied in [69]. Here, authors formulate the problem as an integer linear program. Even though, these methods improve the results, the largest instance solved with the two methods are 36 and 96 requests, respectively.

Xiang et al. presented a heuristic for solving a large-scale, static, demand-aware transit problem with complex constraints [78]. The authors use a local search strategy and a diversification strategy to improve the initial solution. Next, the results are further improved by an intensification strategy. Authors show the benefits of constructing good initial solution which results in reducing the computational time significantly.

Authors claim to solve larger instances up to 2000 requests in less than 10 hours on a popular personal computer.

Literature reveals that complexity of the considered static, demand-aware transit systems are relatively low when exact methods are used. In contrast, heuristic approaches can solve relatively large problems. Initial works focus on the single vehicle case where as the latest works focus on vehicle fleets. However, since requests are known well in advance, the runtime for computation is not a significant issue in static demand-aware transit systems. Thus, the applicability of techniques for demand-aware bus transit is relatively low.

Dynamic Request Based

Psaraftis introduced the single vehicle, demand-aware transit problem for the dynamic case [70]. Here, new requests occur dynamically in time and no information on future requests is available. However, if the algorithm is too slow, it may not be possible to solve a single request before the arrival of the next request. Thus, if requests arrive in quick succession, the response time of the algorithm will be severely affected. One workaround to this approach, proposed by Gendreau et al. is to pre-compute several scenarios, using parallel computing, in anticipation of future requests [79].

Gendreau et al. proposed a neighbourhood search heuristic to optimize the planned routes of vehicles, where new requests, with a pick-up and a delivery location, occur in real-time [80]. New solutions were explored through a neighbourhood structure based on ejection chains. Numerical results showed the benefits of these procedures in a real-time context. This work used the single vehicle, demand-aware transit problem as a sub problem of solving the multi-vehicle, demand-aware transit problem.

Teodorovic and Radivojevic proposed two approximate reasoning algorithms, for a dynamic, demand-aware transit problem where the first will make the decision on which vehicle will accept the new request while the second algorithm is used to design the new route for the chosen vehicle [81].

Coslovich et al. proposed a two-phase strategy for the insertion of a new request into an existing route [82]. An offline phase is first used to create a feasible neighbourhood of the current route through a 2-opt mechanism. An online phase is then used to insert the new request with the objective of minimizing user dissatisfaction.

Hanne et al. proposed to model intra-hospital transport of passengers for diagnostic and therapeutic reasons as a dynamic, demand-aware transit problem [83]. A similar work on patient transportation is presented in Beaudry et al. [84]. Here, authors study a scenario of dynamic arrival of passenger requests and propose a two-phase heuristic capable of quickly inserting new requests in the current vehicle routes. The two-phase heuristic consists of an insertion scheme to generate feasible solutions and tabu search algorithm for improvement. Authors claim to handle dynamic aspect of the problem by providing high-quality solutions by minimizing the waiting times for patients while using fewer vehicles.

Häll and Peterson study the effects of using ruin and recreate methods in a re-planning phase of a dynamic, demand-aware transit problem [85]. Authors claim that better results can be obtained by using the ruin methods based on removal of sequences of requests.

Wong et al. study operational issues of the overall system performance of a partially dynamic, demand-aware transit problem [86]. Authors conclude that the dispatching system incurs higher transportation costs and accepts fewer requests when the request arrival is partially dynamic, as compared to fully static or dynamic system.

Santos and Xavier studied a dynamic, demand-aware transit system based on a taxi service [87]. Authors use a setting where passengers put requests using a mobile phone application to specify the current location, destination and maximum cost. Car owners and taxi drivers also update the origin, destination and start time of service. The problem is to compute routes, matching requests to vehicles in such a way that ride-sharing is allowed within the constraints. Authors propose to handle new dynamic

requests by dividing a day into time periods and solving the problem using a greedy randomized adaptive search procedure.

Compared to literature on static requests, dynamic requests are commonly encountered in the recent years due to the increase in real-time demand-aware transit systems. Literature reveals that dynamic, demand-aware transit systems are mostly solved using heuristic approaches due to the significant complexity of the problems and responsiveness requirements. Most heuristic techniques use a two-phase approach, where vehicle selection and route computation are handled separately. Also, works propose to breakdown long planning horizons in order to solve dynamic problems using a near real-time technique. However, in most cases experiments are conducted with low complex scenarios.

Heterogeneous Fleet Based

Madsen et al. [88] solve a real-life, demand-aware transit problem involving services to elderly and disabled people. In this work, user requests arrive dynamically throughout the day with a desired pick-up or delivery time window. Also, several types of vehicles are used to provide the service and it may become unavailable due to breakdowns. The authors develop an insertion algorithm based on [75]. Authors report results for an instance with 300 requests and 24 vehicles.

A real-life problem, where transportation is supplied by a fleet of capacity constrained minibuses and special cars is presented in Toth and Vigo [89]. The authors presented a parallel insertion procedure, which is improved using a tabu thresholding post-optimization phase.

Wong and Bell presented a modified insertion heuristic to solve a demand-aware transit problem with a heterogeneous fleet [90]. The objective of the work is minimization of a linear combination of total operating time, passenger ride time and taxi cost for unassigned requests. The authors claim that the proposed methodology produces superior results when compared with the classic algorithms.

In the conventional demand-aware transit problem, a single vehicle is used for the whole journey. However, due to practical requirements such as moving passengers with limited mobility at airports, recent literature present works that allow passenger transfers between vehicles. For example, the works by Cortés et al. [91], Reinhardt et al. [92], Masson et al. [68] study demand-aware transit systems with passenger transfers in the middle of the journey. The main challenge in these works is to synchronize the passenger transfer such that the user waiting time is minimized. Authors claim that even though the trips become less direct, such transfers have many benefits. Cortés et al. [91] show that the overall efficiency of the system is increased as a result of reducing total ride time. Similarly, Masson et al. [68] show savings in terms of total travel distance.

Parragh [67] studies transporting patients and disabled people with four different types of constraints. The author considers staff seats, patient seats, stretchers and wheelchair places and presents models to match different types of users and vehicles. Similarly, the work by Detti et al. [93] studied non-emergency transportation of patients. Authors study features such as heterogeneous vehicles, vehicle–patient compatibility constraints, quality of service requirements and patients’ preferences. Similar works by Qu and Bard [94] also study different configuration options for vehicles.

Although, most of the literature consider homogeneous fleets, special scenarios such as patient transportation, disabled people movements have considered heterogeneous fleets which consists of different types of vehicles. The proposed algorithms can be helpful in scenarios if the fleet consists of multiple types of vehicles in demand-aware transit system. At the same time, the proposed algorithms can be used if multiple bus services have to be synchronized to move passengers from the origin to the destination.

Multiple Objectives Based

Colorni and Righini proposed to optimize three different objectives, namely the maximization of serviced requests, the maximization of the perceived level of service by users and the minimization of travelled distance [95]. The insertion of new requests is handled by a clustering phase. The routing algorithm applies branch-and-bound to a set of requests whose time windows are not too distant in the future.

Aldaihani and Dessouky presented a hybrid, demand-aware transit system which offers two types of services, namely door-to-door transportation, and transportation to and from a bus stop [96]. The authors developed a two-phase heuristic consisting of an insertion-reinsertion descent phase, followed by tabu search. Authors consider two objectives, namely minimization of vehicle travel distance and minimization of passenger travel time.

Rekiek et al. presented a genetic algorithm to solve the handicapped person transport problem [97]. The objective of the authors is to implement grouping genetic algorithm to find near-optimal routes for transporting handicapped people with highest service quality and minimum number of used vehicles. Experimental results claim improvements in terms of service quality and computational effort.

Chevrier et al. proposed three evolutionary algorithms to find routes for a multi-objective demand-aware transit system [98]. The objectives of the study are to minimize the total number of required vehicles, reduce the total journey time of vehicles and delays. The authors extend this work in Chevrier et al. [99] by providing computation time as input data to be used as the stopping criterion. Here, authors present the results obtained after 1 minute to show the real-time capability of the algorithm and compare it with the results obtained after 10 minutes to validate the quality of the solution. Authors claim that good quality solutions can be obtained in 2 minutes and thus, reduce the deadline to accept ride requests.

Mendes et al. [100] proposed to optimize three objectives of a demand-aware transit system that connects passengers to mini buses. Authors consider both the

operator and user perspectives while solving the resulting problem. They use a random method to generate initial solutions and then combine it with NSGA-II algorithm.

Viana et al. [101] proposed a demand-aware transit service for users who formulate requests specifying desired locations and times of pick-up and delivery. The problem is modelled as a multi-objective problem and solved by three multi-objective evolutionary algorithms.

As highlighted earlier, the demand-aware transit problem for bridging the connectivity gap has to be modelled as a dual-objective optimization problem. However, limited literature has considered the case of optimizing multiple objectives. At the same time, the runtime of proposed algorithms is significantly high and thus, not suitable for real-time demand-aware transit systems.

Electric Vehicle Based

Scheldes and de Almeida Correia studied a demand-aware transit system by replacing conventional vehicles with EVs [102]. However, the authors only analyse a personalized EV system and hence the complexities arising as a result of multiple passengers sharing an EV is not considered.

Masmoudi et al. studied a demand-aware transit system for customers with special needs and/or disabilities [103]. The fleet consists of EVs and authors assume that battery swapping is possible in the middle of the journey. Authors propose three enhanced evolutionary variable neighbourhood search algorithms to solve the problem.

Zhu et al. proposed a dynamic, path-planning strategy based on a greedy algorithm for a peer-to-peer ride-sharing service [104]. Authors claim that the proposed algorithm reduces the computation time by 22% in comparison to an exhaustive search. In the experiments, authors use characteristics of a Tesla model S car with five seats and supercharger facilities.

Keskin et al. studied the EV routing problem in a setting where vehicles arriving at a recharging station have to queue before being recharged [105]. Authors model the problem using a mixed integer linear program and solve it for small instance. For larger instances, author propose to combine an adaptive large neighbourhood search with the mixed integer linear program.

Bongiovanni et al. studied a demand-aware transit problem with electric autonomous vehicles [106]. Authors study the impact of battery management, detours to charging stations, recharge times and selection of destination depots. Authors formulate the problem as a mixed-integer linear program and propose a branch-and-cut algorithm to solve it. Experiments are performed with 5 vehicles and 40 requests.

With strict enforcements on fossil fuel-based vehicles, operators are forced to incorporate electric vehicles in the fleet. As a result, EV-based demand-aware transit literature can be identified in the recent past. However, existing works mostly consider personalized EV systems and hence ignore the complexities of sharing rides. Also, several works consider battery swapping at the middle of a journey, which can be significantly costly for an operator to maintain. Also, the scalability of the proposed algorithms for large fleets have not been studied.

Dynamic Stochastic Features Based

Several works also focus on different types of stochastic features such as future user requests [107] [108], user cancellations [107], stochastic travel times [107] [109] and stochastic locations [110]. These scenarios are used to better prepare for the future unexpected events and make more informed decisions compared to the dynamic, deterministic cases. Therefore, these works can be applied in demand-aware transit systems to anticipate the demand for public transit.

2.3.3 Simulators

Prior to the implementation of algorithms using actual vehicles, it is essential that such a system can be tested and validated using simulations [111]. This not only saves a considerable amount of time but also reduces the cost of implementation. Furthermore, simulations can help to easily create realistic scenarios which can occur in real-world implementation. Also, evaluation and demonstration of a proposed idea prior to implementation and testing and training the users through simulation models can immensely benefit transit service providers [112]. Additionally, simulations help to visualize the flow of vehicles in order to understand the behavior. Also, the complexity of describing some systems using analytical models calls for simulation models. Thus, works have studied several simulation frameworks that can be used to test and validate demand-aware transit systems.

Simulation models are classified as macroscopic, mesoscopic and microscopic models based on the level of details incorporated in the model [113]. A macroscopic model describes entities and their activities and interactions at a low level of detail. In contrast, a microscopic model describes both the system entities and their interactions at a high level of detail. Mesoscopic models generally represent entities at a high level of detail but describes their interactions at a much lower level of detail. However, in the case of demand-aware transit systems, each vehicle needs to be controlled independently and hence, microscopic simulations are preferred. Existing microscopic traffic simulation packages such as PARAMICS [114], VGSim [115], SUMO [116] and Aimsun [117] provide functionality for route choice, traffic light control, vehicle communication and demand modeling. Further, these tools allow users to simulate road networks of a given map.

Macedo et al. [118] propose a model to simulate electric buses in a realistic urban mobility environment. They propose to couple SUMO with a model of an electric vehicle generated using MatLab/Simulink. However, the main goal of this work is to model electric vehicle behavior. Similarly, Mayer et al [119] propose a simulation model and its corresponding discrete event simulator for a rich vehicle

routing problem. Bischoff et al [120] propose an approach to dynamically simulate ridesharing services with a fleet of vehicles of a capacity between two and four. Nagavarapu et al. [111] propose a simulation platform developed using Vissim to test vehicle routing algorithms in a fleet management system. Further, it allows users to perform a graphical illustration of vehicles serving on-demand customer requests. Fiedler et al. [121] studies a demand-model for ride-sharing systems and use the multi-agent transportation simulation framework AgentPolis to test the system using the data from the city of Prague.

2.3.4 Commercial Efforts

Commercial efforts of demand-aware transit, which pools two/three passengers to a single vehicle (often a car with limited seats) have emerged lately. The main difference compared to taxi services in this scenario is the ability to share the rides. These services are often run by private companies and provide point-to-point, demand-aware transit services. GrabShare [122], UberPool [25], Ryde [123], SWAT [124], Shotl [125], Bridj [126] and Via [127] are few examples of such services. Existing commercial efforts are mainly focused on the use of small vehicles, thus complexity of the problem is relatively low. Responsiveness of these works depends on the availability of vehicles and hence passengers are not guaranteed of the service. Several works also do not guarantee a maximum travel time restriction and hence the user experience is relatively low. Further, some of the works require advance requests to cater to the relatively high computation times of the algorithms.

2.3.5 Scalability Issues

Demand-aware transit systems that need to be deployed for bridging the connectivity gap in public transit network necessitate real-time solutions. However, the public transit network experiences high levels of demand, which increases the workloads of route generation leading to higher computation time. At the same time, demand-

aware transit systems that bridge the connectivity gap in the public transit network need to be deployed in large geographical areas, which leads to high complexity. Therefore, works have explored methods to reduce the complexity of the route generation [128] [129] [130]. Request clustering is a common method used to decompose the given large problem into independent smaller subproblems in order to leverage parallel computing paradigms to speed-up computation and manage scalability. However, the solution quality can degrade drastically if request clustering is performed unwisely. This section describes existing works that study cluster-first, route-second approaches in order to address scalability issues.

Bodin and Sexton construct clusters of passengers before applying the single vehicle algorithm in [73] to each cluster [131]. Desrosiers et al. further improves the clustering based approach by creating "mini-clusters" of users. The mini-clusters are then optimally combined to form feasible vehicle routes, using a column generation technique [132].

Ioachim et al. [128] present a request clustering algorithm for a door-to-door transportation service. The work presents a hard clustering technique in combination with a heuristic routing algorithm to generate routes. Given a feasible set of itineraries, the method proposed to divide the itineraries into clusters, with each cluster containing a segment of an itinerary where the vehicle is never empty between the first pick-up point and the last drop-off point. Next, the authors create a network of clusters and solve a multiple TSP by column generation. Further, authors proposed a heuristic to reduce the size of the network without compromising the quality of the solution.

Jorgensen et al. [133] proposed a cluster-first, route-second based method to the demand-aware transit problem. Authors use a genetic algorithm to generate hard clusters which consists of a set of passengers and a vehicle. Next, independent routes are generated for each cluster using a routing heuristic.

Saez et al. [134] propose a hybrid adaptive control method for a multi vehicle, dynamic, VRP with pick-up and delivery. The work proposes a soft clustering technique

to predict the demand, which is used to dispatch vehicles. A genetic algorithm is then used in each cluster to obtain routes.

Quadrifoglio et al. [129], Shen et al. [135] evaluate centralized and de-centralized clustering strategies for a demand-aware transit service. Here, the service area is divided into clusters based on natural boundaries such as highway corridors, administrative zones and depot locations. Authors cluster based on natural boundaries to balance the number of inter-cluster trips, which results in reducing dead-head and empty vehicle miles. Further, authors proposed a soft clustering technique, which reduces the empty vehicle miles.

Pelzer et al. [136] proposed a clustering-based match-making algorithm for dynamic, demand-aware transit system that strives to maximize the utilization while limiting the vehicle detours. Authors introduce a clustering algorithm to divide the entire road network into clusters of custom size and shape. Similar to Shen et al. [135], the road network is used to identify the boundaries of each cluster. However, in this work, authors propose to scale certain clusters by a predefined factor which results in overlapping clusters. Authors claim that overlapping clusters increases the chances of matching passenger requests.

Zheng et al. [130] proposed a cluster-first, route-second approach to generate routes for a shuttle service that caters large-scale peak demands. In the proposed method, distance between two points is represented using the temporal distance instead of the actual routes. In addition, authors also introduce a request rejection mechanism and a scheme for ensuring solution feasibility. Further, authors claim that soft clustering provides better results compared to conventional hard clustering.

Chen et al. [137] propose a method to solve a demand-aware transit problem using autonomous vehicles. Authors formulate a mixed integer linear programming model to determine the ride-sharing schemes to minimize operating costs. Authors also introduce a hard clustering technique that facilitates parallel computing in order to reduce computational times in large problems.

Comert et al. [138] presented a case study of cluster-first, route-second approaches using conventional hard clustering methods for the capacitated VRP. Authors explore three hard clustering algorithms, namely k-means clustering, partitioning around medoids (PAM) and random clustering based on vehicle capacity and conclude that PAM provides better solutions.

Lowalekar et al. [139] proposed a cluster-first, route-second approach for a real-time demand-aware transit system. Authors propose to use the clustering method to reduce the computational complexity when handling large test cases. Authors explore three clustering algorithms, namely grid-based clustering, hierarchical agglomerative clustering with complete linkage (HAC_MAX) and hierarchical agglomerative clustering with mean linkage (HAC_AVG). Based on experiments, authors conclude that HAC_MAX gives better results.

Literature on scalable demand-aware transit systems reveal that request clustering approaches which enable parallel processing reduce the computational complexity of the VRP. Further, it highlights that techniques where a single road/bus stop can belong to multiple clusters outperform standard request clustering techniques where it belongs to only one cluster. Works also highlight that most existing works generate clusters based on static boundaries, which leads to low-quality solutions. Further, works do not consider the division of workloads when generating clusters leading to an increase in runtime compared to the runtime that can be achieved by parallel processing.

2.4 Demand-aware Bus Transit Systems

As defined in [24], demand-aware transit systems are characterised by the smaller capacity vehicles that operate in shared-ride mode with flexible routes and schedules. However, bridging the connectivity gap by using a large fleet of small capacity vehicles will create issues such as increased traffic congestions and greenhouse gas emissions. Thus, this section looks at works that focus on demand-aware high-capacity transit, generally termed as demand-aware buses.

Uchimura et al. proposed a door-to-door, demand-aware transit service using small buses that connects a community to/from the transit stations [140]. The proposed service follows flexible routes and serves demand originating within the service area. Authors propose a genetic algorithm for route optimization. Experimental results are presented for an instance of a single bus with 10 passengers. However, the scalability of the algorithm for multiple vehicles has not been validated.

Tsubouchi et al. proposed a demand-aware bus system that enables users the service through a cloud computing platform [141] [142]. Here, a user provides the pick-up, drop-off locations and desired pick-up/drop-off time to request the service. Authors propose a heuristic algorithm with linear time complexity for selection of vehicles and routing. However, in most cases the ride-sharing ratio is relatively low and hence, suitability of the algorithm for large instances need to be further validated.

Kawamura and Mukai proposed a model for a demand-aware bus system that caters the point-to-point transport requirements of passengers using EVs [143]. Simulation results performed with 2 vehicles and passenger requests generated every 30 minutes show that the proposed transport model reduces CO_2 emission by 80% and the running costs by 60% compared with fixed route transportation.

Uehara et al. proposed a complementary demand-aware bus system, which connects to mass transit services [144] [145]. facilitating passengers to continue the journey via the existing transit service and arrive at the destination transit stop. The proposed system transports passengers from sub-urban areas to urban areas. However, a drawback of the proposed demand-aware transit system is the requirement for advance reservation (one day prior to departure).

Martinez et al. proposed a demand-aware point-to-point minibus service in locations where high-capacity, efficient public transit options are limited [146]. Authors propose a clustering algorithm to group passengers into minibuses and show results of the application of proposed system to the Lisbon metropolitan area during morning peak hours.

Markovic et al. proposed a demand-aware bus system, which provides daily service for 450 passengers [147]. Authors analyse the trade-off between level-of-service and various system characteristics, including fleet composition, fleet size and dispatch rules. However, authors mostly consider static requests during evaluation.

Tao et al. proposed a customized bus transit system which supports request booking using the Internet, telephone or a smart phone application [148]. The authors present an analysis framework to quantify the operational performance of customized buses, public transport and personal vehicles.

Tong et al. formulate a multi-commodity network flow-based optimization model for a customized bus service network design problem [149]. Authors present a space-time network modelling method to optimize the utilization of the vehicle capacity while satisfying individual transit requests of passengers. Further, authors propose a Lagrangian decomposition algorithm to reduce the solution search space.

Archetti et al. studied a large scale demand-aware bus service, which satisfies the point-to-point transit requests of passengers [150]. The proposed service has flexible bus stops and generates routes and schedules of dynamic ride requests with a lead time of 10 minutes. Authors study the impact of lead time, size of vehicle fleet and number of passenger requests on the travel time of ride requests. Authors compare the obtained results against the travel data obtained from private vehicles and public transit using simulation studies.

Wang presented a study on a demand-aware bus system with batch demands that results from passengers arriving at a mass transit station such as a train station or bus stop at the same time [151]. The author proposes algorithms to schedule a fleet of multiple types of delivery vehicles that transport passengers to the end destinations with the objective of minimizing waiting time and riding time. Authors propose an exact mixed-integer programming model and also a meta-heuristic approach based on tabu search.

Raghunathan et al. advances the study of Wang [151] by considering multiple time-windows for arrival of passengers on the mass transit service [152]. The authors propose a clustering heuristic and a subsequent integer linear programming formulation to optimally schedule the passengers such that the total travel time (waiting time and riding time) is minimized.

Kutsuplus, provided a demand-aware transit system in Helsinki, Finland. The service matched passengers who were heading roughly in the same direction with a minibus driver, allowing passengers to share a ride that cost more than a regular city bus but less than a taxi. In order to request a ride on Kutsuplus, a user would log onto a website, top up your account, select the starting and ending points for your journey, and walk to the closest bus stop to wait for the pick-up [153] [154]. Outcome of the pilot study highlighted the lack of a mobile application and the need to make a booking for a ride more than an hour in advance as main drawbacks.

Demand-aware bus transit systems have also been deployed in large universities. In large universities, prior to regular office hours and at late night there might be sporadic transport requests which could be ideally catered using a demand-aware transit system. This concept has been further studied and practically explored by universities such as Duke [155] and Georgia Tech [156]. In the University of Duke, from 5:00pm - 6:45am, when/where Duke Transit is not in service, Duke Vans provides free, demand-aware transit for students and employees. Rides are available to and from most Duke facilities that are open during these hours. On the contrary, the Georgia tech university "Stingerette", provides demand-aware transit service for Georgia Tech students aimed to provide safe and efficient evening point-to-point transportation to locations within the main campus boundary, as well as to selected off-site Georgia Tech facilities and the Marta midtown train station.

Literature on demand-aware bus transit reveal that computation time for matching a large number of requests into the high-capacity vehicles consume a significant time. Therefore, most works require advanced reservations from passengers. At the same time, scalability of some of the proposed algorithms have not been validated using

complex scenarios. Also, most works do not consider optimizing for conflicting objectives such as passenger travel time and VMT.

2.5 Summary

In the public transit network, buses that operate with fixed schedules and routes provide connectivity to the trains. However, the connectivity gap created by additional waiting and high riding times of feeder buses have forced alternative transit solutions. As discussed in this chapter, demand-aware transit systems have gained traction as an alternative for bridging the connectivity gap. Demand-aware transit systems are characterised by a fleet of small vehicles that respond to passenger ride requests by operating with flexible routes and schedules that strive to enhance travel times of passengers. Since travel time is a combination of waiting time for a vehicle and riding time on the vehicle, if the computation time of the routes is excessively high, the waiting time increases significantly. Therefore, the routes need to be computed rapidly (real-time) such that it does not impede travel time constraints of passengers. At the same time, solutions need to be optimized to obtain high-quality routes to minimize the waiting time and riding time. In particular, noting that urban cities have a dense feeder bus system, deploying demand-aware transit system calls for scalable solutions. Feeder bus services also experience high levels of demand that has forced high-capacity demand-aware transit, commonly known as demand-aware bus transit. However, existing works on demand-aware bus transit systems fail to propose scalable techniques that generate near-optimal routes in real-time. Thus, the need to develop responsive, near-optimal solutions for large scale deployment of a demand-aware bus transit system is identified.

A demand-aware bus transit system is not only expected to improve passenger travel times but also to minimize the costs of operators. Also, it is expected to meet constraints such as maximum travel time of passengers in order to meet service standards. At the same time, regulators have imposed strict rules on the use of fossil

fuel-based vehicles. Thus, operators are forced to accommodate EVs into the public transit fleets. However, existing works on demand-aware bus transit using EVs do not account for simultaneously enhancing these conflicting objectives. Additionally, works have not considered requirements such as maximum travel time constraints and vehicle driving range limitations. In order to generate optimal routes for such conflicting objectives, the need to develop a novel exact model can be identified.

Although, exact models can generate optimal routes, literature reveal that runtime of optimally solving a model increases exponentially with the complexity of the problem. Thus, exact models can only be used to generate routes for real-time deployment in scenarios with low complexity. Therefore, relatively fast heuristic techniques, which generate near-optimal routes have been proposed to solve complex scenarios. However, existing heuristic methods do not facilitate to simultaneously optimize for passenger travel time and VMT, while considering the additional constraints. Thus, the need for developing heuristic techniques for accelerating constraint-aware flexible route generation is identified.

Heuristic techniques can generate near-optimal routes for a fleet of vehicles, significantly faster when compared to exact models, especially in complex scenarios. However, the runtime of heuristic techniques increases with problem complexity, resulting in low responsiveness. Existing works overcome this by segmenting large problems into a collection of small subproblems enabling to exploit parallel computing architectures. However, existing bus transit network segmentation techniques do not consider the directionality along the road network, and thus result in inferior solutions. Therefore, the need for developing a directionality-centric bus transit network segmentation technique is identified.

Although, bus transit network segmentation techniques can speed-up computations in relatively small geographical areas, adopting the segmentation techniques to large geographical areas result in a significant increase in runtime due to the complexity of the road network and high workloads. Therefore, large geographical areas need to be decomposed into independent, workload balanced small areas, where the proposed bus

transit network segmentation technique can be deployed in parallel to rapidly find near-optimal routes. However, existing works propose techniques for decomposing areas based on static boundaries such as administrative zones, which significantly affects the solution quality. At the same time, literature ignores balancing the workloads while generating areas, resulting in a higher runtime compared to the runtime that can be achieved by parallel processing. Therefore, the need for adopting a demand-aware bus transit system for large scale deployment is identified.

Methodologies for decomposing large geographical areas and directionality-centric bus transit network segmentation facilitate to compute near-optimal routes in real-time for a large scale demand-aware bus transit system. However, when deploying the proposed methodologies in real-life, techniques need to be devised to accurately obtain real-time and near-future demand. At the same time, it is necessary to develop techniques that enable integration of demand estimation techniques with flexible route computation techniques. Also, it is important to improve the responsiveness. However, existing works fail to propose highly responsive techniques for integration of demand estimation and route computation techniques for citywide deployment of a demand-aware bus transit system. Thus, the need to propose a framework for citywide deployment of demand-aware bus transit system is identified.

Chapter 3

Modelling Demand-aware Bus Transit System

3.1 Introduction

Public transit network topology is based on a hub & spoke model. Therefore, a typical public transit journey consists of multiple legs and is serviced by different modalities of transit such as buses, subway and trains. The feeder bus transit services are densely distributed and brings passengers from a bus stop closer to their origin, to a bus interchange which in turn connects to the rapid subway/train system. However, fixed schedules of feeder buses cause higher waiting times at bus stops. Also, due to the fixed routes of feeder buses which often have long detours, riding time is relatively high compared to direct travel time from the bus stop to the destination. This connectivity gap is repeated in a similar fashion at the destination. Therefore, passengers tend to use alternative transit solutions to bridge the connectivity gap.

Among the alternative transit solutions proposed to bridge the connectivity gap, demand-aware transit is the most widely adopted. Demand-aware transit systems need to optimally manage travel requests of passengers by a fleet of vehicles that operate with flexible schedules and routes. Although, demand-aware transit systems where

ride requests are known in advance have been studied, real-time demand-aware transit has been a reality only in the recent past. At the same time, high-capacity, demand-aware transit systems are also gaining traction. This chapter proposes a high-capacity, real-time, demand-aware transit system formally termed as demand-aware bus transit as an alternative to existing fixed route and schedule feeder bus transit services.

In a demand-aware bus transit system, passengers expect to minimize the travel time, while operators strive to minimize costs by reducing the VMT. This necessitates to compute optimal routes for enhancing these conflicting objectives. Computing optimal routes using exhaustive techniques is significantly time consuming. Thus, efficient modelling techniques that leverage algorithms such as branch and bound and branch and cut have been proposed to systematically subset the full design space and eliminate branches which do not produce superior solutions. However, existing works do not account for simultaneously enhancing the conflicting objectives. This necessitates to develop novel models, which can be used to generate optimal routes for the proposed demand-aware bus transit system.

The rest of the chapter is organized as follows. Section 3.2 presents the characteristics of bus transit systems. Section 3.3 presents the design considerations of the proposed demand-aware bus transit system. Section 3.4 presents a technique for modelling the proposed demand-aware bus transit system. Section 3.5 evaluates the model and Section 3.6 summarizes the chapter.¹

3.2 Characteristics of Bus Transit System

Public transit network is designed based on a hub and spoke model. In a hub and spoke model, the hubs provide a centralized service and the spokes supplement with additional services. In the public transit network, a transit hub is an integrated development that consist of a high speed rapid transit stop and a bus interchange. The hubs are connected by the high speed rapid transit system. The bus transit system,

¹The work in this chapter has been published in [157] and [158]

which brings passengers from the origin locations in a neighbourhood to the bus interchange (hub) and vice versa functions as the spokes. Figure 3.1 shows the hubs in the public transit network in Singapore. It shows the rapid transit system in coloured lines (interconnection between hubs) and the hubs in black circles. Two hubs, namely Boon Lay and Eunos, that are illustrated in this chapter are marked using black and purple squares respectively.

Figure 3.1 Transit Hubs of the Public Transit Network

As mentioned, a transit hub connects a bus interchange and the rapid transit stop. In a bus interchange, two types of bus services can be identified. The first type, generally termed as trunk bus services, provide point-to-point connections across long distances. The second type, termed as feeder bus services, provide transfers from hubs to the surrounding neighbourhood. In general, trunk bus services bring passengers from a transit hub to another point in a different region in the network. However, with the development of transit hubs, trunk services are constantly modified to minimize the overlap [6]. On the other hand, a feeder bus originates the service at the transit hub, travels over a route to serve passengers in a neighbourhood and finally, terminates the service back at the transit hub. Thus, feeder bus services operate as a loop service. Figure 3.2 shows selected feeder bus services (spokes) of the two transit

(a) Feeder Buses Originating at Boon Lay Bus Interchange

(b) Feeder Buses Originating at Eunos Bus Interchange

Figure 3.2 Densely Distributed Spokes of the Public Transit Network

hubs. Figure 3.2a & 3.2b shows bus stops of 4 bus services operating at the Boon Lay and Eunos bus interchange respectively. Here, the colour scheme represents the bus stops of a bus service and the transit hub is shown using a black circle. It can be seen that all buses operate on a fixed, short distance route. Thus, Figure 3.2 confirms that the same topology is repeated at each bus interchange. In addition, Figures 3.1 & 3.2 show that transit hubs are scattered in the regions and spokes are densely distributed in the neighbourhood around the transit hub.

Figure 3.3 Illustration of a Typical Public Transit Journey

In a typical public transit journey, passengers prefer to use a feeder bus service to travel to a transit hub and then connect to the rapid transit service, which is significantly faster, especially for long distance travel. Although, trunk bus services offer the comfort of less connections, it is generally slow as the buses have to share the same road network as other vehicles. Also, since the area of coverage is large, it tends to have long detours compared to the shortest path. Hence, a typical public transit journey of a passenger can be illustrated as given in Figure 3.3. The illustration shows a passenger walking from the origin to the nearest bus stop to board the feeder bus. The fixed route feeder bus brings the passenger to the bus interchange, which is connected

to the high speed rapid transit system. Then, the passenger uses the rapid transit system to travel to the bus interchange near the destination. Thereafter, a feeder bus brings the passenger closer to the destination. The passenger alights at the bus stop near the destination and finally, walks from the bus stop to reach the destination.

Figure 3.4 Travel Times for a Typical Public Transit Journey

Travel times of a typical public transit journey is illustrated in Figure 3.4. This gives an example of the schedule explorer extracted from Google Maps, for a journey to the Nanyang Technological University from Clementi area in Singapore. In the provided schedule, the first leg of the journey is completed by walking from the origin to the nearest bus stop. Then, the passenger waits for the feeder bus for 9 minutes. After boarding the bus, the passenger travels for 8 minutes to reach transit hub. Next, the rapid transit system is used to travel to the destination transit hub. Thereafter, the passenger waits for 6 minutes to board the feeder bus. The feeder bus brings the passenger closer to the destination after travelling for 11 minutes. Finally, the passenger walks from the bus stop to the destination. Figure 3.4 shows that the total travel time of the journey is 75 minutes. The red rectangles show the waiting times while the purple rectangles show the riding time on the bus. Thus, the total waiting time for the feeder bus and riding time on the feeder bus accounts for 34 minutes. Thus, it is evident that a passenger spends over 45% of the journey time, waiting for the feeder bus and riding on the feeder bus. This creates a significant connectivity gap for public transit users.

3.3 Design Considerations

Section 3.2 showed that the connectivity gap created by fixed routes and fixed schedules of the feeder buses significantly affect passengers due to the hub and spoke topology of the public transit network. Limitations such as excessive waiting times and riding times encountered during the first leg (origin to transit hub) of a typical trip are repeated similarly in the last leg (transit hub to destination) as well. At the same time, operators also incur losses with low utilization of buses due to the fixed schedules especially during off-peak hours. However, due to the similar topology of the public transit network, an effective solution to bridge the connectivity gap at a single transit hub can be replicated at each transit hub in the network. Hence, a need to develop robust solutions that can be deployed at a transit hub to reduce the travel time of passengers and at the same time increase profits of an operator is identified. Therefore, a demand-aware transit system that can be deployed by adjusting the schedules and operate with flexible routes is proposed.

Demand-aware scheduling strives to minimize excessive waiting times at bus stops. At the same time, operators can minimize costs by scheduling buses only when required. Flexible routes enable faster rides to the destination without following fixed routes, thus reducing the travel times and VMT of a vehicle. In the current transportation landscape, the preferred demand-aware transit alternatives are using SOV and point-to-point ride-hailing services such as Uber, Grab and Lyft with limited ride-sharing capacity. Even though, users can travel on flexible routes at their discretion in SOVs, it creates significant issues such as parking limitations, traffic congestions and increased greenhouse gas emissions. On the other hand, issues such as excessive waiting time due to the availability of vehicles are inherent problems of point-to-point ride-hailing services. For example, since vehicles are not specifically dedicated to serve a particular region, most of the time, passengers must wait for a long duration. In addition, point-to-point transit services are provided by low-capacity vehicles. Therefore, issues such as traffic congestion and greenhouse gas emission are also applicable for point-to-point ride-hailing services. This necessitates a high-capacity,

demand-aware transit system (demand-aware bus transit system), that serves with flexible routes to bridge the connectivity gap of public transit.

Literature confirms that majority of existing feeder bus service users travel to/from a transit hub [159]. Therefore, a demand-aware bus transit system using mini vans/buses that facilitates travelling to the transit hub from the surrounding neighbourhood and vice versa is proposed. It is not only expected to provide real-time services to passengers but also be profitable to the operator. Due to the high demands for rides to/from transit hubs, such a demand-aware bus transit system will enable the operator to optimally utilize the fleet of vehicles. In addition, the proposed system can reduce the usage of SOVs contributing towards reducing greenhouse gas emissions. Also, the proposed demand-aware bus transit system can be deployed at all transit hubs, which enables seamless connectivity in a journey.

Figure 3.5 Snapshot of the Proposed Demand-aware Bus Transit System

A snapshot of operation of the proposed demand-aware bus transit system is shown in Figure 3.5. Here, passengers to be picked-up are shown using the passenger symbol and the fleet of vehicles are shown using the bus symbol. The transit hub of the area is shown using a red square symbol. This scenario represents transporting passengers to the transit hub from the selected locality. Based on the requirements of

the system, passengers are picked-up by the allocated buses (sharing of rides), which travel on flexible routes, such that the constraints of the system are not violated. Design considerations of the demand-aware bus transit system are explained subsequently.

User Experience

User experience plays a crucial role in the widespread adoption of a demand-aware bus transit system. In this thesis, user experience is measured by the travel time of passengers for a trip. In the existing fixed route, fixed schedule bus transit system, regulators publish the operating frequency of the bus services and the expected travel time between the origin and the destination. Thus, passengers can anticipate the worst case waiting time and riding time details of a trip. A sample of the published operating frequency details of bus service 242 operating in the Jurong West region of Singapore is shown in Figure 3.6. This shows that passengers can expect a worst case waiting time of 10 minutes during off-peak operating hours.

Feeder 242	SBS Transit	Weekdays		Saturdays		Suns / P.Hs	
		1st bus	Last bus	1st bus	Last bus	1st bus	Last bus
	Boon Lay Int	0600	0040	0600	0040	0600	0040

Period	0630 - 0830	0831 - 1659	1700 - 1900	After 1900
Freq. Range	04 - 06 minutes	04 - 10 minutes	05 - 06 minutes	06 - 10 minutes

Figure 3.6 Example of Published Bus Frequencies

The proposed demand-aware bus transit system responds to the demand of passengers in contrast to operating on a fixed schedule. In addition, the demand-aware bus transit system operates with flexible routes. Therefore, a passenger is unable to anticipate the alighting time at the destination. For example, a passenger who has already boarded the vehicle may expect longer detours to pick-up other passengers which in some instances may cause additional riding time compared to the riding time of the existing fixed route transit service. Hence, it is imperative that a maximum travel time (addition of waiting time and riding time) constraint is imposed in order to ensure the user experience of the proposed demand-aware system. Thus, a maximum travel

time constraint is enforced for each request based on the location of the passenger. The next subsection explains the rationale of selecting a location based maximum travel time constraint and the method of computing the maximum travel time for a request.

Travel Time

The demand of the system originates from passengers requesting for rides. It is assumed that passengers use a mobile application to provide the required details. A passenger indicates the pick-up location (origin), drop-off location (destination) and the number of required seats in a request. Then, the request is tagged with the maximum travel time constraint. Travel time combines both the waiting time for the vehicle to reach the origin and the riding time on the vehicle to pick-up other passengers along the route to the destination. The maximum travel time indicates the worst case time a passenger has to spend prior to reaching the destination.

The benefits of imposing a maximum travel time constraint is twofold. Firstly, it ensures that passengers do not travel excessively in a vehicle when multiple rides are shared, which will reduce the user experience of the demand-aware bus transit system. Secondly, it improves the reliability of the system as passengers are informed of the worst case arrival time at the destination. The operator needs to ensure that the demand-aware transit system does not violate the travel time constraint. Maximum travel time is computed using Equation 3.1. It is equal to the tolerance factor multiplied by the travel time between the origin and destination (O-D) in a SOV under prevailing traffic conditions. The value of the tolerance factor is set by the operator such that travel time on the demand-aware bus transit system is always less than the existing bus transit system. Thus, tolerance factor is the multiplicative factor of the direct travel time between the O-D, which governs the time a passenger has to travel prior to reaching the destination. A low value for the tolerance factor implies that the passengers can reach the destination much quicker, improving the user experience. For example, a tolerance factor of 2 implies that the demand-aware bus transit system will be close to a SOV. However, as the demand-aware bus transit system facilitates

ride-sharing, detours are required from the direct path between the O-D. Therefore, typically a higher tolerance factor is required to accompany high-capacity ride-sharing.

$$\text{Maximum Travel Time} = \text{Tolerance Factor} * \text{Travel Time between O-D in SOV}; \quad (3.1)$$

It should be noted that the maximum travel time constraint can also be imposed as a constant value. For example, each passenger can be guaranteed to reach the destination within 30 minutes of the request time. However, such a system will not be beneficial for passengers closer to the destination as the detour time of passengers can be significantly high. In contrast, passengers with an origin far from the destination might not have sufficient time to reach the destination. Thus, such requests will always be rejected by the system. Hence, the proposed location-based maximum travel time constraint is justified for both passengers with a location closer to the destination as well farther away from the destination.

Operator Profitability

Operating a demand-aware bus transit system should not only benefit passengers by improving the user experience but also the operator by increasing the profit margins. From the viewpoint of the operator, profits can be increased by reducing the VMT, which in turn aids to reduce the fuel costs. This can be achieved by making intelligent scheduling and routing decisions. For example, in a fixed schedule transit system, the operator does not have an alternative to sending a bus even if the demand is significantly low. In contrast, in a demand-aware bus transit system, since the schedules are flexible, the operator can schedule buses when the utilization of a bus reaches a predefined value. This can contribute to reducing unnecessary buses with low utilization. For example, it may be possible to schedule a lower number of demand-aware buses to pick-up passengers in comparison to a higher number for the same fixed schedule transit system. Therefore, the VMT of the fleet will be reduced. Also, since the routes are flexible, the operator can select the routes which minimize the VMT instead of

traversing a fixed route. Both options add to increase the profits for an operator by reducing fuel costs.

At the same time, transportation was the second biggest source of greenhouse gas emission in the United States in 2014 accounting for 26% of the total emissions [160]. Similarly, according to the European environmental agency, road transportation accounts for 72% of greenhouse gas emissions from the transportation sector in the European Union in 2018 [161]. As a result, driving bans for diesel-powered vehicles in major cities [162] and nationwide bans for petrol and diesel vehicles [163] have been imposed by regulators. This has enforced transit operators to include zero emission EVs to the existing bus fleets [164] [165]. Thus, it is assumed that the fleet of the demand-aware bus transit system consists of EVs. However, it should be noted that this will introduce additional constraints to an operator. For example, EVs have a limited driving range even with a fully charged battery and also the charging time for high-capacity batteries is significantly high. Therefore, operators are entrusted with the added complexity of optimally managing a fleet of EV. Thus, improving the VMT of a fleet of EV not only reduces the charging cost of the operator but also the availability of EVs for deployment due to the high charging times.

Vehicle Capacity

Capacity of the fleet of vehicles of a demand-aware bus transit system is a key factor that decides the profit margins for an operator. For example, if the fleet consists of large capacity buses (e.g. in excess of 80 seats) the vehicles may be plying empty during off-peak hours. On the other hand, if the vehicle size is very small (e.g. 4 seater cars) the number of vehicles required will be significantly high, increasing the VMT. Therefore, selection of vehicle capacity is an important consideration of the fleet which should balance utilization and realizability.

Existing bus transit systems use large capacity single decker (80 - 88 passengers) or double decker (120 - 125 passengers) buses. Even though, the utilization of large

buses during peak periods are high, these buses are often underutilized during off-peak periods. Hence, from the point of view of the operator, it is costly to operate large capacity buses during off-peak periods. In contrast, low-capacity ride-hailing systems use passenger cars (3 - 4 passengers) to provide real-time demand-aware transit services. Since shared passenger cars are the closest alternative to SOVs, the user experience of demand-aware transit is significantly boosted by using such vehicles. However, replacing a single/double decker bus with a large number of low-capacity passenger cars can create traffic congestion and also increase the VMT. Thus, it is proposed to use mid capacity mini vans (8 - 16 passengers) or mini buses (18 - 22 passengers) for the proposed demand-aware bus transit system. This not only aids to increase the utilization of vehicles during off-peak periods but also minimizes the impact of a large fleet of small capacity vehicles during peak periods. However, it should be noted that the labour costs associated with the variation of the number bus drivers is not considered in the analysis.

Driving Range

The driving range of an EV is determined by the capacity of the battery and the availability of charging infrastructure. Fully charging a large capacity battery consumes a significant amount of time [166]. Hence, the operator needs to ensure that the fleet is managed properly to increase the availability of EVs for deployment. This turns out to be an added complexity in addition to managing the demand-aware transit system. Thus, from the viewpoint of the operator, it will be preferable to have a system where the added complexity arising as a result of "range anxiety" is minimized. To this end, it is assumed that using small/mid capacity batteries will aid fast charging of EVs. Fast charging provides relatively quick (10 - 30 minutes) charging of small capacity batteries [166]. Even though, a smaller battery capacity will reduce the range of the EV, considering real-life deployment of the proposed demand-aware bus transit system, this issue can be mitigated by appropriately taking the driving range constraint into account. Hence, it is assumed that fast charging is more beneficial for the proposed

demand-aware bus transit system. Another approach that can be used instead of fast charging is battery swapping. However, having a pool of large capacity extra batteries for battery swapping will incur a significant cost to an operator which will impact profits. Thus, battery swapping is not considered in this thesis. However, it should be noted that, the work assumes that the infrastructure required for fast charging of EVs is existent.

Deployment

It is assumed that the proposed demand-aware bus transit system consists of a fleet of EVs (supply) dispersed in the selected area surrounding a transit hub. EV can be dispersed based on the population density and the demand for public transit in the service areas. Also, it is assumed that EVs will be parked in the nearby car parks which are located adjoining major bus stops in the selected area. Car parks are assumed to be equipped with fast charging infrastructure. The system responds to passenger requests (demand) in real-time by picking passengers from the bus stops and dropping passengers at a predetermined transit hub. Once a trip completes, the EVs will be fully charged at the transit hub and then redeployed to move passengers from the transit hub to destination bus stops. Similarly, EVs will be charged at car parks when the assigned trips are completed. However, the models presented in this thesis are based on the case of transporting passengers to the transit hub. The presented models are extensible and can be adapted for the reverse trip with minimal effort. Concepts such as zone level boarding can be used to reduce the complexity. The following assumptions are considered for model generation. (1) all EV have a fixed capacity and driving range (homogeneous), (2) driving range is compared with the road network distance, (3) capacity is measured by the empty seats, (4) passengers request rides using a mobile application indicating the pick-up, drop-off location and number of seats, (5) each request has a travel time constraint to guarantee the user experience and (6) charging of EVs can be done only after the assigned trips are complete.

Typically, EV driving range is dependent on the loading and climate conditions. However, for simplicity these factors are ignored, and it is assumed that driving range can be directly compared against the VMT on the road network. Even though, the mobile application can automatically detect the origin of the ride-request using GPS signals, it is assumed that allowing passengers to select the pick-up location will be more user-friendly. Further, a user may want to request a ride from a different location compared to the physical location. Charging times of EVs are relatively high. Thus, when assigning trips to an EV, the operator has to ensure that it can reach the destination with the remaining driving range since it is unable to re-charge in the middle of a trip without violating the user experience.

In the background, flexible scheduling decisions will be evaluated using a hybrid of immediate and periodic scheduling mechanisms. By default, the EVs are scheduled when the demand reaches a specific utilization ratio of the EV capacity (in the experiments 80% utilization is considered). However, when the demand is significantly low, a periodic scheduling will be done to prevent significantly high waiting times. Irrespective of the scheduling scenario, all requests which fall in the selected planning horizon (between two successive scheduling instances) are considered as demand and all empty and fully charged EVs are used as the supply of the system. The routing algorithm strives to allocate the passengers to the fleet of EVs while striving to enhance the objectives. After the allocation, passengers will be notified of the estimated time of arrival (ETA) of the EV at the bus stop.

The objective of any passenger using the proposed demand-aware bus transit system is to reach the destination as quickly as possible. Thus, in order to enhance the user experience of the system, the total passenger travel time has to be minimized. On the other hand, the operator wants to minimize the costs of operation. Thus, operator profitability is enhanced by minimizing the VMT of the fleet (fleet travel distance).

3.4 Model Generation

The demand-aware routing problem studied in the context of public transit can be classified into the vehicle routing problem category. Based on the existing exact solution techniques, the problem is initially converted to a graph-based representation. In a graph-based representation, two different modelling approaches have been proposed. In the first approach, integer variables are associated with each arc or edge of the graph, which counts the number of times the arc or edge is traversed by a vehicle. It is known as vehicle flow formulations [167]. The second approach has an exponential number of binary variables each associated with a different feasible route. Then the vehicle routing problem is formulated as a set partitioning problem. Although, the linear program relaxations of the first type of models can be weak, it is the most widely used type of model [168].

The vehicle flow formulations are further classified into the two-index or the three-index models. The two-index model, such as the work proposed by Lu and Dessouky [169], contains a polynomial number of constraints but relies on extra variables to impose pairing and precedence constraints. On the other hand, the three-index model, such as the work by Cordeau [76] supports a polynomial number of constraints and uses several classes of valid inequalities that are adapted from similar VRPs. Also, inequalities facilitate to take advantage of the structure of the problem. Thus, the proposed model is based on a three-index formulation and adapts the structures from the work of Chevrier et al. [170]. However, compared to the work of Chevrier et al. the proposed model contains additional constraints such as individual travel time constraints and simultaneously optimizing for multiple objectives.

In addition, to ensure that the model is more comprehensive and realistic, the cost matrices used in the models are assumed to be asymmetric. Thus, travelling from point A to point B is not considered as the same as the opposite trip. This is a typical situation encountered in real-life road network. Further, the real distance on the road network is considered in the models as opposed to Euclidean distance considered in

some models [171] [172] [150]. Also, the models inherit the principle of triangle inequality. Thus, the direct travel distance/time from point A to point B is lower compared to travelling through point C to reach point B. Also, the proposed model uses arc elimination rules based on the structure of the problem and tighter bounds on variables. The model is described in detail in the subsequent paragraphs.

Graph-based Representation

The given problem is converted to a mathematical representation using a graph data structure. The problem is defined using a directed and acyclic convergent graph G . The graph G consists of nodes to represent the fleet of m EV denoted by $\{V_1, V_2, V_3, \dots, V_m\}$, set of n passengers denoted by $\{P_1, P_2, P_3, \dots, P_n\}$, and a node to represent the transit hub of the service. Hence, G consists of $(m + n + 1)$ nodes, where nodes $1, 2, 3, \dots, m$ refer to the fleet of EV, nodes $m + 1, m + 2, m + 3, \dots, m + n$ refer to the set of passengers and node $m + n + 1$ refers to the transit hub. The terminology is summarized in Table 3.1. An example graph consisting of 3 EV ($m = 3$), 6 passengers ($n = 6$) and the transit hub is given in Figure 3.7.

Table 3.1 Terminology of Passengers and EV

Term	Description
m	number of EV
n	number of passengers
V_v	v^{th} EV in the fleet of m EV
P_p	p^{th} passenger in the set of n passengers

Figure 3.7 Graph Representation of Demand-aware Flexible Routing

The set of all nodes, $1, 2, 3, \dots, m+n+1$ is denoted by Φ . The subset of nodes $1, 2, 3, \dots, m$, representing EV is denoted by v and the subset $m+1, m+2, m+3, \dots, m+n$, representing passengers is denoted by ρ . The transit hub represented by node $m+n+1$ is denoted by ϑ . All the termination nodes, where a valid edge can terminate is denoted by Ψ . Table 3.2 summarises the terminology of the graph.

Table 3.2 Terminology of Nodes of the Graph

Term	Description	Nodes represented in graph
Φ	set of all nodes	$1, 2, 3, \dots, m+n+1$
v	subset of EV	$1, 2, 3, \dots, m$
ρ	subset of passengers	$m+1, m+2, m+3, \dots, m+n$
ϑ	transit hub	$m+n+1$
Ψ	termination nodes	$\Phi \setminus v$

The set of edges, represents all the direct connections between the fleet of EVs, the set of passengers and the transit hub. There are no edges ending at node set v and beginning from node ϑ . Further, each edge $\langle i, j \rangle$ has an associated weight $\langle t_{i,j}, d_{i,j} \rangle$, which denotes the shortest path travel time and the shortest path distance

between the two nodes $i \rightarrow j$ respectively. Time and distance matrices ($t_{i,j}, d_{i,j}$) are obtained using the Open Source Routing Machine (OSRM) [173]. Here, OSRM Route Service API is used to obtain the distance and time of the shortest path route by providing the GPS coordinates of the passenger locations, EV locations and destination. The details are summarized in Table 3.3.

Table 3.3 Terminology of Edges of the Graph

Term	Description
i	node i
$\langle i, j \rangle$	edge from node $i \rightarrow j$
d_{ij}	travel distance from node i to j
t_{ij}	travel time from node i to j

The constraints of the model consist of EV constraints, (i) seating capacity (l), (ii) driving range (r); and passenger constraints, (i) maximum travel time of a passenger (u_{p_p}). Also, the time till the scheduling instance of each passenger request is termed w_{p_p} . These are summarized in Table 3.4.

Table 3.4 Constraints of the Model

Term	Description
l	maximum capacity of an EV
r	maximum range of an EV
u_{p_p}	maximum travel time of p^{th} passenger
w_{p_p}	time till scheduling instance of p^{th} passenger

Next, the mathematical formulations for the two conflicting objectives of the problem, namely minimizing passenger travel time and VMT is presented. However, prior to that formal concepts of single-objective and dual-objective combinatorial optimization problems are introduced.

Single-Objective Combinatorial Optimization

Single-objective combinatorial optimization is a problem in which certain parameters (design variables) need to be determined to achieve the best measurable performance (objective function) under given constraints [174]. The constraints of the problem can be of different forms such as requirements and physical limitations. In general, formulating optimization problems involve many aspects such as problem formulation, solution generation and result interpretation. The general form of a single-objective combinatorial optimization problem is given in Equation 3.2 - 3.5.

$$\text{minimize } f(x) \quad \text{subject to;} \quad (3.2)$$

$$g_i(x) \leq 0, \quad \forall \quad i = 1, \dots, m \quad (3.3)$$

$$h_j(x) = 0, \quad \forall \quad j = 1, \dots, p \quad (3.4)$$

$$x_k^l \leq x_k \leq x_k^u; \quad \forall \quad k = 1, \dots, n \quad (3.5)$$

Equation 3.2 shows the objective function or the goal to be minimized or maximized. All constraints in the problem are given in Equation 3.3 - 3.5. The i^{th} inequality constraint is given in $g_i(x)$ in Equation 3.3. There are m different inequality constraints which need to be satisfied in the final solution. Similarly, j^{th} equality constraint is given in $h_j(x)$ in Equation 3.4. There can be p different equality constraints. x is the vector of design variables given in $x = [x_1, x_2, \dots, x_n]^T$ with n design variables. For each design variable k , the lower and upper bounds are given by x_k^l and x_k^u respectively. This is given in Equation 3.5. Also, it should be noted that a general constrained optimization problem can have both equality and inequality constraints or either.

Dual-Objective Combinatorial Optimization

A general dual-objective combinatorial optimization problem is defined as given in Equation 3.6, where X is the set of feasible solutions. The evaluation vectors are denoted in Equation 3.7 and Equation 3.8.

$$\min f(x) = (f_1(x), f_2(x)) \quad \text{subject to } x \in X; \quad (3.6)$$

$$z = f(x); \quad (3.7)$$

$$z_i = f_i(x); \quad \forall i = 1, 2 \quad (3.8)$$

The objective space is defined by $Z = \{z = (z_1, z_2) : z_i = f_i(x), \forall x \in X, i = 1, 2\}$. However, there is no solution that optimizes both objectives simultaneously. Therefore, a solution will be an acceptable solution only if there is no strictly better solution for a single objective, even though the other objective is weak. This concept is defined by the dominance relation. This is used to characterize pareto efficiency that is used to replace the concept of optimal solution in a single-objective optimization problem. Next, these concepts are formally defined as given in Bérubé et al. [175].

Dominance relation

Let z and $z' \in Z$. z dominates z' ($z \succ z'$) if and only if $z_1 \leq (z_1)'$ and $z_2 \leq (z_2)'$ where at least one inequality is strict.

Pareto efficiency

A solution $x \in X$ is efficient in X , if and only if $\forall x' \in X$ such that $f(x') \succ f(x)$.

Efficient set

The efficient set $\Xi = \{x \in X : x \text{ is Pareto efficient in } X\}$

Pareto front

The Pareto front $\Delta = f(x) : x \in \Xi$

The efficient set (Ξ) and Pareto front (Δ) contain all the pareto efficient solutions and all the non-dominant points in the objective space respectively. Thus, the Pareto front is used as the typical representation of the objective space of a MOCO problem.

3.4.1 Mathematical Formulation

The mathematical formulation for the two design objectives is presented here. It is modelled using an optimal mixed-integer quadratically constrained programming (MIQCP) mathematical formulation. CPLEX Optimization Programming Language (OPL) is used to model the problem and the in-built MIP solver to obtain the optimal solution. All the models are coded on *IBM ILOG CPLEX Optimization Studio 12.7.1* [176]. The decision variables of the model are presented first followed by the objective functions for the single-objective formulation to minimize passenger travel time and VMT. Next, the constraints of the model are given. Finally, the dual-objective formulation is derived.

Decision Variables

The model consists of two decision variables denoted by x and s . The binary decision variable x_{ijv} is defined for each edge $\langle i, j \rangle$ and each EV V_v , where $i \neq j$, $i \neq \varnothing$ & $j \neq v$; denotes whether EV V_v travels along the edge $\langle i, j \rangle$ from node i to j . Similarly, the decision variable s_{iv} is defined for each node i and each EV V_v , where $i \neq v$; denotes the service time of node i by EV V_v . The two decision variables are presented in Equation 3.9 and Equation 3.10 respectively.

$$x_{ijv} = \begin{cases} 1, & \text{EV } V_v \text{ travels from node } i \text{ to node } j, \\ 0, & \text{otherwise.} \end{cases} \quad (3.9)$$

$$s_{iv} = \begin{cases} f, & \text{EV } V_v \text{ services node } i, f \in \mathbb{Z}^+, \\ 0, & \text{otherwise.} \end{cases} \quad (3.10)$$

Objective Functions

The objective functions of the formulations are given in Equation 3.11 and 3.12 which minimize the total travel time of passengers and the VMT respectively. The two objective functions reflect the user experience and operator profitability respectively.

Minimizing Passenger Travel Time

$$\text{minimize } \sum_{i \in \rho} \sum_{j \in \Phi} \sum_{k \in \nu} (x_{ijk} * s_{\partial k}); \quad (3.11)$$

The objective function in Equation 3.11 minimizes the travel time of passengers. However, it should be noted that the initial waiting time (w_{P_i}) of each passenger that depends on the scheduling instance should be added to the outcome of the routes to compute the final travel time of passengers.

Minimizing VMT

$$\text{minimize } \sum_{i \in \Phi} \sum_{j \in \Phi} \sum_{k \in \nu} x_{ijk} * d_{ij}; \quad (3.12)$$

The objective function in Equation 3.12 minimizes the travel distance between the edges in the graph, which in turn reduces the total VMT of the fleet.

Constraints

Following assumptions are considered on the constraints of the formulation. (1) each passenger is serviced by exactly one EV; (2) all routes start at node set ν and end

service at the destination node, i.e. node $m + n + 1$; (3) travel time of each passenger does not violate the maximum travel time constraint; (4) capacity of each EV is not exceeded; and (5) driving range of the each EV is not exceeded. All the constraints in the model are presented in Equation 3.13 - Equation 3.27. Constraints are classified into 4 categories, namely routing, timing, side and subtour elimination. Routing constraints model the flow of vehicles in the graph. Timing constraints model the travel time between two nodes in the graph. Side constraints model the EV capacity, driving range and maximum travel time of a passenger. Finally, subtour elimination constraints tighten the model by preventing direct travel between origin and destination, loops at a single node and multiple EV visiting a node.

* *Routing Constraints*

$$\sum_{j \in \rho} x_{ijk} = 0 \quad \forall i \in \nu, \forall k \in \nu; \text{ where } i \neq k; \quad (3.13)$$

$$\sum_{j \in \rho} x_{kjk} \leq 1 \quad \forall k \in \nu; \quad (3.14)$$

$$x_{\varnothing jk} = 0 \quad \forall j \in \Phi, \forall k \in \nu; \quad (3.15)$$

$$x_{ikk} = 0 \quad \forall i \in \rho, \forall k \in \nu; \quad (3.16)$$

$$\sum_{i \in \Phi} x_{ibk} - \sum_{j \in \Psi} x_{bjk} = 0 \quad \forall k \in \nu, \forall b \in \rho; \quad (3.17)$$

$$\sum_{i \in \rho} x_{i\varnothing k} \leq 1 \quad \forall k \in \nu; \quad (3.18)$$

$$\sum_{i \in \rho} \sum_{k \in \nu} x_{i\varnothing k} \geq 1 \quad (3.19)$$

* *Timing Constraints*

$$s_{kk} = 0 \quad \forall k \in \nu; \quad (3.20)$$

$$x_{ijk}(s_{ik} + t_{ij} - s_{jk}) \leq 0 \quad \forall i \in (\Phi \setminus \varnothing), \forall j \in \Phi, \forall k \in \nu; \quad (3.21)$$

* *Side Constraints*

$$\sum_{i \in \rho} \sum_{j \in \Psi} x_{ijk} \leq l \quad \forall k \in v; \quad (3.22)$$

$$\sum_{i \in \Phi} \sum_{j \in \Phi} x_{ijk} * d_{ij} \leq r \quad \forall k \in v; \quad (3.23)$$

$$s_{\partial k} * x_{ijk} + w_{P_i} \leq u_{P_i} \quad \forall i \in \rho, \forall j \in \Psi, \forall k \in v; \quad (3.24)$$

* *Subtour Elimination Constraints*

$$x_{i\partial k} = 0 \quad \forall i \in v, \forall k \in v; \quad (3.25)$$

$$x_{iik} = 0 \quad \forall i \in \rho, \forall k \in v; \quad (3.26)$$

$$\sum_{j \in \Psi} \sum_{k \in v} x_{ijk} = 1 \quad \forall i \in \rho; \quad (3.27)$$

The first routing constraint in Equation 3.13 sets the origin of the routes to each EV location. This implies that a valid route of an EV starts from the node representing the EV. Equation 3.14 ensures that at least one EV moves from its' origin to serve passenger requests. Equation 3.15 and 3.16 prevent routes from originating at the transit hub and finishing at EV nodes respectively. Equation 3.17 ensures that a route does not end at a passenger node. In other words, it implies that an EV visiting a passenger node must leave either to another passenger node or the destination. The final two routing constraints deal with the completion of the route. Equation 3.18 ensures that an EV will reach the destination only through a passenger node. Similarly, Equation 3.19 ensures that at least one EV reaches the destination. Thus, in the best case, a single EV serves passenger requests and reaches the destination.

Timing constraints in Equation 3.20 and 3.21 initialize the service time of each EV and models the travel time between nodes respectively. The service time of each EV is initialized to 0 for each scheduling instance. Thus, all EV have a start time of 0 as set in Equation 3.20. The travel time between nodes is modelled in Equation 3.21. However, due to the multiplicative factor this constraint is non-linear. Thus, the big M method proposed in [177] is used to linearise the constraint. The linearised constraint

is given in Equation 3.28.

$$\begin{aligned} s_{ik} + t_{ij} - s_{jk} &\leq M_{ij}(1 - x_{ijk}) \quad \forall i \in (\Phi \setminus m + n + 1), \\ &\quad \forall j \in \Phi, \forall k \in v; \end{aligned} \quad (3.28)$$

Side constraints, namely EV capacity, driving range and maximum travel time of passengers are modelled in Equations 3.22, 3.23 and 3.24 respectively. For the proposed demand-aware system, empty EV are considered as the supply. Thus, in Equation 3.22, the capacity of the EV is directly compared with the number of passengers picked up. Also, it should be noted that, the range of the EV is directly compared against the VMT assuming that loading and other criteria does not affect the range. Thus, in Equation 3.23 the summation of the VMT is compared with the maximum allowable range. In Equation. 3.24, the summation of the waiting time of each passenger till the scheduling instance and the time the EV which picked-up the passenger reaches the destination is compared with the maximum travel time constraint of each passenger.

Subtour elimination constraints further tighten the model and prevent any loops or subtours. Subtours are multiple cycles formulated as paths, instead of one complete path from origin to destination. Equation 3.25 prevents the EVs travelling directly from origin to the transit node. This implies that if an EV is not assigned passengers it will remain in the current location. Equation 3.26 prevents loops in the route. Finally, Equation 3.27 ensures that there is only one outgoing edge at a passenger node. This ensures that multiple EVs do not visit a passenger node.

The objective functions in Equation 3.11 and Equation 3.12 can be solved individually with the constraints given in Equation 3.13 - Equation 3.27 to obtain the single-objective optimal results for the two objectives. Formally, these points are termed as the utopia points (Z^U) of the objective. The solutions provide the optimal set of routes for the fleet of EVs that can individually optimize each objective. However,

since the two objectives in the formulations are conflicting, the normalized weighted technique is used to generate Pareto points for minimizing both the objectives.

Minimizing Passenger Travel Time & VMT

Normalized weighted technique is derived from the weighted sum method for multi objective optimization. The technique allows the operator to fine tune the transportation system based on user preference. Thus, the operator can select the weights prior to running the optimization algorithm and it allows the algorithm to converge to a single point that reflects the selected user preference [178].

$$\text{minimize } \sum_{i=1}^n w_i * F_i(x) / z_i^U; \quad (3.29)$$

where;

$$\sum_{i=1}^n w_i = 1; \quad (3.30)$$

Equation 3.29 shows the normalized weighted sum method. Here, $F_i(x)$ defines the i^{th} objective function, w_i defines the weight of the i^{th} objective and z_i^U is the utopia point of the i^{th} objective. Equation 3.30 indicates that the sum of assigned weights to each objective should be equal to 1. Thus, in the case of the proposed formulation the two objectives given in Equation 3.11 and Equation 3.12 will be reflected as $F_1(x)$ and $F_2(x)$ along with all the constraints given in Equation 3.13 - Equation 3.27. Changing the assigned values of w_i results in a different solution, which will produce a design point in the efficient set. Multiple solutions from the above formulation provides the pareto optimal front for minimizing both passenger travel time and VMT. Each solution in the Pareto front gives a set of routes for the fleet of EVs.

3.5 Performance Evaluation

In this section, the performance is evaluated of the demand-aware routing model. The experimental setup is presented first, followed by results. The model is validated using an exhaustive search for several test cases. Next, the impact of vehicle capacity and driving range on optimizing passenger travel time and VMT are presented. Thereafter, the Pareto front of the scenarios are analysed. Finally, the runtime variation with the complexity of the proposed model is studied. The runtime is measured on a PC with 16 GB RAM, running Windows 7 on an Intel Xeon E5-1650V2 CPU at 3.50 GHz.

3.5.1 Experimental Setup

Geographical Area for Experiments

In order to generate test cases and validate the techniques presented in this chapter, the zone encompassing Nanyang Technological University (NTU), Singapore shown in Figure 3.5 is used. Generally, population density in a University is relatively high and the penetration of public transit is comparatively low. Thus, universities often provide fixed route shuttle services to the nearest rapid transit node. Hence, the selected zone provides an ideal locality to experiment the demand-aware bus transit system presented in this thesis.

Experimental Data Generation

Data for all the experiments are generated using the real road network encompassing NTU. In the selected zone, 30 existing bus stop locations have been selected. Public transit demand data from Land Transport Authority (LTA) is used to quantify demand. EVs are assumed to be distributed in the car parks in the selected zone. In each experiment, locations of demand (passengers) and supply (EVs) are selected randomly. To analyse the benefits of flexible routing, the demand is assumed to be originating at

the same time hence, impact of scheduling instance is not applicable. Further, transit data from OSRM is used to obtain the travel time and distance.

3.5.2 Experiments

Validation

In order to validate the model, all the possible routes of several small test cases are manually computed (exhaustive search) and the best solution is compared with the result obtained from solving the model using CPLEX. The results confirmed that the proposed model generates the optimal routes for the given scenario. Once the model is verified using this technique, for the remaining experiments, the result obtained from CPLEX is considered as the optimal route. It should be noted that the ‘branch and bound’ algorithm used to solve the model can significantly speed-up the computations by pruning the design space. The pre-solve step eliminates a significant number of infeasible branches and thereby reduces the runtime compared to an exhaustive search. This is evident by the runtime results shown later in this chapter.

Vehicle Capacity

For this experiment, a scenario with 40 passengers distributed over 30 bus stops is selected. Therefore, some bus stops have multiple passengers while some can be empty. A fleet of 5 vehicles distributed in the vicinity is analysed by varying the capacity of a vehicle. Tolerance factor is set to 4 and the driving range is set to 30 km in each experiment. These values are set such that the impact of the variables is minimal for the experiment. Each experiment is repeated 3 times and the output of optimizing passenger travel time and VMT is analysed. The results are shown in Figure 3.8 and Figure 3.10 respectively.

Passenger Travel Time Optimization

Figure 3.8 Impact of Vehicle Capacity on Passenger Travel Time Optimization

Figure 3.8 shows the results for optimizing passenger travel time. In Figure 3.8, horizontal axis shows the capacity of a vehicle in the fleet, while the left vertical axis shows the average passenger travel (waiting + riding) time and right vertical axis shows the number of vehicles used in the final routes. The green column shows the passenger travel time while the orange diamond symbol shows the number of vehicles. Results indicate that average passenger travel time can be reduced by using higher capacity vehicles. Higher capacity vehicles facilitate more passengers to be allocated to a vehicle along the path to the destination. Therefore, passenger waiting time will be reduced. This is indicated by the breakdown of travel time results for 8 and 14 seat capacity vehicles given in Figure 3.9. The two columns show waiting time and riding time for the scenario of 8 and 14 seat vehicles. The vertical axis gives the average time. This indicates that waiting time is reduced as the capacity is increased. On the other hand, the riding time of passengers has increased. This is caused by extra detours along the path of the vehicle to pick-up additional passengers. However, it should be noted that all 5 vehicles are used to fulfil the demand in all the cases. This implies that in order to minimize passenger travel time, it is beneficial to use a higher number of vehicles in the fleet even though the demand can be satisfied with a lower number of vehicles. Consequently, the utilization of the fleet is low.

Figure 3.9 Waiting & Riding Time Variation with Changing Capacity (for Passenger Travel Time Optimization)

VMT Optimization

Figure 3.10 shows the results for optimizing VMT of the fleet. In Figure 3.10, horizontal axis shows the capacity of a vehicle, while the left vertical axis shows the distance and right vertical axis shows the number of vehicles. The blue column shows the average VMT while the orange diamond symbol shows the number of vehicles. Similar to the previous case, the VMT is reduced by using higher capacity vehicles. However, in contrast to the previous case, the number of vehicles used in each experiment has reduced with increasing capacity. The number of vehicles used in each experiment is 5,4,4 and 3 respectively. It implies that the model has minimized the number of vehicles in order to reduce the VMT.

Figure 3.10 Impact of Vehicle Capacity on VMT Optimization

Figure 3.11 Waiting & Riding Time Variation with Changing Capacity (for VMT Optimization)

It should be noted that when the capacity of the vehicles is increased more passengers will be allocated to a single vehicle. Therefore, the number of used vehicles are reduced. Hence, when optimizing VMT, passenger travel time will increase with higher capacity vehicles. This is explained using Figure 3.11. The two columns show waiting time and riding time for the scenario of 8 and 14 seat vehicles. The vertical axis gives the average time. It is observed that the waiting time and riding time of passengers have increased in this case with higher capacity vehicles. This is attributed to the lower number of vehicles used in the case of capacity 14 (3 vehicles) compared to capacity 8 (5 vehicles). This necessitates the use of a maximum travel time constraint to ensure that passengers are always guaranteed to reach the destination without excessive travel time. Although, the selected tolerance factor does not have an impact in this case for optimization, if it is set at a very low value (e.g. 2) this limits the number of passengers that can be allocated to a single vehicle. Hence, more vehicles will be required to serve the demand which limits the opportunity to reduce the VMT.

Driving Range

(a) Passenger Travel Time Optimization

(b) VMT Optimization

Figure 3.12 Impact of EV Driving Range

A similar setup with 40 passengers distributed in 30 bus stops are considered for analysing the impact of EV driving range. Each experiment is repeated 3 times. A fleet of 5 EV with a capacity of 12 each, distributed in the vicinity is analysed by varying the driving range. Results for optimizing for passenger travel time and VMT are given in Figure 3.12a and Figure 3.12b respectively. The horizontal axis in the graph shows the 3 driving range constraints and the vertical axis shows the average passenger travel time and VMT respectively. In both cases, as the driving range constraint is relaxed the objective of the problem is further minimized. This is because the EV can detour a larger distance when the driving range is high. Therefore, when detour distance is limited, another EV that is far away from the passenger origin has to visit

to pick-up the passenger. This increases respective optimization objective. In the case of passenger travel time optimization, a 10% improvement in the objective is observed when the driving range is relaxed from 8 km to 30 km. The same improvement for VMT optimization is 19%. The number of EVs used in the experiments are 5 and 4 respectively for Figure 3.12a and Figure 3.12b. Similar to the case of varying EV capacity, this implies that VMT optimization results in a higher EV utilization. However, the number of EVs do not vary across the different test cases in this scenario.

Optimizing Passenger Travel Time & VMT

In the above experiments, only a single objective (either passenger travel time or VMT) is minimized without considering the impact for the second objective. In the experiments given in this section, the weighted sum method is used to obtain the Pareto front, which reflects simultaneous optimization of both objectives, namely passenger travel time and VMT. The utopia points of the Pareto front reflect the scenarios that optimize a single objective. Figure 3.13 shows the Pareto front in the objective space for the experiments. Here, the horizontal axis shows the VMT while the vertical axis shows the total passenger travel time.

Initially, the utopia points for the two objectives are obtained by setting the weights equal to 0 and 1 in Equation 3.29. The values obtained for the utopia points are then used in the normalized weighted technique. In the experiments, passengers and EV are distributed randomly in the selected bus stops and car parks. The number of passengers and EV in each test case is given in Figure 3.13. EV capacity and driving range is fixed at 12 and 30 km respectively. Tolerance factor is set to 4. Solutions corresponding to weights of 0.25, 0.5 and 0.75 in the normalized weighted technique is obtained. Results show that varying the weights of Equation 3.29 produces solutions from the efficient set. Each weight corresponds to a different point in the objective space and thus represents different routes for the fleet of EVs with varying user experience and operator profitability. Thus, the operator can select a suitable point for deployment based on the physical requirements of the system.

(a) 2 EV, 15 passengers

(b) 4 EV, 20 passengers

(c) 6 EV, 40 passengers

(d) 8 EV, 50 passengers

Figure 3.13 Pareto Front for Simultaneous Optimization of Passenger Travel Time and VMT

Runtime

The runtime to solve an instance of the mathematical formulation in Section 3.4.1 is given in Figure 3.14. Results are presented for two scenarios which reflect a low demand observed in the locality. In each graph, the horizontal axis shows the number of passengers for each scenario and the vertical axis shows the runtime in hours to generate optimal results. Each data point is shown using the blue diamond symbol. Also, the trend in increase of the runtime is given using the blue continuous line in Figure 3.14. In all experiments, EV capacity, driving range and tolerance factor are set to 10, 30 km and 4 respectively. Results show that optimal results can be obtained significantly faster for low complex scenarios with a few EVs and passenger demand. However, as the complexity of problem increases, the exponential increase of the runtime can be observed. For example, it takes only a few seconds to generate the optimal results for a test case with 5 EV and 10 passengers as opposed to several hours for a test case with 5 EV and 20 passengers.

Figure 3.14 Runtime Variation

3.6 Summary

This chapter proposed a demand-aware bus transit system that links a transit hub to the nearby neighbourhood using a fleet of EVs that operate over flexible routes and schedules. The proposed transit system leverages the hub & spoke topology of public transit network to provide seamless connectivity between the neighbourhood and transit hubs. It strives to bridge the connectivity gap created by the fixed routes and schedules of existing feeder bus services, while enhancing user experience and operator profitability.

In order to generate the optimal routes for a fleet of EVs, the proposed transit system is converted into a graph-based representation and subsequently modelled based on the widely used vehicle flow formulation using a mixed integer program. The mixed integer program was solved using branch and bound algorithm that prunes the design space to expedite the search without compromising optimality. In order to closely replicate real-life scenarios, the model relied on real distances of the road network as opposed to the Euclidean distances used in several existing works. In addition, asymmetric distance/time matrices that represent the different ‘to and fro’ distance/travelling times between points A and B, were used in the modelling step. The model uses the normalized weighted technique to generate the Pareto front, which enables the operator to analyse solutions with varying weights for user experience and operator profitability.

The proposed model has been evaluated using test cases generated on a real road network encompassing the Nanyang Technological University campus. Openly available public transit data has been used to quantify demand and supply parameters. The proposed model was verified to produce the same results as the exhaustive search. Further, the experiments conducted to analyse the impact of input parameters show that increasing the vehicle capacity by 25% can improve the travel time by 6% or reduce the VMT by 28%. Increasing the EV driving range by 25% can improve the travel time by 6% or reduce VMT by 13%. Experimental evaluations show the usefulness of the proposed model in performing diverse what-if analyses by varying different input parameters. In addition, the results also confirm the complex relationship between different input parameters on user experience and operator profitability.

The techniques proposed in this chapter can be used to generate the optimal routes for a fleet of EVs at a significantly faster rate when compared to an exhaustive search. However, the runtime of the formulation increases exponentially with the increasing problem complexity. Thus, the proposed model in this chapter can only be used to generate routes for real-time deployment in scenarios with low complexity. This necessitates accelerating route generation for complex scenarios. Hence, a genetic algorithm based heuristic technique that can produce near-optimal routes at a faster rate is presented in the next chapter.

Chapter 4

Accelerating Constraint-aware Flexible Route Generation

4.1 Introduction

Demand-aware bus transit systems have grown in popularity in the recent past. In a demand-aware bus transit system, ride requests of passengers are serviced by a fleet of medium capacity vans/buses, which do not follow fixed routes. As a result, the routes need to be computed in real-time. From the viewpoint of the operator, the fleet must be managed optimally in order to increase profits. On the other hand, passengers expect a better user experience. Thus, route generation of a demand-aware bus transit system is modelled as a dual-objective, constrained optimization problem with real-time requirements. Even though, the exact model proposed in Chapter 3 can generate optimal routes, it can be time consuming in the case of more complex problems. Therefore, the need to accelerate flexible route generation to further improve on the runtime by proposing less time consuming heuristic techniques that generate near-optimal solutions is identified. However, existing heuristic techniques do not account for simultaneously optimizing for the two conflicting objectives and also, do not consider constraints such as maximum travel time restrictions and EV driving ranges.

This necessitates the development of a robust, heuristic technique for accelerating flexible route generation of the demand-aware bus transit system.

Heuristic methods such as simulated annealing, tabu search and neighbourhood search have been used to generate flexible routes. However, in comparison to heuristic approaches that operate on a single solution, genetic algorithms operate on a solution pool. Thus, not only can it provide multiple feasible solutions but also does not get trapped in a local minima. As a result, it can generate a diverse set of solutions. Also, GA parameters can be tuned to expedite the search. Therefore, this chapter proposes a technique to combine a genetic algorithm with the ϵ -constraint method to obtain a set of feasible routes for simultaneously optimizing user experience and operator profitability. The technique can generate near-optimal routes in real-time. As a result, the technique has been subsequently used to compare the demand-aware bus transit service with an existing bus transit service.

The rest of the chapter is organized as follows. Section 4.2 discusses the methodology and results are presented in Section 4.3. Section 4.4 presents a comparison with existing bus transit. The conclusions are given in Section 4.5.¹

4.2 Methodology

This section discusses the proposed methodology for generating near-optimal routes in real-time while simultaneously optimizing user experience and operator profitability in a demand-aware bus transit system. The technique is based on combining an ϵ -constraint method and a genetic algorithm. The genetic algorithm relies on improving the routes of the fleet of EVs by intelligently selecting candidates from the pool of solutions that are subjected to genetic operations in each iteration. The techniques used in the design of the algorithm are explained in the following sections.

¹The work in this chapter has been published in [179] and [180]

Genetic Algorithm

Genetic algorithm (GA) is a subclass of evolutionary algorithms, which inherit principles of biological evolution. The basic principle of a GA is to manage a set of solutions that evolve with progressive generations. However, compared to other meta-heuristic algorithms such as tabu search and simulated annealing which operate on a single solution, GAs operate on a solution pool. A GA starts with an initial solution pool, called a population. Each solution in the population is termed as a chromosome and it represents a design point in the solution space. Chromosomes evolve progressively through successive iterations, termed generations. During each generation, the chromosomes are evaluated using a measure known as fitness. Fitter chromosomes are selected to perform genetic operations such as mutation and crossover. The mutation operator maintains the diversity in the population while the crossover operator speeds up the process to reach better solutions. In order to maintain a constant population size in each generation, some of the parents and offspring are rejected based on the fitness value. Finally, based on a termination criterion such as the number of iterations, the algorithm converges to a near-optimal solution to the problem.

ϵ -Constraint Method

The ϵ -constraint method for dual-objective combinatorial optimization relies on converting one of the objectives into an additional constraint. Therefore, the dual-objective combinatorial optimization problem can be defined as given in Equation 4.1. The value of the new constraint (ϵ_j) is usually provided as the nadir point of the efficient solutions (z_j^N). This is defined in Equation 4.2. This implies that when an objective is converted to a constraint, the corresponding constraint value should be the value of the objective space of the secondary objective when the primary objective has the ideal solution (utopia point) and vice versa. The utopia point is defined in Equation 4.3. This technique is used to evaluate the genetic operators in the proposed methodology.

$$\min f_i(x) \quad \text{subject to} \quad x \in X, \quad f_j(x) \leq \varepsilon_j, \quad i, j = 1, 2 \quad \& \quad i \neq j; \quad (4.1)$$

$$z_1^N = \min\{z_1 : z_2 = z_2^U\} \quad \text{and} \quad z_2^N = \min\{z_2 : z_1 = z_1^U\} \quad \forall z \in Z; \quad (4.2)$$

$$z_1^U = \min z_1 \quad \text{and} \quad z_2^U = \min z_2 \quad \forall z \in Z; \quad (4.3)$$

4.2.1 Dual-Objective Genetic Algorithm

The proposed GA starts with a set of candidate solutions that satisfies all constraints. These solutions evolve through generations until a termination criterion is reached. In each generation, a set of candidates are selected, which are subject to genetic operators that usually produce better offspring. An offspring that violates a constraint is removed from the population. Next, offspring and parents are combined to select a subset to be retained for the next generation. For a single-objective combinatorial optimization problem, selection is generally done using the fitness value for each solution. However, in the case of dual-objective combinatorial optimization problems (generally for multi-objective optimization problems), the main goal is to produce a Pareto front. Thus, the technique of selecting parents for the next generation is different compared to the fitness based technique used in single-objective GAs.

The main objective when selecting parents of a dual-objective combinatorial optimization problem is to generate the Pareto front as close as possible to the true Pareto front and also diverse in the objective space. Also, in order to be useful for real-time operations, the Pareto front should be generated quickly. Therefore, the design goal of an algorithm involves three factors, minimizing the distance from the true Pareto front, uniform diversity of the Pareto front in the objective space and fast convergence (low runtime) to the Pareto front. However, simultaneously optimizing for three design goals is often a cumbersome task. For example, having a large population will result in a long runtime and hence negatively impacts the real-time performance. In contrast, having a small population size requires reasonable selection of the population

to maintain diversity. Similarly, finding optimal results for a single-objective using the ϵ -constraint method will impact the diversity of the Pareto front. Thus, the following sections describe the techniques used in modelling the demand-aware bus transit system using a dual-objective GA to achieve the three main goals in generating the Pareto front.

Figure 4.1 Proposed Methodology of the Dual-Objective GA for Simultaneous Optimization of Passenger Travel Time and VMT

The block diagram of the proposed methodology is given in Figure 4.1. The ϵ -constraint method is utilized in each generation when selecting the optimization objective. Thus, each alternative generation will strive to optimize for a single-objective while the other objective is converted to a constraint. The terminology of the parameters used in the algorithm are summarized in Table 4.1.

Table 4.1 Terminology

Term	Description
V	EV fleet
P	set of all passengers
V_v	v^{th} EV
P_p	p^{th} passenger
C_v	seating capacity of v^{th} EV
D_v	driving range of v^{th} EV
MTT_p	maximum travel time of p^{th} passenger
n_v	number of passengers in v^{th} EV
vmt_v	vehicle miles travelled of v^{th} EV
tt_p	travel time of p^{th} passenger
iwt_p	initial waiting time of p^{th} passenger till scheduling instance
wt_p	waiting time of p^{th} passenger from the scheduling instance
$t_{i,j}$	shortest path travel time from origin i to destination j
$d_{i,j}$	shortest path distance from origin i to destination j

Solution Encoding

A solution, termed as a chromosome represents the route of all EVs. It gives the sequence in which passengers are visited. Thus, the encoding scheme converts the EV routes into a mathematical representation, which can be used to perform genetic operators. Therefore, the path-based integer encoding scheme is used to represent a chromosome. Further, in the proposed demand-aware bus transit system, the destination of the trip is the transit hub. Therefore, it is only required to encode the passengers assigned to an EV to distinguish the pick-up sequence. Hence, each passenger is represented using an integer.

Figure 4.2 shows a sample set of routes considering 3 EVs and 10 ride requests. In Figure 4.2, each node represents either an EV (triangle), ride-request (circle) or

the destination (square). Here, the three routes are encoded into three vectors which represent the pick-up sequence for each EV. For example, V_1 visits nodes 3 and 8 before reaching the destination.

Figure 4.2 Example Solution Encoding
 $V_1 : 3, 8$; $V_2 : 2, 7, 4, 6, 1$; $V_3 : 9, 5, 10$

Initial Population Generation

Initial population is a set of feasible solutions (chromosomes) which satisfy all the constraints of the problem. In general, chromosomes are generated using random assignment of passengers to EVs. The constraints are validated to ensure that the solution is feasible. However, works have proved that good initial solutions expedite the search process and helps to converge faster [181], which is crucial for the real-time performance of the proposed system. On the other hand, generating good initial solutions may be time consuming, which affects the runtime of the algorithm. Therefore, a fast minimalistic local search technique is used to generate a set of initial solutions which expedites the search process. Even though, the algorithm does not use complex search criteria, it can produce initial solutions that are far superior to random solutions. The pseudo code of the local search algorithm is given in Algorithm 1.

Algorithm 1 Local Search Algorithm**Input:** all passengers (P), EV fleet (V), distance matrix ($d_{i,j}$), time matrix ($t_{i,j}$)**Output:** routes of the EV fleet

```

1: Execution:
2: for  $p \in P$  do
3:   sort  $v \in V$  in ascending order of the local search criteria
4:   for  $v \in V$  do
5:     if capacity constraints are satisfied then
6:       if driving range constraints are satisfied then
7:         if travel time constraints are satisfied then
8:           assign passenger to  $v$ ;
9:           update EV parameters;
10:          update routes;
11:          break;
12:        else  $v = v + 1$ ;
13:      else  $v = v + 1$ ;
14:    else  $v = v + 1$ ;
15:  done
16:  if passenger is unassigned then
17:    update unassigned passengers list;
18: done

```

Here, for a selected passenger, all EVs are sorted based on a predefined search criterion (line 3). The search criteria can be (1) nearest neighbour assignment such as the distance from the EV location to the passenger, travel time from the EV location to the passenger; (2) priority-based assignment such as EV with the maximum remaining capacity. For example, if the search criterion is the distance from the EV location to the passenger, the EV closest to the passenger location is the first in the queue as opposed to the EV farthest away from the passenger location, which is the last. It should be noted that time and distance values are obtained using the OSRM route service APIs [173]. If the search criterion is maximum remaining capacity, EV with the maximum remaining capacity is filled prior to an EV with a lower remaining capacity. Then, based on the sorted list of EVs, the algorithm validates the three constraints (lines 5 - 7) and assigns the passenger to the first EV which satisfies all three criteria (lines 8 - 11). Passengers failing to meet all the criteria are added to a separate list at the end of execution. The algorithm repeats this sequence for all the passengers. Also, it can be recursively used with different local search criteria to

generate multiple solutions. It should be noted that generating a population with only local-search techniques can negatively affect the diversity of the population. Therefore, only 20% of the population is generated using the proposed local search technique while the remaining 80% is generated using random allocation. The selected ratio has been determined after testing the algorithm with multiple scenarios.

Constraints

Each chromosome has to be a feasible solution that satisfies the operational constraints given in Equation 4.4 - Equation 4.6. The three equations represent the EV capacity, driving range and maximum passenger travel time respectively.

$$\text{capacity: } n_v \leq C_v \quad \forall v \in V; \quad (4.4)$$

$$\text{driving range: } vmt_v \leq D_v \quad \forall v \in V; \quad (4.5)$$

$$\text{maximum travel time: } tt_p \leq MTT_p \quad \forall p \in P; \quad (4.6)$$

VMT of an EV and the travel time of each passenger is computed using the distance and time matrices obtained from the OSRM route service API. Equation 4.7 shows an example for VMT computation for V_1 in Figure 4.2. Equation 4.8 shows the travel time computation of P_1 in Figure 4.2. The travel time of P_1 is equal to the waiting time of P_1 (wt_1) and the riding time ($t_{P_1, Dest.}$). Here, since P_1 is the last passenger in the sequence, the riding time is equal to the direct travel time from the location of P_1 to the destination. Waiting time computation is performed using Equation 4.9. This is equal to the addition of initial waiting time before the scheduling instance (iwt_1), the algorithm execution time (t) and the EV travel time.

$$vmt_1 = d_{V_1, P_3} + d_{P_3, P_8} + d_{P_8, Dest.}; \quad (4.7)$$

$$tt_1 = wt_1 + t_{P_1, Dest.}; \quad (4.8)$$

$$wt_1 = iwt_1 + t + t_{V_3, P_2} + t_{P_2, P_7} + t_{P_7, P_4} + t_{P_4, P_6} + t_{P_6, P_1}; \quad (4.9)$$

Genetic Operations

Genetic operators produce the offspring by using crossover and mutation operators. Crossover reflects the actual reproduction operation while mutation ensures genetic diversity. Most of the GAs use different combinations of genetic operators and thus mainly vary in this phase. Further, based on the encoding scheme specialized operators can be used for fast convergence and maintaining diversity in the objective space.

Mutation In the mutation phase of the GA, two mutation operations are used, namely exchange mutation and insertion mutation. Each operation is performed on a route of a chromosome. In exchange mutation, the position of a randomly selected sub-tour of a route is exchanged to optimize the route. The 2-opt local search heuristic is used to find the optimal position to exchange the sub-tour. For example, in Figure 4.3a the initial route of $2 \rightarrow 7 \rightarrow 4 \rightarrow 6 \rightarrow 1$ is optimized by exchanging the sub-tour $4 \rightarrow 6 \rightarrow 1$ (outlined using a red-dashed rectangle). The final route is $2 \rightarrow 4 \rightarrow 6 \rightarrow 1 \rightarrow 7$.

Figure 4.3 Mutation Operators

In insertion mutation, a randomly selected gene is inserted into a different position in the route using Clarke-Wright savings heuristic [34]. Therefore, while generating the offspring, the selected gene is inserted to the route which generates the maximum

decrease/minimum increase of the objective. The example in Figure 4.3b shows the generation of the offspring $2 \rightarrow 6 \rightarrow 7 \rightarrow 4 \rightarrow 1$ from the initial route of $2 \rightarrow 7 \rightarrow 4 \rightarrow 6 \rightarrow 1$ by inserting gene 6 (shown outlined using a red-dashed square) into the new position.

Crossover The crossover phase of the GA is performed using two crossover operations, namely heuristic crossover and re-insertion crossover. Crossover operation is performed on two routes of a chromosome. In heuristic crossover, genes with the lowest contribution to the fitness value of the EV routes are swapped between the two routes. In the example given in Figure 4.4a, genes 4 of V_2 and 10 of V_3 are crossed between the two routes. It shows the two selected genes in red and blue dashed squares.

Figure 4.4 Crossover Operators

During re-insertion crossover, a single gene of a selected route is inserted into the route of another EV. However, in this operation, the gene with the lowest contribution to the fitness value is selected for re-insertion. Further, while inserting into the new route, the 2-opt local search is used to find the best position. In Figure 4.4b, gene 7 of V_2 is inserted to the route of V_3 . Figure 4.4b shows the selected gene and the new insertion position in a red-dashed square.

Offspring Validation

The offspring generated using the crossover and mutation operators are validated for the constraints in this step. It ensures that the offspring consists of valid routes. Therefore, offspring are validated for the capacity (Equation 4.4), driving range (Equation 4.5), maximum travel time (Equation 4.6) and finally the appropriate ε -constraint (Equation 4.2) for each generation. For example, a generation which optimizes for VMT is validated with the nadir point of passenger travel time and vice versa. The nadir points of the two objectives can be obtained by using the single-objective GA presented later in this chapter. An offspring which fails the validation is removed from the population.

Selection

After offspring validation, the remaining offspring and parents of the previous generation are combined to produce the population for the current generation. However, if the population size is not maintained at a fixed value, it results in an exponential increase in the runtime of the algorithm. Therefore, at this stage, suitable candidates have to be retained for the next generation (select n solutions from the population (X), where $n \ll |X|$). For example, even though selecting solutions from the efficient set ($\mathcal{E}$) benefits solution quality, it can negatively affect the uniform diversity in the objective space. Preserving diversity in the population is important such that unexplored regions can be traversed.

Figure 4.5 Illustration of Selecting Parents

A hybrid technique for parent selection is proposed. The technique combines the efficient set with a cell-based ranking method proposed in Yen et al. [182]. In the cell-based ranking method, the objective space is divided into cells and a density is computed for each cell based on the number of solution points occupied in the cell. Therefore, if the solution points are highly concentrated in a single cell, the density of that cell will be higher. Further, it implies that the surrounding region in the objective space has been thoroughly searched. However, a cell with a lower density has a high probability of generating a new offspring in the objective space which has not been explored before. Thus, cells with a low density are given priority during selection. This scenario is explained using Figure 4.5, where the objective space (Z) of a dual-objective combinatorial optimization problem is shown in the two axes with $z_1 = f_1(x)$ and $z_2 = f_2(x)$. The Pareto points are marked using the star symbol and all other points are shown in circles. Here, the objective space is divided into 4 cells. The cell density of each cell is 6, 3, 7 and 2 respectively. Thus, the density implies that cell 4 has the highest potential to produce an offspring that maps to the unexplored region in the objective space. This is followed by cell 2. However, selecting parents using only cell density can be detrimental to the solution quality as most of the Pareto points are located in cells 1 and 3. This motivates to propose a hybrid parent selection

technique that inherits the benefits of quality using the Pareto front and diversity using cell-based density.

This hybrid parent selection technique is given in Algorithm 2. It consists of two stages, namely pre-processing (line 1 - 5) and execution (line 7 - 35). In the pre-processing stage, the algorithm finds the Pareto front (Δ) of the population (X) (line 2) and initializes the selected parents (I) (line 3), cell-density based solutions (S_c) (line 4) and Pareto front based solutions (S_p) (line 5) to the null set. During the execution stage, the algorithm strives to find a balance between optimality and diversity by selecting 50% of the parents each from the efficient set ($\mathcal{E}$) and the remaining solutions ($X \cap \mathcal{E}$). Next, the algorithm computes the number of solutions in the Pareto front ($|\Delta|$). If it is less than 50% ($0.5 * n$) (line 8), all the solutions of the efficient set are selected as parents (line 9). Next, the algorithm fills the parents from the remaining solutions (lines 10 - 18). Here, the cell densities are calculated for each cell (line 13). Then, the cells are sorted based on cell density (line 15) and a solution is selected randomly from each cell (starting from the lowest cell density) until the required number of parents are filled (line 16). Finally, they are added to the cell-density based solutions set (line 17). This is combined with the Pareto front based solutions set to produce the selected parents (line 18). If the number of solutions in the Pareto front exceed the 50% limit (line 19), the solutions of the Pareto front are ranked based on the product of the objectives $f_1(x) * f_2(x)$ (line 21). Then, the solutions are divided into $0.5 * n$ groups (line 23). Then, a solution is selected from each group (line 24) and added to the Pareto front based solutions set (line 25). When the required solutions are identified, they are included in the selected parents set (line 26). Similar to the case explained above, the algorithm fills the parents from the remaining solutions (lines 27 - 35). Therefore, the proposed technique not only maintains a constant population size in each generation to reduce the runtime explosion but also provides solution quality and diversity.

Algorithm 2 Parent Selection Algorithm

Input: population (X), number of parents (n)**Output:** selected parents (I ; where $|I| = n$)

```

1: Pre-Processing:
2:   find the Pareto front ( $\Delta$ );
3:    $I = \emptyset$ ;
4:    $S_c = \emptyset$ ;
5:    $S_p = \emptyset$ ;
6:
7: Execution:
8: if  $|\Delta| \leq 0.5 * n$  then
9:    $I = I \cup \Delta$ ;
10:   $n = n - |\Delta|$ ;
11:   $X = X \cap \Delta$ ;
12:  for  $x \in X$  do
13:    calculate cell densities;
14:  done
15:  sort cells based on density;
16:  select  $x'$  from each cell until  $n$  is reached;
17:   $S_c = S_c \cup x'$ , where  $f(x') = z'$ ;
18:   $I = I \cup S_c$ ;
19: else
20:   for  $x \in \Delta$  do
21:     sort  $x$  based on  $f_1(x) * f_2(x)$ ;
22:   done
23:   divide  $\Delta$  into  $0.5 * n$  distinct groups;
24:   select  $x''$  from each group;
25:    $S_p = S_p \cup x''$ , where  $f(x'') = z''$ ;
26:    $I = I \cup S_p$ ;
27:    $n = 0.5 * n$ ;
28:    $X = X \cap \Delta$ ;
29:   for  $x \in X$  do
30:     calculate cell densities;
31:   done
32:   sort cells based on density;
33:   select  $x'$  from each cell until  $n$  is reached;
34:    $S_c = S_c \cup x'$ , where  $f(x') = z'$ ;
35:    $I = I \cup S_c$ ;

```

Termination Criteria

There are multiple techniques that can be used to terminate a GA such as (1) predefined number of iterations, (2) limiting the runtime of the algorithm, (3) observing the solution quality and terminating if there is no change. In the proposed method, criteria (1) and (3) are combined. The main criterion is the number of iterations. However, since runtime is crucial for real-time performance, the algorithm also computes the sum of the objectives ($f_1(x) + f_2(x)$, $\forall x \in \mathcal{E}$) for each generation and if there is no improvement in 5 consecutive iterations, further execution is prevented.

Generating the Nadir Points

Figure 4.6 Block Diagram of the GA for Generating Nadir Points

The methodology given in Figure 4.1 produces the Pareto front of the dual-objective optimization problem discussed in this chapter. However, for offspring validation the corresponding nadir points of the two objectives need to be obtained. For this, a single-objective GA can be used to optimize either the user experience or operator profitability. The block diagram of the GA for single-objective optimization is given

in Figure 4.6. It should be noted that the structure of the single-objective GA is similar to the method presented in Figure 4.1. All the blocks in Figure 4.6 use the same techniques mentioned for the dual-objective GA. However, for selection of parents, a fitness-based selection strategy can be used instead of the hybrid parent selection strategy. Here, fitness of each solution is evaluated based on the optimization objective, which can be performed by aggregating the fitness of each chromosome using Equations 4.7 and 4.8 for operator profitability and user experience optimization respectively. Therefore, the fittest solutions are retained for the next generation.

4.3 Performance Evaluation

This section presents the experimental results of the proposed algorithm. The same zone used in Chapter 3 encompassing the NTU campus is used for creating the test cases. All assumptions considered in Chapter 3 on location of EVs, random distribution of passenger requests, driving range and capacity constraints of EVs, time and distance matrix generation using OSRM are considered in the experiments. The main purpose of the computational experiments is to show the benefits of the proposed routing algorithm for real-time deployment in demand-aware bus transit. Therefore, experiments are conducted to show the accuracy, convergence and computation time of the results of different scenarios by varying the supply and demand. The proposed GA is implemented in C++ and runtime is measured on a PC with 16 GB RAM, running Windows 7 on an Intel Xeon E5-1650V2 CPU at 3.50 GHz.

4.3.1 Experimental Setup

Experimental Parameters

Table 4.2 shows the values of the parameters used in the experiments. EV capacity, driving range and tolerance factor are set to 8, 30 km and 4 respectively. In the GA,

number of cells for the two objectives, population and number of iterations are set to 20, 50 and 100 respectively. These values have been used based on similar experiments performed to tune the parameters.

Table 4.2 Values of the Parameters

Parameter	Value
EV Capacity	8
EV Driving Range	30 km
Tolerance Factor	4
Number of Cells	20
Population Size	50
Iterations	100

Test Cases

Experiments are conducted by generating multiple test cases by varying the demand and supply and comparing the performance of the proposed algorithm. Table 4.3 shows the test cases used in the experiments. Here, the demand value shows the total number of ride requests considered as input in the algorithm and the number of empty EV are given as the supply. Compared to Chapter 3, these test cases use relatively large demands.

Table 4.3 Test Cases

Test Case	Demand (Requests)	Supply (EV)
1	50	10
2	75	15
3	100	20
4	125	25

4.3.2 Experiments

Runtime results in Chapter 3 showed that optimal results can be obtained for relatively small test case in a reasonable time duration. Thus, the performance of single-objective

optimization scenarios for small test cases are evaluated by comparing with optimal values. Using the same setup, experiments are conducted to optimize either the user experience or operator profitability. Next, for the dual-objective case, impact of using local search technique in the initial population generation is shown. Next, the quality of the Pareto front obtained from the proposed algorithm is compared with the NSGA-II algorithm [183], cell density-based parent selection strategy and selecting all parents of the efficient set strategy. It should be noted that, due to the excessive computation time required to generate optimal results for relatively large test cases, the Pareto front is not compared with the true Pareto front. Finally, the runtime results of the proposed algorithm are shown.

User Experience Optimization

Results for optimizing passenger travel time are shown in Table 4.4. In each test case, the experiment is repeated 5 times and the average values are presented. The columns, ‘Travel time GA’ (column 4) and ‘Travel time Optimal’ (column 5) show the results obtained from the proposed single-objective GA and the optimal formulation presented in Chapter 3. Results show an average deviation of 2.49% from the optimal values for the considered test cases. This confirms the suitability of the proposed GA for optimizing passenger travel time.

Table 4.4 Results for User Experience Optimization

Test Case Number	Parameters		Travel Time (min.)		Deviation (%)
	Vehicles	Passengers	GA	Optimal	
1	4	6	95.6	93.6	2.17
2	4	7	106.4	104.4	1.99
3	4	8	131	126.2	3.73
4	5	8	117.4	115.2	1.95
5	6	8	117	114.6	2.10
6	7	10	143.8	139.6	3.00

Operator Profitability Optimization

Results for optimizing VMT is shown in Table 4.5. Similar to the previous experiment, the values shown are the average values obtained by repeating each test case 5 times. Experimental results indicate that the VMT of the routes generated by the proposed single-objective GA is on average 2.58% higher compared to optimal values. However, as shown later, the runtime of the proposed algorithm is much superior compared to the optimal formulation. Therefore, the near-optimal results that are generated can be effectively used for real-world deployment of the demand-aware bus transit system.

Table 4.5 Results for Operator Profitability Optimization

Test Case Number	Parameters		Travel Distance (km)		Deviation (%)
	Vehicles	Passengers	GA	Optimal	
1	5	10	9.6	9.5	0.96
2	5	15	10.7	10.7	0.43
3	6	18	11.5	11.2	2.67
4	6	14	12.0	11.5	4.33
5	7	17	13.4	12.8	4.50

Impact of Local Search Technique for Initial Population Generation

As mentioned in Section 4.2.1, high quality initial solutions expedite the search and helps to converge faster. To validate this, experiments are performed to show the benefits of the local search algorithm (Algorithm 1). Experiments are conducted using the test cases in Table 4.3. Each test case is repeated twice (henceforth referred as instance 1 and 2) by randomly changing the distribution of the supply and demand in the zone. The experiment is executed by running each test case using the proposed algorithm (with local search technique) and without the local search technique. Next, the two Pareto fronts obtained after the same number of iterations are compared.

Results for the test cases are shown in Figure 4.7. In all the sub-figures of Figure 4.7, the horizontal axis shows the normalized total passenger travel time and the vertical axis shows the normalized VMT of the EV fleet. The blue diamonds and orange dots show the Pareto front obtained from the proposed algorithm and without the local

Figure 4.7 Impact of the Local Search Technique

search technique respectively. Results confirm that the proposed local search technique provides significant benefits in generating a high quality Pareto front. Further, as the complexity of the test case increases (high demand and supply), the distance between the two Pareto fronts has also increased. This can be attributed to the large search space of the test cases. For example, in Figure 4.7g - Figure 4.7h the search space for the genetic operators is significantly large compared to instances in Figure 4.7a - Figure 4.7b. Therefore, an algorithm without a local search technique will take a considerable time to generate a high quality Pareto front. However, as the proposed local search technique provides a superior initial starting point to the algorithm, it can generate a superior Pareto front.

Comparison of the Pareto Front

In this experiment, the performance of the proposed algorithm is compared using the test cases in Table 4.3. Here, the Pareto front obtained from the proposed algorithm is compared with that of a cell-density based parent selection strategy, selecting all parents of the efficient set and the NSGA-II algorithm. The only difference between the proposed methodology and cell-density based parent selection strategy and selecting all parents of the efficient set is the different parent selection strategies. Cell-density based parent selection strategy uses the same population size as the proposed algorithm. Here, parents are selected using only on cell densities. The strategy of selecting all parents of the efficient set uses a variable population size. NSGA-II algorithm uses the same parameters as the proposed algorithm. The number of iterations of all the techniques are set to 100. Each test case in Table 4.3 is repeated twice and the results are shown in Figure 4.8. The horizontal axis shows the total passenger travel time and the vertical axis shows the VMT of the EV fleet. The Pareto front obtained by the NSGA-II, proposed algorithm, cell-density based parent selection and selecting all the parents of the efficient set are shown using blue diamonds, orange squares, grey triangles and black crosses respectively.

(a) Test Case 1 - Instance 1

(b) Test Case 1 - Instance 2

(c) Test Case 2 - Instance 1

(d) Test Case 2 - Instance 2

(e) Test Case 3 - Instance 1

(f) Test Case 3 - Instance 2

(g) Test Case 4 - Instance 1

(h) Test Case 4 - Instance 2

Figure 4.8 Comparison of the Pareto Fronts

In general, the proposed algorithm generates solutions which are superior to the solutions obtained from the NSGA-II algorithm using the same parameters. However, NSGA-II algorithm generates a diverse population. In contrast, the cell-density based selection and selecting all parents of the efficient set results in poor solutions in terms of quality and diversity. In the case of cell-density based selection strategy, the selective pressure on retaining superior parents is not considered. Therefore, the selection strategy always finds solutions in the unexplored objective space. While this can result in potential new pareto optimal points, the convergence of such a technique is significantly slow. In contrast, when selecting all parents of the efficient set, the objective space is denied the opportunity to explore other regions, which limits the search to a local minimum. Thus, in most cases the Pareto front obtained from this strategy is relatively close to the Pareto front obtained from the NSGA-II and proposed algorithm. However, the Pareto front of selecting all parents of the efficient set significantly lacks diversity.

Even though, the diversity of the objective space is relatively low in the proposed algorithm, the operator has to select only one solution for real-life deployment of the demand-aware bus transit system. From an operational point of view, the main objective in such a scenario is to rapidly generate a few superior solutions rather than obtaining a significantly diverse solution set. Each solution of the efficient set obtained from the proposed algorithm gives a unique set of routes to pick-up passengers. For example, the two ideal solutions of operation in Figure 4.8a are 14% and 10% apart in the objective space respectively for VMT and passenger travel time. This implies that the VMT of the fleet would benefit 14% by using the operator ideal solution where as, the routes corresponding to this are penalized by 10% in terms of passenger travel time compared to the ideal solution for passengers. However, in all cases, passengers are guaranteed to meet the maximum travel time constraints. In general, the solutions generated by the proposed algorithm have a spread of 6.75% for passenger travel time and 7.25% for VMT. .

Runtime

This experiment is conducted to measure the speed-up that can be obtained from the proposed algorithm in comparison to the NSGA-II algorithm. For this, the proposed algorithm is executed with a population size of 50 and 100 iterations. For the NSGA-II algorithm, the same population size of 50 is used. However, the algorithm is executed with a cut-off time limit of 1 hour and the obtained Pareto front and runtime in each iteration is logged. Then, Pareto front obtained from the proposed algorithm is compared with the Pareto fronts for each iteration and the one which has the minimum generational distance is selected for comparison. The runtime of the two techniques are then used to compute the speed-up.

Figure 4.9 Runtime Comparison with NSGA-II Algorithm

The test cases in Table 4.3 are repeated 3 times using the above comparison technique for this experiment. The corresponding average runtime of the test cases are shown in Figure 4.9. Here, the horizontal axis shows the test case number and the vertical axis shows the average runtime of the test case. The black column shows results for NSGA-II algorithm and the white column shows results of the proposed algorithm. The results show that runtime of both algorithms increase with the complexity of the test case. However, compared to the NSGA-II algorithm, the proposed algorithm shows better scalability for large test cases. On average, a runtime improvement of 26% is observed for the proposed algorithm in comparison to NSGA-II algorithm. Runtime

improvement is obtained due to the proposed parent selection algorithm. The number of computations that needs to be performed in the proposed technique is relatively low compared to the NSGA-II algorithm, which requires multiple Pareto front and crowding distance computations. Therefore, the proposed algorithm is beneficial for deployment in a demand-aware bus transit system compared to the NSGA-II algorithm. It should also be noted that compared to the runtime values obtained in Chapter 3, the results show a significant gain. Thus, the proposed algorithm is suitable to accelerate flexible route generation of demand-aware bus transit system for relatively large demands.

4.4 Comparing with Existing Bus Transit System

The heuristic routing technique proposed in this chapter enables real-time deployment of the proposed demand-aware bus transit system for enhancing user experience and operator profitability. However, for the proposed demand-aware bus transit system to be deployed as an alternative to existing fixed schedule, fixed route bus transit system, it needs to be evaluated for user experience and operator profitability. For example, passengers will prefer to use the existing fixed schedule, fixed route bus services if the waiting/riding time is lower compared to the proposed demand-aware service even though, passengers have a guaranteed seat. Thus, in this subsection, the proposed demand-aware bus transit service is compared with the existing bus transit service using a case study. In the case study, two scenarios with different scheduling strategies are considered for the feeder bus service, namely fixed schedules and demand-aware schedules. Details of the two techniques are given in the relevant subsections. First, the experimental setup is explained.

Setup

The two alternatives are compared by analysing the route of a single feeder bus service due to the complexity of using multiple services. For this, the 243G bus service originating at Boon Lay transit hub in Singapore is used. Bus service 243G serves the residents and schools around Jurong West neighbourhoods connecting these estates to the shopping centre and Boon Lay transit hub. Auxiliary data of the 243G bus service such as the maximum waiting time, the number of bus stops, travel time and distance between bus stops are extracted from openly available transit data [184] [185]. The extracted data is summarized in Table 4.6 and Table 4.7.

Table 4.6 Auxiliary Data of Feeder Bus Service 243G

Parameter	Value
Maximum Waiting Time (min.)	3 - 8
Number of Bus Stops	13

Table 4.7 Travel Time and Distance Between Bus Stops of Bus Service 243G

Bus Stop ID	22441	22591	27371	27381	27391	27409	27419
Travel Time between stops	5	2	2	3	2	2	2
Travel Distance between stops	800	300	400	400	200	300	200

Bus Stop ID	27361	27309	22369	22239	22249	22259	22009
Travel Time between stops	3	3	2	2	2	2	3
Travel Distance between stops	500	400	200	300	200	400	500

The case study considers a scenario where 70 passengers request for rides in the selected 13 bus stops in a 15 minute timeframe. The number of passengers has been selected based on the real demand observed on the selected bus service. In order to simplify travel time comparison, it is assumed that the feeder bus does not facilitate point-to-point transit in the route, except for passengers who will alight at the transit

hub. Therefore, all passengers that use the two transit methods are assumed to be travelling to the transit hub. In order to directly compare the alternatives, EVs proposed for deployment in the demand-aware bus transit system are replaced by fossil-fuel based vehicles.

In the proposed demand-aware bus transit system, vehicles do not have to follow a fixed route or a fixed schedule. In contrast, for fixed route public transit, it is assumed that vehicles are scheduled every 5 minutes. It should be noted that the selected scheduling frequency of 5 minutes, falls within the range of the maximum waiting time for the 243G bus service as shown in Table 4.6. However, for the case of demand-aware scheduling of fixed route public transit, scheduling is performed dynamically based on the predefined criteria explained later. It should be highlighted that the frequency of 5 minutes selected for the existing fixed route public transit bus service can be a high value in certain cities. In such a scenario, the same experiment can be performed with a large time period and an increased scheduling period. However, as mentioned in Section 3.3, the proposed demand-aware transit system can schedule passengers when the demand reaches 80% of EV utilization. Thus, the scheduling algorithm will not be affected by the extended scheduling period of the fixed route public transit service.

In terms of vehicle capacities, it is assumed that the operator of the proposed demand-aware bus transit system has the choice of deploying a different type of vehicle available in the fleet based on the demand and profitability (mini vans, mini buses). For fixed route transit services, it is assumed that the operator has only single decker buses with a capacity of 84. However, to analyse the impact of vehicle capacity, a hypothetical case with mini vans/mini buses is also considered. Tolerance factor is set at 4 for each passenger request which guarantees a maximum travel time constraint.

Although, the proposed methodology can generate a Pareto front to reflect the routes of different scenarios, the scenario for optimizing VMT is considered for the analysis of the proposed demand-aware bus transit system. As shown in Section 3.5, this scenario generates routes with the worst travel time for passengers. Hence,

comparing with this scenario will ensure that all other routes outperform the feeder bus service.

For the proposed demand-aware bus transit system, an experiment is performed to analyse the impact of vehicle capacity on passenger waiting time and the VMT of the fleet. The results of the study allow the operator to select the type of vehicle that should be used based on the demand. It is assumed that similar experiments can be performed offline and a vehicle database can be updated with the suitable vehicle type. Therefore, it can be used to select the type of vehicle at runtime. When obtaining the VMT, the generated results are multiplied by 2 to reflect passengers traveling to and from the transit hub, which enables a realistic comparison with the feeder bus service. Also, it should be noted that since different types of vehicles are used in the case study (mini vans/mini buses/buses), VMT is converted to a fuel consumption metric for analysis. For example, a low VMT for buses can be more costly in terms of fuel consumption compared to a relatively high VMT for mini vans. Thus, the metrics provided by the Bureau of Transportation Statistics of the United States Department of Transport is used in the analysis [186]. The metrics are summarized in Table 4.8.

Table 4.8 Comparison of Vehicles

Vehicle	Capacity	Fuel Consumption (km/l)
Mini Van	8 - 16	10.6
Mini Bus	18 - 22	7.5
Bus	84 - 88	3.5

Comparison with Fixed Route, Fixed Schedule Bus Transit

Comparison with fixed route, fixed schedule bus transit enables to evaluate the performance with the existing feeder bus transit services. Existing feeder bus transit services do not adjust the schedules based on real-time demand. Also, buses travel on fixed routes even if passengers are not present in the bus stops. It is assumed that the buses are scheduled every 5 minutes.

Comparison with Fixed Route, Demand-aware Scheduling of Bus Transit

A demand-aware scheduling technique can be used for scheduling short distance feeder bus services if the future demand is known especially in areas of low-demand in order to mitigate the costs of operating a fixed schedule service. In areas of low-demand, existing fixed route, fixed schedule transit services are operated at a very low frequency. Thus, the waiting time for a bus is significantly high. Also, the number of passengers boarding the bus may be very low, which is not profitable for an operator. For instance, in the worst case, buses will be travelling empty for a long duration of the journey. Thus, operators can explore the technique of demand-aware scheduling of fixed route bus transit systems. Therefore, in this case study, it is assumed that a vehicle will be scheduled in one of following criteria, (1) if the demand reaches 80% of the vehicle capacity, (2) in the case of insufficient demand, at a low frequency compared to existing service (in the case study, a vehicle will be scheduled every 10 minutes).

Passenger Distribution

Table 4.9 presents the distribution of passengers in the 13 bus stops of service 243G for the selected 15 minute timeframe. Here, $T = 1$ in the time column (column 1) represents the current time. The remaining columns represent the different bus stops in the sequence given in Table 4.7. The values given in Table 4.9 indicate the total number of new passengers (arrivals) that needs to be picked up by the vehicles. Therefore, it is considered that there are no existing buses that are operating in the route prior to $T = 1$, which can pick-up these passengers.

For the case of fixed route and fixed schedules, feeder buses are scheduled at $T = 1$, $T = 6$ and $T = 11$. For demand-aware scheduling of fixed route feeder bus, the algorithm will periodically (every minute) evaluate the utilization ratio of the bus and will dispatch when the criterion is satisfied. During evaluation, the algorithm will add up the demand at a given bus stop up to the estimated time of arrival (ETA). ETA is calculated by summing current time and travel time between the transit hub

Table 4.9 Passenger Distribution at Bus Stops

Time	Bus Stop												
	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8	B_9	B_{10}	B_{11}	B_{12}	B_{13}
1	0	1	0	0	2	0	0	1	1	0	0	1	0
2	1	2	1	0	1	0	0	0	1	0	0	0	0
3	0	0	0	1	1	0	0	0	0	0	1	0	1
4	1	1	0	0	0	0	1	1	0	0	0	0	0
5	1	1	1	0	0	0	0	0	0	0	0	0	0
6	0	0	1	1	0	0	1	0	0	1	1	0	0
7	0	0	0	0	1	0	0	1	0	0	1	0	0
8	0	1	0	1	0	1	1	0	1	0	0	1	0
9	0	0	1	1	0	1	1	0	0	0	0	0	1
10	0	0	0	0	0	0	0	0	0	0	1	3	0
11	1	0	0	0	0	0	0	2	0	0	0	0	0
12	0	1	0	0	1	1	0	1	1	0	1	0	0
13	1	0	0	1	0	1	0	0	0	1	0	0	1
14	0	0	0	0	0	1	0	0	1	0	1	1	0
15	1	0	0	1	1	0	1	0	1	0	0	0	1

and the bus stop. For example, given that the travel time between the transit hub and bus stop B_1 and bus stops B_1 and B_2 are 5 and 2 minutes respectively, the ETA at bus stop B_2 is $T = 8$ for the bus that leaves the transit hub at $T = 1$. The corresponding demand by $T = 8$ for data given in Table 4.9 is 6. This process is repeated in a similar way for the remaining bus stops. Once a bus is scheduled, the appropriate number of passengers will be updated. For the case of the proposed demand-aware bus transit system, schedules are computed based on the same strategy and the routes are computed as explained in Section 4.2.

Selection of Vehicles for the Proposed Demand-aware Bus Transit System

The capacity of the vehicles used in the proposed demand-aware bus transit system plays a significant role in determining the user experience and operator profitability. Therefore, the impact of vehicle capacity on passenger waiting time and the VMT of the proposed demand-aware bus transit system is analysed. It is assumed that the fleet consists of 10 vehicles which are available for scheduling at $T = 1$. Figure 4.10

Figure 4.10 Impact of Vehicle Capacity on Average Waiting Time and VMT

presents the results obtained by executing the techniques proposed in Section 4.2. In Figure 4.10, the horizontal axis gives the vehicle capacity while the left vertical axis gives the average waiting time and right vertical axis gives the VMT of the fleet. The figure shows the trend of reducing VMT (increasing waiting time) with vehicle capacity, which subsequently stabilizes even if the capacity is increased. The trend can be explained using the travel time constraint (tolerance factor) of the proposed demand-aware bus transit system. Due to the imposed travel time constraint, increasing the vehicle capacity after a certain value has no impact on the routes selected by the algorithm. Alternatively, the optimization capability of the algorithm is constrained by the travel time. Therefore, the number of vehicles used is not reduced further and hence the same VMT. From the results, the vehicle which has the minimum VMT is selected as the type of vehicle for deployment in the proposed demand-aware bus transit system, which in this case has a capacity of 16.

Evaluation

Table 4.10 shows the results of the case study. In Table 4.10, rows 1 & 2 give results for fixed-route, fixed-schedule transit. Rows 3 & 4 give results for fixed-route, demand-aware scheduling and rows 5 & 6 give results of the proposed demand-aware bus

transit system. Rows 1 & 3 represent scenarios with 84 seat buses and rows 2 & 4 show the results for the hypothetical case of mini vans. Row 6 represents a scenario with the same number of vehicles used as the fixed-route, demand-aware scheduling instance of row 4. However, the travel time constraints are relaxed in this case. The columns represent the capacity of the vehicles, the scheduling policy, number of buses in operation, average travel time of passengers, VMT of the fleet of vehicles and fuel consumption of the fleet respectively.

Table 4.10 Comparison of Results

Capacity	Schedule	No. of Buses	Average Time	Travel	VMT (km)	Fuel Con- sumption (l)
84	Fixed Fre- quency	8	35 min. 36 sec.		40.8	13.16
16	Fixed Fre- quency	8	44 min. 06 sec.		40.8	3.85
84	Demand- aware	3	37 min. 56 sec.		15.3	4.94
16	Demand- aware	7	34 min. 20 sec.		35.7	3.37
16	Proposed	9	25 min. 32 sec.		44.3	4.18
16	Proposed	7	29 min. 12 sec.		33.2	3.13

In order to operate with a fixed frequency for the selected bus service, 8 vehicles are required irrespective of the capacity of the vehicle. This is due to the 35 minute estimated round trip time of the service. However, when the capacity of the vehicle is low (row 2) the average travel time of passengers is significantly high. The reason for the increase can be accounted for the low capacity of vehicles, which makes the seats of the mini vans fill up prior to reaching bus stops farther away from the transit hub. Therefore, passengers must wait extra time till an empty mini van reaches the bus stop. In the case of demand-aware scheduling, the number of vehicles required is significantly reduced for the case of buses (row 3). However, the average travel time for this case, has increased compared to the fixed frequency (row 1) service. This happens as a result of the low demand for the service. Therefore, the demand-aware scheduling algorithm will schedule buses at a low frequency (every 10 minutes) compared to

the fixed frequency case of 5 minutes. In contrast, when mini vans are used, the travel time is reduced since vehicles are scheduled more frequently (row 4). For the case of the proposed demand-aware bus transit system, the travel time of passengers is low compared to all other scenarios (row 5). The imposed travel time constraint ensures that passenger travel time is improved compared to the existing fixed route bus service. As a result, 9 vehicles are required to meet the travel time constraint. Therefore, the VMT of the fleet is high. This is in line with the examples provided in Section 3.5. However, when the travel time constraints are relaxed, such that number of used vehicles match (row 6) the number of buses used in the demand-aware service with mini vans (row 4), the travel time of passengers increase. This is due to the fact that the algorithm can optimize the VMT by reducing the number of used vehicles such that travel time constraints are not violated. The fuel consumption results indicate that buses (row 3) consume a considerable amount of fuel, even when the number of vehicles and VMT is low. Results convey that the proposed demand-aware bus transit service provides a 15% benefit in travel time compared to a fixed route, demand-aware schedule (row 4 and row 6). This is achieved with a 7% reduction in VMT, which increases operator profitability. At the same time, compared to existing fixed route, fixed schedule transit services the proposed system can provide a 28% improvement in travel time for passengers (row 1 and row 5). Thus, experimental results confirm the benefits of the proposed demand-aware bus transit system. Therefore, it is beneficial to further extend the techniques for citywide deployment of the proposed demand-aware bus transit system.

4.5 Summary

This chapter proposed a heuristic routing technique that generates near-optimal routes for a fleet of EVs to be deployed in real-time to simultaneously enhance user experience and operator profitability in a demand-aware bus transit system. The proposed methodology that combines a GA and the ϵ -constraint method has been shown to not

only be rapid but also highly accurate. This is achieved by using methods such as nearest neighbour heuristic to generate superior initial solutions, selection of genetic operators for fast convergence and a hybrid parent selection algorithm for balancing solution quality and diversity in the solution space.

The proposed methodology has been evaluated using test cases generated on a real road network encompassing the Nanyang Technological University campus, which is the largest university campus in Singapore. Experiments conducted based on openly available public transit demand data show that the routes deviate only 3% from the optimal values. Further, experiments confirm that the Pareto front converges significantly fast helping generate optimized routes for time-constrained applications. In comparison to a widely-used baseline algorithm, the proposed methodology can achieve a 26% improvement in runtime with marginal loss of diversity. This affirms the suitability of the proposed methodology for accelerating flexible route generation for a demand-aware bus transit system.

The techniques proposed in this chapter can be used to generate near-optimal routes for a fleet of EVs significantly faster when compared to the exact model proposed in Chapter 3. However, the runtime of the proposed methodology increases with the complexity of the problem, affecting the responsiveness. This necessitates the development of a technique that can exploit parallel computing to speed-up the computation of routes while maintaining accuracy. Hence, a directionality-centric bus transit network segmentation technique that can leverage parallel processing to speed-up the computation and also be accurate is presented in the next chapter.

Chapter 5

Directionality-centric Segmentation of Bus Transit Network

5.1 Introduction

Demand-aware bus transit systems, unlike the fixed route counterparts respond to passenger demand by changing the routes of the fleet of vehicles. Hence, it necessitates to compute the new routes in real-time. In addition, these systems are not only expected to provide a better user experience to passengers but also be profitable to the operator. At the same time, demand-aware bus transit system needs to cater high demands. Although, the proposed heuristic technique in Chapter 4, which generates near-optimal routes in real-time can be used for accelerating flexible route generation of demand-aware bus transit system, the runtime increases for highly complex problems, affecting responsiveness. Therefore, the need to propose a strategy that can segment the bus transit network to ensure responsiveness by enabling parallel processing of the proposed heuristic technique is identified.

Segmenting the bus transit network has to be performed in an intelligent manner such that optimality of routes is not compromised. However, existing works on demand-aware transit, segment the transit network using the proximity of the bus

transit network such as grouping nearby bus stops. Even though, runtime is improved, existing techniques generate sub-optimal routes [135]. Thus, this chapter proposes a directionality-centric bus transit network segmentation methodology that not only provides a significant speed-up in computation time hence improving the responsiveness but also generates near-optimal routes. The methodology generates sub-zones that consist of clusters of bus stops that share the same directionality, allocates EVs based on demand and finally relies on generating flexible routes for sub-zones using parallel computations that provide the flexible routes for the given zone.

The rest of the chapter is organized as follows. Section 5.2 discusses the methodology and the results are presented in Section 5.3. The conclusions are given in Section 5.4.¹

5.2 Methodology

Figure 5.1 Proposed Methodology for Generating Flexible Routes for a Zone

The proposed methodology shown in Figure 5.1 uses three steps to generate the flexible routes of the fleet of EV. The input to the methodology is the road network of the zone. The first step generates sub-zones by leveraging directionality & shortest-path centric

¹The work in this chapter has been published in [187] and [188]

clustering of bus stops in the zone. The second step allocates EVs to the sub-zones. The final step generates the flexible routes of the fleet of EV for sub-zones. The subsequent sections explain the methodology in detail.

5.2.1 Directionality & Shortest-path Centric Generation of Sub-Zones

Figure 5.2 Technique for Generation of Sub-Zones

The proposed technique for directionality & shortest-path centric generation of sub-zones is shown in Figure 5.2. It is assumed that all passengers travel to the transit hub (destination) in the given zone. The technique first identifies the bus stops and the destination of the zone. Next, it obtains the shortest path routes from each bus stop to the destination. The shortest path routes are obtained as a sequence of polylines. Then, a polyline tree (tree data structure) is generated by using the obtained shortest path routes of the bus stops to facilitate clustering. Next, leveraging the polyline tree,

elementary clusters are identified. Thereafter, elementary clusters are combined to generate sub-zones. The output of this step consists of bus stops within sub-zones and outlier bus stops. Outlier bus stops do not belong to a sub-zone since the shortest path route differ significantly from that of a sub-zone. This subsection describes the technique in detail.

Identification of Bus Stops and Destination

Initially, all the bus stops and the destination of the zone are identified. As mentioned, it is assumed that all passengers travel to the transit hub, which is considered as the destination. The output of this step is a set of GPS coordinates of the bus stops and the destination.

Obtaining Shortest Path Routes

The shortest path data is obtained using OSRM [173] by providing the GPS coordinates of the bus stops and the destination. The output contains the shortest path route from each bus stop to the destination. In addition, the distance matrix, which gives the shortest path distances between all the bus stops and the destination is also obtained. Information of the shortest path route is represented by a sequence of encoded polylines. In general, an encoded polyline represents a route segment which is constructed by connecting multiple waypoints. This is explained using the example given in Figure 5.3. Let, the bus stop (origin) and destination GPS co-ordinates (latitude, longitude) be $(1.344260, 103.703293)$, $(1.337736, 103.697289)$ respectively. The points are shown using green B1 marker and the blue flag respectively. The shortest path route obtained by OSRM for the given coordinates is shown in Figure 5.3 using the blue line. The junctions (waypoints) along the path are marked in red markers.

The sequence of polylines for the respective route is given in Table 5.1. Each encoded polyline represents a segment of the route. For example, polyline 1 represents the route segment starting from the bus stop to the first junction marked in Figure 5.3.

Figure 5.3 Shortest Path Route Obtained from OSRM

Table 5.1 Encoded Polyline Sequence Obtained from OSRM

Polyline Number	Encoded Polyline using [189]
1	<i>upeGsp}wRg@T_B'AaAd@EDMH</i>
2	<i>sweGel}wRMVKVEL?^Xp@~AxDd@hAd@dABH FJx@rArIhLpA~Aj@v@l@x@HJf@x@v@ CvB'@ \</i>
3	<i>{vdGm~{wRt@HHBRBVAb@Q'Ae@hAc@rBi@ A]</i>
4	<i>mgdGac wREUaiF</i>
5	<i>ugdGak wR</i>

This is shown in Figure 5.4a. Similarly, polyline 2 represents the next segment of the route continuing from the first junction to the second junction as shown in Figure 5.4b. The remaining route segments (polyline 3 - 5) can also be identified similarly by tracing the route along the shortest path. Further, we observe that the length of the route segment is proportional to the number of encoded characters in the polyline (the length of the polyline). For example, polyline 2 has the highest number of characters, which is the longest segment as shown in Figure 5.4b. This property of polylines is exploited to generate the clusters.

Generation of the Polyline Tree

After obtaining the shortest path routes from all the bus stops to the destination as a collection of polyline sequences, a tree is constructed using the similarities of polylines. The tree facilitates clustering by arranging the bus stops in an organized structure based on the similarity of the shortest path routes. In the polyline tree, bus stops, junctions

(a) Polyline 1 (Bus Stop to First Junction)

(b) Polyline 2 (First Junction to Second Junction)

(c) Polyline 3 (Second Junction to Third Junction)

(d) Polyline 4 (Third Junction to Destination)

(e) Polyline 5 (Destination)

Figure 5.4 Route Segments Representing Polylines in Table 5.1

and the destination are represented using nodes and each polyline is represented by an edge. The root, branch and leaf nodes of the tree represent the destination, junctions and bus stops respectively. The edges starting from leaf nodes represent polylines (route segments) starting at bus stops, while edges at branch nodes represent the polylines starting from junctions. Therefore, traversing the tree from a leaf node to the root provides the polyline sequence of the shortest path route from the bus stop to the destination. Thus, it can be seen that a group of nodes which have the same parent will have an overlapped sequence of nodes.

Figure 5.5 Shortest Path Routes from Bus Stops B1, B2 and B3 to the Destination

(a) Polyline Tree of the Shortest Path Routes

(b) Generated Tree

Figure 5.6 Polyline Tree for Example in Figure 5.5

This is explained using the example shown in Fig. 5.5. Here, three bus stops (green drop) B1, B2 and B3, four junctions (red ball) J1, J2, J3 and J4, destination (blue flag) and the shortest path routes from each bus stop are shown. The nodes along the shortest path routes are marked in Fig. 5.6a. The generated polyline tree for this scenario is shown Fig. 5.6b. In this polyline tree, the destination is node 1, while the bus stops B1, B2 and B3 are represented by node 6, 7 and 8 respectively. The junctions J1, J2, J3 and J4 are represented by node 5, 4, 3 and 2 respectively. The polyline sequence of the shortest path routes from each bus stop can be shown as set of nodes; Bus Stop B1: 6,4,3,2,1; Bus Stop B2: 7,4,3,2,1; and Bus Stop B3: 8,5,3,2,1.

Identification of Elementary Clusters

Figure 5.7 Example of Elementary Clusters

The next step is to identify elementary clusters by leveraging the generated polyline tree. Elementary clusters contain a group of bus stops with similar shortest path routes to the destination except for the first polyline, which is the polyline starting from the bus stop. Assume that the bus transit network consists of only 6 bus stops as shown in Figure 5.7. Based on the given definition, the 6 bus stops can be classified into 3 elementary clusters as shown in Figure 5.7a. However, an elementary cluster where the shortest path route does not overlap with any other shortest path route of another

elementary cluster is specifically termed as a base elementary cluster. For example, elementary clusters 1 and 2 in Figure 5.7a are base elementary clusters. This is shown in Figure 5.7b. The shortest path route of bus stops in the elementary cluster 3 falls along the path of the bus stops in the base elementary clusters (elementary cluster 1 and 2). Therefore, EVs travelling from a base elementary cluster can pick-up the passengers in the elementary clusters along the shortest path route. This property is leveraged when combining clusters. The technique of identifying base elementary clusters and elementary clusters from the polyline tree is given in Algorithm 3. Variables used in Algorithm 3 are summarized in Table 5.2. Objects and associated functions used in Algorithm 3 are introduced in Table 5.3.

Table 5.2 Variables used in Algorithm 3

Term	Description
Q	Queue of nodes to process (initially empty)
B	List of junctions (initially empty)
s	new base elementary cluster
C	List of elementary clusters (initially empty)
C_i	Elementary cluster in the i^{th} index of C
T	List of leaf nodes (initially empty)
D	Root node of the polyline tree
W	List of child nodes (initially empty)
W_i	Child node in the i^{th} index of W
L	All leaf nodes

Algorithm 3 is explained using the example scenario shown in Figure 5.8a. This shows 5 bus stops ($B1 - B5$), 2 junctions ($J1, J2$) which connects the remaining bus stops in the vicinity in the road network. The corresponding polyline tree is given in Figure 5.8b. The polyline tree is traversed using a breadth-first search starting from the root node (denoted by D in Figure 5.8b). Initially, an empty queue (Q) is defined and the root node of the polyline tree is enqueued to Q (line 2). Then, while Q is not empty, a node (n) is dequeued from Q (line 5) and the child nodes of n are put into the list W (line 6). Next, the child nodes in W are sorted in the descending order of polyline length (line 7). After sorting, the first node (W_0) in W is the node with the polyline that ends farthest away from the polyline represented by the parent node (n). While W is not empty, in each iteration, the first element (W_0) of W is removed (line 9

Table 5.3 Objects and Functions of Algorithm 3

Node (n)

Function	Description
$n.getChildren()$	returns the list of child nodes of the node n
$n.getMatchScore(n_2)$	returns the matching score between the two nodes n and n_2

Cluster (c)

Function	Description
$c.getMatchScore(c_2)$	returns the matching score between the two clusters c and c_2
$c.addNode(n)$	adds node n to the cluster c
$c.isBase()$	returns true, if the cluster c is a base elementary cluster, else returns false
$c.setBase(T/F)$	set an elementary cluster c as a base elementary cluster (true) or elementary cluster (false)
$c.setListOfJunctions(b)$	sets the list of nodes, b , as the list of junctions of the cluster c
$c.getListofJunctions()$	returns the list of junctions of the cluster c
$c.setNewBaseCluster(c_s)$	sets new base elementary cluster c_s of elementary cluster c
$c.getNewBaseCluster()$	returns the new base elementary cluster of cluster c

List (h)

Function	Description
$h.add(m)$	add the element m to the end of the list h
$h.remove(i)$	removes the element at index i from the list h
$h.get(i)$	returns the element at index i in the list h
$h.clear()$	removes all the elements from the list h

Queue (q)

Function	Description
$q.enqueue(m)$	adds the element, m , to the end of the queue q
$q.dequeue()$	returns the element that is in front of the queue and removes that element from the queue

Algorithm 3 Generation of Elementary Clusters**Input:** Polyline Tree (T_p)**Output:** List of Elementary Clusters (C)

```

1: Execution:
2:  $Q.enqueue(R)$ 
3:  $s = null$ 
4: while  $Q \neq \emptyset$  do
5:    $n = Q.dequeue()$ 
6:    $W = n.getChildren()$ 
7:   Sort the nodes in  $W$  in the descending order of polyline length
8:   while  $W \neq \emptyset$  do
9:      $W_0 = W.get(0)$ 
10:     $W.remove(0)$ 
11:    if  $W_0 \in L$  then
12:      make a new cluster:  $c_{base}$ 
13:       $c_{base}.addNode(W_0)$ 
14:      for  $W_i \in W | i \neq 0$  do
15:         $m = W_0.getMatchScore(W_i)$ 
16:        if  $m \geq P_n$  then
17:          if  $W_i \in L$  then
18:             $c_{base}.addNode(W_i)$ 
19:             $W.remove(i)$ 
20:          else
21:            break
22:          else
23:            continue
24:       $s = c_{base}$ 
25:       $C.add(c_{base})$ 
26:    else
27:       $Q.enqueue(W_0)$ 
28:       $B.add(W_0)$ 
29:      for  $W_i \in W | i \neq 0$  do
30:         $m = W_0.getMatchScore(W_i)$ 
31:        if  $m \geq P_n$  then
32:           $W.remove(i)$ 
33:          if  $W_i \in L$  then
34:             $T.add(W_i)$ 
35:          else
36:             $Q.enqueue(W_i)$ 
37:          if  $T \neq \emptyset$  then
38:            make a new cluster  $\rightarrow c_{ele}$ 
39:            add all the nodes in  $T$  to  $c_{ele}$ 
40:             $c_{ele}.setListOfJunctions(B)$ 
41:             $c_{ele}.setBase(false)$ 
42:            if  $s \neq null$  then
43:               $c_{ele}.setNewBaseCluster(s)$ 

```

```

44:          $C.add(c_{ele})$ 
45:          $T.clear()$ 
46:          $B.add(W_i)$ 
47:     else
48:         continue
49: else
50:     continue
51: if  $T \neq \emptyset$  then
52:     make a new cluster  $\rightarrow c_{ele}$ 
53:     add all the nodes in  $T$  to  $c_{ele}$ 
54:      $c_{ele}.setListofJunctions(B)$ 
55:      $c_{ele}.setBase(false)$ 
56:     if  $s \neq null$  then
57:          $c_{ele}.setNewBaseCluster(s)$ 
58:      $C.add(c_{ele})$ 
59:      $T.clear()$ 
60:  $s = null$ 
61:  $B.clear()$ 

```

(a) Network

(b) Corresponding Polyline Tree

Figure 5.8 Example Network for Algorithm 3

- 10) and either a base elementary cluster or an elementary cluster is initialized. For the example given in Figure 5.8a, W_0 will be a leaf node (bus stop B1). Therefore, the case where W_0 is a leaf node is explained first followed by the case where W_0 is a non-leaf node.

Case 1: W_0 is a leaf node (Bus Stop) (line 11 - 25). If W_0 is a leaf node, a new cluster (c_{base}) is initialized and W_0 is added to c_{base} (line 12 - 13). Then, starting from the 1st index, all remaining nodes are read iteratively from W (let a removed node be denoted by W_i). Here, node W_i is determined to be combined with W_0 in order to form a base elementary cluster if the character match between the polyline sequences representing the shortest path route to the destination from the two nodes are greater than or equal to the $P_n\%$ (*similarityPercentageForNodes*) (line 16). The value for P_n (80%) was determined through experimentation. The relatively high value ensures that the bus stops in base elementary clusters are physically nearby in the road network. However, it should be noted that only leaf nodes are added to the newly created cluster, c_{base} (line 17 - 19). Otherwise, the algorithm moves on to the next node and repeats the process until a non-leaf node with a similarity percentage greater than or equal to $P_n\%$ is encountered (non-leaf node $J1$). In Figure 5.8b, leaf nodes $B1$ and $B2$ are added to c_{base} . Also, c_{base} is added a new base elementary cluster for the elementary clusters that will be discovered during the search process of the remaining nodes in W (line 24). New base elementary cluster functions similarly to base elementary clusters. However, new base elementary cluster is denoted separately since it is not possible to identify these clusters using the contents of the list of junctions (B), which is used to identify the base elementary clusters. Then, the new instantiated cluster (c_{base}) is added to the list of elementary clusters, C (line 25), and the algorithm moves to the next iteration of the nested while loop, starting at line 8. In the next iteration, the removed node from W at line 9 and line 10, W_0 , will be a non-leaf node, representing the junction $J1$. Therefore, the algorithm moves to case 2.

Case 2: W_0 is a non-leaf node (Junction) (line 26 - 59). If W_0 is a non-leaf node, then W_0 is enqueued to Q (line 27) and added to the list of junctions (B) (line 28). B

will be used to identify the base elementary clusters which have an overlap in the path with an elementary cluster. Then, starting from the 1st index, all remaining nodes are read from W iteratively (let a removed node be denoted by W_i). If the similarity percentage of W_i is higher than $P_n\%$, W_i is validated if it is a leaf node. If it is valid, W_i is added to T (temporary buffer of leaf nodes) in each iteration (in this example, leaf nodes $B3$ and $B4$ will be added to T in two iterations) (line 33 and line 34). If W_i is a non-leaf node (in this example, the non-leaf node representing $J2$), it is enqueued to Q (line 36) and a new elementary cluster, c_{ele} is instantiated with all the leaf nodes stored in T (line 37 - 39). At the same time, all the contents of B (in this example, $J1$) are added to the list of junctions of c_{ele} (line 40). The reason of identifying c_{ele} as an elementary cluster is that it will be placed along the shortest path route of base elementary clusters formed from the child nodes of the nodes in B . For example, bus stops $B3$ and $B4$ in c_{ele} will be along the path of all the base elementary clusters formed farther away from the destination, where the routes overlap from junction $J1$. Similarly, if a new base elementary cluster (s) exists, it is also updated (line 42 and line 43). Then, the new instantiated elementary cluster c_{ele} is added to list of elementary clusters, C (line 44). Afterwards, T is cleared (line 45) and W_i is added to the list of junctions, B (line 46, now B will contain $J1$ and $J2$). This process is repeated until the algorithm iterates over all the elements in W . Then, the algorithm checks if T is empty (line 51). T can be non-empty, if the last node removed from W is a leaf node (in this example, last node removed from W is the leaf node $B5$). Therefore, another elementary cluster is added in the same manner explained above (line 52 - 59). Then, s is set to null (line 60) and B is cleared (line 61), which indicates the end of clustering the nodes, whose parent is n . Thereafter, the next node in Q is dequeued (line 5) and the whole process is repeated until Q is empty.

Table 5.4 summarizes the clusters and the corresponding data for the example in Figure 5.8. The columns show the elementary cluster number, if the cluster is a base elementary cluster, bus stops in the elementary cluster, junctions which are in the tail of the cluster and the head of the cluster (parent node) and the corresponding base elementary clusters respectively.

Table 5.4 Identified Elementary Clusters of the Example in Figure 5.8

Elementary Cluster	Base (Y/N)	Bus Stops	Junctions (Tail)	Junction (Head)	Corresponding Base Elementary Clusters
1	Y	B1,B2	-	D	-
2	N	B3,B4	J1	D	Cluster 1
3	N	B5	J1, J2	D	Cluster 1

Directionality & Shortest-path Centric Combining of Elementary Clusters for Identification of Sub-Zones

Output of Algorithm 3 provides all the identified elementary clusters of the polyline tree with the corresponding data as shown in Table 5.4. The next step leverages this data to identify the sub-zones by directionality & shortest-path centric combining of elementary clusters. The identification of sub-zone process consists of two main functions. First, it combines the base elementary clusters followed by combining with the elementary clusters along the shortest path. As shown in Figure 5.2 this process is repeated until combining does not create any new sub-zones. The two functions are explained subsequently.

When combining elementary clusters to create sub-zones, all the base elementary clusters are first extracted from the generated list of elementary clusters. Then, the base elementary clusters are sorted by the distance from the destination such that combining can start from the farthest away base elementary cluster. However, since an elementary cluster can be a collection of leaf nodes, which represent different bus stops, a single leaf node must be selected to represent the whole elementary cluster. For this, the node which has the median distance to the destination among all the nodes of the elementary cluster is selected. Here, distance is identified by the length of the polyline sequence representing the route to the destination. Then, the algorithm tries to combine the selected base elementary cluster with the remaining base elementary clusters. The criterion for combining base elementary clusters is the similarity of the route to the destination. During the combining process, the bus stop with the median distance to the destination of the two base elementary clusters are compared

for a $P_c\%$ (*similarityPercentageForClusters*) character match in the encoded polylines. The value for P_c (70%) was determined through experimentation. If the comparison is valid, the two base elementary clusters are combined to form the initial sub-zone (G). The list of elementary clusters is also updated with the new data. The algorithm for combining base elementary clusters is given in Algorithm 4. The supplementary functions used in Algorithm 4 are given in Table 5.5.

Algorithm 4 Combining Base Elementary Clusters

Input: List of elementary clusters (C)

Output: List of updated elementary clusters (C')

```

1: Execution:
2:  $L = \text{getAllBaseElementaryClusters}(C)$ 
3: while  $L \neq \emptyset$  do
4:   sort  $L$  based on distance
5:    $L_0 = L.\text{remove}(0)$ 
6:   for  $L_i \in L | i \neq 0$  do
7:     if  $L_0.\text{getMatchScore}(L_i) \geq P_c$  then
8:       Combine  $L_i$  with  $L_0$ 
9:        $L.\text{remove}(i)$ 
10:       $G.\text{add}(L_i)$ 
11:  $C' = \text{updateElementaryClusters}(G)$ 

```

Table 5.5 Supplementary Functions for Algorithm 4

Function	Description
$\text{getAllBaseElementaryClusters}(C)$	returns a list containing all base elementary clusters in C
$\text{updateElementaryClusters}(G)$	given the initial sub-zone (G), returns the updated list of elementary clusters

Next, the formed initial sub-zone (G) is combined with the elementary clusters along the shortest path route. Here, data in Table 5.4 such as tailing junctions and head of the elementary cluster are used. Elementary clusters are first filtered using the tailing junction (head of the sub-zone should match the tailing junction of the elementary cluster). After combining the elementary cluster with the sub-zone, the corresponding data in Table 5.4 are updated. Sub-zone data are also updated with the addition of new elementary clusters. The sub-zone formation stops when there are no elementary clusters remaining to combine with it. This implies that the sub-zone

for the selected base elementary cluster has reached the destination. This will exit the execution and as mentioned earlier, the process will be repeated for a new base elementary cluster to identify a sub-zone.

When the list of base elementary clusters are explored and the combining process does not produce new sub-zones, the technique exits. However, when the sub-zone generation process exits, the list of elementary clusters in Table 5.4 may contain a few base elementary clusters, which were not combined with any sub-zone due to the similarity mismatch. These base elementary clusters are then decomposed to the individual bus stops, which are henceforth referred to as outlier bus stops. All the other bus stops in the network will be part of a sub-zone. These bus stops are henceforth referred to as bus stops within sub-zones. In the subsequent step, bus stops within sub-zones and outlier bus stops are intelligently used to minimize the number of used EVs. It should also be noted that when combining the formed sub-zone (G) with the elementary clusters, the algorithm can be modified to consider criterion such as travel time between the sub-zone and the elementary cluster to prevent combining of elementary clusters which are significantly apart in the road network. This can be helpful in scenarios where routing is performed considering the stochastic future demands.

5.2.2 Allocating EVs to Sub-Zones

Figure 5.9 Technique for Allocating EVs to Sub-Zones

The technique for allocating EVs to sub-zones is shown in Figure 5.9. The technique consists of three stages, namely associating nearest outlier bus stops to sub-zones, demand-aware allocation of EVs and workload balancing.

Associating nearest Outlier Bus Stops to Sub-Zones for Allocation of EVs

After generating the list of bus stops within sub-zones and outlier bus stops, the next step in the methodology is to associate nearest outlier bus stops to sub-zones for allocation of EVs. To identify the nearest sub-zone for each outlier bus stop, sub-zones are sorted based on the proximity. Here, proximity is calculated by using actual road network distance between the bus stop with the median distance of each sub-zone and outlier bus stop. The distance matrix obtained from OSRM is used to identify the distance. All the outlier bus stops are associated to the nearest sub-zones using this technique.

Demand-aware Allocation of EVs

The next step allocates EVs based on the demand of bus stops within sub-zones while taking into account the demand of the associated outlier bus stops. The objective of the EV allocation algorithm is to minimize the number of EVs to improve the responsiveness of the routing step. Since EVs are placed in car parks near the bus stops, the number of EVs in each sub-zone can be computed. Therefore, some sub-zones contain a sufficient number of EVs to satisfy the demand while other sub-zones require additional EVs. Thus, the EV allocation step has to optimally manage the supply and demand. For example, sub-zones with excess EVs can assign the additional EVs to other sub-zones which require EVs. However, the main problem to address in the allocation is to decide which EVs to retain to satisfy the demand of the bus stops within the sub-zone. Also, the allocation algorithm needs to take into account the demand of the associated outlier bus stops. Thus, allocation will significantly affect the quality of the results.

Initially, the number of required EVs for each sub-zone with associated outlier bus stops is computed based on the demand such that *number of passengers in bus stops within sub-zones and associated outlier bus stops* $\leq$ *no of EVs* $\times$ *EV capacity*. Then, all EVs near outlier bus stops are added to a list of EVs, *vehiclePool*. All EVs in outlier bus stops are added to the *vehiclePool* since it is preferred to satisfy the demand of outlier bus stops from the EVs of the associated sub-zones, such that the number of EVs are minimized. Next, excess EVs in each sub-zone are added to the *vehiclePool*. When selecting the excess EVs, the algorithm uses the polyline sequence of each location to find the proximity of EVs to the destination. Thus, EVs which are farthest away from the destination in each sub-zone are allocated to pick-up the passengers in the respective sub-zone. The motivation in this case is to allow the EVs to pick up a higher number of passengers from other bus stops along the route to the destination. Thereafter, the algorithm allocates EVs to sub-zones, which require additional EVs from the *vehiclePool*. The same proximity-based distance is used when allocating EVs such that EVs closest to the bus stop with the median distance of the sub-zone are allocated. At the end of allocation, the algorithm outputs the allocated EVs for bus stops within sub-zones and associated outlier bus stops.

Workload Balancing

After allocation of EV for the sub-zones, the technique balances the workloads of sub-zones including its' associated outlier bus stops in order to improve the responsiveness. For example, assume that 3 sub-zones have been identified and the number of passengers in each sub-zone including passengers in associated outlier bus stops are 20, 40 and 15 respectively. Therefore, if the routing algorithm is executed with the current workload distribution, the runtime of sub-zone 2 will be significantly high compared to others. Therefore, it is beneficial to split sub-zone 2 such that workload is balanced. The smallest sub-zone size can be used as the baseline for splitting a sub-zone. For example, a sub-zone where the number of passengers is more than twice the smallest sub-zone (sub-zone 2 has a capacity of 40, which is more than 15×2 passengers) can

be potentially split into a few workload balanced sub-zones. Therefore, depending on the capacity of the EV, the workload can be split into relatively balanced sub-zones. However, since all passengers share a common shortest path to the destination with minimal detours to passengers at the associated outlier bus stops, splitting passengers of a sub-zone including its' associated outlier bus stops has minimal impact on the quality of solutions.

5.2.3 Flexible Routing for Sub-Zones

After allocating EVs to sub-zones, the single-objective GA proposed in Chapter 4 is executed in each sub-zone to find flexible routes for the fleet of EVs. At the end of this step, valid routes that do not violate any of the constraints can be obtained. However, if the responsiveness of the system is not violated, a further optimization can be performed using the dual-objective GA proposed in Chapter 4 using the obtained solution. The technique is shown in Figure 5.10.

Figure 5.10 Technique for Flexible Routing for Sub-Zones

Sub-Zone Routing

At the end of allocating EVs to sub-zones, each sub-zone contains a set of EVs that need to be routed to pick-up passengers such that all constraints are satisfied. To this end, the proposed single-objective GA in Chapter 4 is used with the objective of

minimizing VMT. As shown in Figure 5.10, the algorithm is executed in parallel in each sub-zone, which significantly reduces the runtime. Further, due to the directionality & shortest-path centric generation of sub-zones, the degradation of solution quality is significantly minimized.

Generation of Sub-Zone Routes

The routes generated for the EVs from sub-zone routing is combined to generate the flexible routes for the zone. The solution represents a valid set of routes to pick-up passengers which maximizes the operator profitability (reduce VMT). However, it should be noted that the passenger travel time constraint ensures that the level of user experience is maintained. Noting that the proposed technique can be further extended without violating the responsiveness of the system (given a runtime constraint), the obtained routes of each sub-zone can be used as an initial solution in a dual-objective GA to generate routes that not only strives to improve operator profitability but also to enhance the user experience by minimizing passenger travel time. Here, the dual-objective GA is limited to run only for the remaining time limit such that responsiveness is not violated. However, as shown in Section 4.3, the dual-objective GA converges significantly faster due to the quality of the initial routes. As mentioned in Chapter 4, the output contains multiple pareto optimal design points. Flexible routes for the zone can be generated by utilizing all the pareto optimal design points of each sub-zone. The operator can select the set of routes based on the requirements for real-world deployment.

5.3 Performance Evaluation

This section presents the experimental results of the proposed methodology for directionality-centric segmentation of bus transit network. The selected zone for the experiments is shown in Figure 5.11. Bus stops in the selected zone are shown

Table 5.6 Values of the Parameters

Parameter	Value
EV Capacity	8
EV Driving Range	30 km
Tolerance Factor	4

Test Cases

Test case parameters given in Table 5.7 specify the number of passengers and available EVs in the fleet. In each test case, passengers and EV are randomly distributed in the bus stops in the zone shown in Figure 5.11. The number of passengers in the experiments are chosen based on the real demand observed on fixed route transit services using historical data. In test cases 1 & 2 the average utilization of the fleet is less than 50%. Test cases 3 - 5 reflect scenarios with mid utilization. It should be noted that compared to test cases in Chapter 4, the total demand and supply considered in these test cases are significantly higher.

Table 5.7 Test Cases

Test Case	Demand (Passengers)	Supply (EV)
1	100	35
2	200	40
3	250	50
4	300	60
5	350	65

5.3.2 Experiments

Comparison with Existing Clustering Algorithms

In this experiment, the proposed sub-zone generation technique is compared with the k-means [190] and k-medoids [191] algorithms. Therefore, in the experiments the proposed directionality & shortest-path centric clustering technique is replaced by the k-means and the k-medoids algorithms, while the rest of the methodology is unchanged.

(a) Example 1

(b) Example 2

(c) Example 3

Figure 5.12 Comparison of VMT of the Fleet with K-Means and K-Medoids Algorithms

Here, the VMT of the routes generated by the three techniques after sub-zone routing is compared. All techniques use a population size of 50 and 100 iterations for the GA. However, since k-means and k-medoids algorithms require the value of k (the number of sub-zones) to be specified a-priori, 3 instances representing values of 5, 10 and 20 of k are used. Results for 3 selected test cases, namely test case 2, 3 and 4 in Table 5.7 are given in Figures 5.12a - 5.12c respectively. The vertical axis in each graph gives the VMT while the horizontal axis shows the value of k . The black and grey columns represent the results for k-means and k-medoids respectively. Since the results of the proposed method do not depend on the value of k , it is shown using an orange line with triangle markers. It is observed that in all instances, the proposed methodology gives a low VMT compared to k-medoids algorithm. However, in the case of k-means, in two instances the VMT of the proposed method is marginally higher (the VMT obtained using k-means with $k=5$ for test case 1 and 2 is 0.25% and 0.95% higher respectively). However, finding the best value for k is practically infeasible. Therefore, k-means and k-medoids algorithms cannot be relied upon to produce better results consistently. On average, the proposed sub-zone generation technique generates routes with a 6% reduction in VMT compared to the other algorithms.

Comparison of the Pareto Front

The Pareto front obtained from the proposed methodology is compared with the heuristic technique without sub-zoning proposed in Chapter 4. The population size of the GA for both techniques is set to 50. The proposed methodology consists of a two-step route generation technique, where the initial routes are generated using the single-objective GA that is further optimized using the dual-objective GA. In the single-objective GA, the number of iterations is set to 100. However, as explained in Chapter 4, if the solutions are not improved after 5 continuous iterations, the single-objective GA terminates, and the dual-objective GA begins operation. For the dual-objective GA, a limitation on the number of iterations is not set. Similarly, the technique without sub-zoning performed using the dual-objective GA has no

(a) Test Case 1

(b) Test Case 2

(c) Test Case 3

the Pareto front obtained using the proposed method is better in comparison to the technique without sub-zoning. This can be attributed to the segmentation of the large design space that enables to explore regions, which is not feasible in the technique without sub-zoning for the given runtime constraint. In Figure 5.13a and 5.13e, a few design points of the objective space obtained using the proposed method is marginally inferior compared to the technique without sub-zoning. However, the diversity of the design points indicate that the proposed method has been able to find solutions in a region that the technique without sub-zoning has not been able to traverse.

Runtime

Figure 5.14 Runtime Comparison with the Technique Without Sub-Zoning

This experiment is conducted to measure the speed-up that can be obtained from the proposed methodology. For this, the proposed methodology is executed with a population size of 50 and 100 iterations. Thus, both GAs of the proposed methodology is executed with the above parameters. For the heuristic technique without sub-zoning, the same population size of 50 is used. Then, the heuristic technique without sub-zoning is executed and the obtained Pareto front and runtime in each iteration is logged. Next, the pareto front obtained from the proposed methodology is compared with the Pareto fronts obtained in each iteration from the heuristic technique without sub-zoning and the one which has the minimum generational distance is selected for

runtime comparison. The runtime of the two techniques are then used to compute the speed-up. The corresponding runtime of the test cases are shown in Figure 5.14. Here, the horizontal axis shows the test case number and the vertical axis shows the runtime in seconds. The blue and orange columns represent the technique without sub-zoning and the proposed methodology respectively. It shows the linear trend in increase of the runtime for both scenarios. However, the proposed methodology generates results 7.8x faster (87% improvement in runtime) compared to the technique without sub-zoning. This highlights the fast convergence of the proposed sub-zoning methodology, which provides comparable results significantly faster.

Impact of Balancing Workloads

Figure 5.15 Runtime Variation of Flexible Routing for Sub-Zones Without Workload Balancing

The methodology proposed in Section 5.2 classifies the bus stops into sub-zones using the directionality along the road network. As shown in the above experiments, this provides a significant speed-up for computations. Further, the methodology balances the workloads of each sub-zone. The benefit of balancing workloads in sub-zones is shown by executing test case 2 in Table 5.7 by removing the workload balancing step. The GA is implemented with parameters of 50 and 100. The distribution of runtime of the routing algorithm for each sub-zone is shown in Figure 5.15. Here, the

horizontal axis shows the runtime in seconds and the vertical axis shows the sub-zone number. This shows that sub-zone 1 has a significantly higher runtime (53+ seconds) compared to other sub-zones, which can be reduced in the proposed methodology by workload balancing (reduced to 25+ seconds using workload balancing). Observing the passenger distribution in the sub-zones revealed that sub-zone 1 contained 48 passengers (out of 200), which results in a higher runtime when workloads are not balanced. However, when the workload is balanced the runtime can be reduced as given above.

Sporadic Demands

In the above examples, passenger demand is assumed to be normally distributed in the zone, which represents a typical scenario. However, special occasions such as public events can lead to sporadic distributions. The proposed methodology should be able to provide a similar performance for not only normal demand distributions but also sporadic demand distributions. This experiment is conducted to analyse the performance of the proposed methodology in handling sporadic demand distributions. A sample heat map of the two demand distributions are given in Figure 5.16. Figure 5.16a shows a normal distribution while Figure 5.16b shows a sporadic distribution. In both scenarios 100 passengers are distributed in the zone and 20 EV with a capacity of 8 is considered. Number of passengers in each bus stop is marked in Figure 5.16. The colour code of the markers represents passenger density. Therefore, the sporadic distribution of Figure 5.16b is reflected in the heat map by the increase in the red markers compared to Figure 5.16a. It should be noted that applying the proposed methodology on the two scenarios generate 12 and 7 sub-zones respectively. The lower number of sub-zones for the scenario shown in Figure 5.16b is due to the higher number of empty bus stops, which are not required to be visited by the vehicles.

The technique used in the experiments for measuring the runtime, which matches the two Pareto fronts obtained from the proposed methodology and the heuristic routing technique without sub-zoning is used in this experiment. The obtained runtime values

(a) Normal

(b) Sporadic

Figure 5.16 Example of Passenger Demand at Bus Stops

for the scenario in Figure 5.16b for the heuristic routing technique without sub-zoning and the proposed methodology is 119 seconds and 15 seconds respectively. The runtime shows a 7.9x speed-up, which is similar to the value observed in Figure 5.14. This confirms that the proposed strategy can not only perform well for normally distributed demands but also for sporadic distributions.

5.4 Summary

This chapter proposed a methodology for directionality-centric bus transit network segmentation to speed-up computation in a demand-aware bus transit system. The proposed methodology generates directionality & shortest-path centric sub-zones, which consists of bus stops that share the same shortest path route. In addition, outlier bus stops, where the shortest path route deviates from other bus stops within sub-zones are also identified. Then, nearest outlier bus stops are associated to the sub-zones for allocation of EVs. Allocating to the nearest sub-zone minimizes detours in the EV routes. Next, taking into account the real-time demand of bus stops within sub-zones and associated outlier bus stops, the minimum number of EVs for each sub-zone are allocated. Allocating the minimum number of EVs reduces the complexity of computations and hence the runtime. Thereafter, sub-zones and the associated outlier bus stops are segmented into workload balanced sub-zones to improve responsiveness. Finally, flexible routing for sub-zones is performed using the heuristic techniques proposed in Chapter 4.

The proposed methodology has been evaluated using test cases generated on a real road network encompassing the Nanyang Technological University campus, which is the largest university campus in Singapore. Experiments conducted based on openly available public transit demand data show that the proposed methodology can generate routes that improve the VMT of the fleet by 6% compared to two widely-used unsupervised learning algorithms. At the same time, proposed methodology achieves over a 7x speed-up in comparison to the heuristic technique proposed in Chapter 4

without compromising solution quality. Experiments also confirm the robustness of the proposed methodology to handle not only normal demand distributions but also sporadic demand distributions. This confirms the suitability of the proposed bus transit network segmentation methodology for generating rapid, high-quality solutions in scenarios with high demand while ensuring responsiveness of the demand-aware bus transit system.

The methodology proposed in this chapter can be used to rapidly generate flexible routes for a demand-aware bus transit system operating in a geographical zone when compared to a heuristic technique, which uses the entire bus transit network without segmentation. However, demand-aware bus transit system need to be deployed in large geographical areas such as cities, which have complex road networks and many transit hubs. This necessitates the development of a technique to systematically decompose a city into zones, where the proposed bus transit network segmentation methodology can be implemented in parallel. Hence, a technique for adopting demand-aware bus transit system for large scale deployment that leverages the road network topology, bus transit system and workload balancing techniques is presented in the next chapter.

Chapter 6

Adopting Demand-aware Bus Transit System for Large Scale Deployment

6.1 Introduction

Chapter 5 presented a directionality-centric bus transit network segmentation technique to speed-up computation for deployment of the proposed demand-aware bus transit system in a geographical zone surrounding a transit hub. The sub-zones generated by the presented technique groups bus stops, which share the same shortest path route to the transit hub with minimal detours. As a result, the routing algorithm, which was implemented in parallel in each sub-zone was able to generate near-optimal routes in a zone with a significant benefit in runtime. However, the demand-aware bus transit system needs to be deployed in large geographical regions, such as a city that consists of multiple transit hubs with large workloads. This requires identifying the service area of each transit hub. In addition, efficient partitioning techniques are required that can generate independent zones based on the road network topology, where the proposed bus transit network segmentation technique can be applied in parallel to generate solutions rapidly. Workload balancing techniques also need to be incorporated into the partitioning technique to ensure responsiveness.

This chapter proposes a methodology for adopting demand-aware bus transit system for large scale deployment. The proposed methodology can systematically identify independent transit hub regions from a large geographical region by leveraging hub & spoke topology of public transit network and the feeder bus services. Then, a partitioning algorithm that incorporates both the road network topology and workload balancing, segments a transit hub region into multiple zones. This facilitates to implement the techniques proposed in Chapter 5 in the zones to generate high-quality solutions rapidly. The chapter also proposes techniques for demand-aware scheduling of point-to-point connections provided by the trunk bus services, which is facilitated in the existing public transit network as a supplementary service to rapid transit.

The rest of the chapter is organized as follows. Section 6.2 discusses the proposed methodology and the results are presented in Section 6.3. Section 6.4 proposes a methodology for demand-aware scheduling of point-to-point connections. The conclusions are presented in Section 6.5.

6.2 Methodology

The method proposed in this chapter leverages the hub & spoke topology of public transit network and existing bus transit system to systematically decompose a large geographical region into zones, where the technique proposed in Chapter 5 can be implemented in parallel to generate the routes for the fleet of EVs. In general, a large geographical region encompasses multiple transit hubs. The large geographical region can be divided into multiple regions based on the location of transit hubs (transit hub regions). Further, the transit hub regions can be segmented into small regions based on the locations of bus stops and the workload (zones). This enables to implement the technique in Chapter 5 in each zone in parallel. This strategy is exploited in this chapter to decompose the large geographical region.

Table 6.1 summarizes the definitions of the geographical regions used in this chapter. A large geographical region is an area which contains multiple transit hubs.

Table 6.1 Definition of Regions

Area	Description
Large Geographical Region	An area serviced by multiple transit hubs identified by the transport planner as a potential area for the deployment of the demand-aware bus transit system.
Transit Hub Region	An independent region containing one transit hub. Transit hub regions can have overlapping geographical boundaries. However, demand for each transit hub must be identified distinctively. The collection of transit hub regions form the large geographical region.
Zone	A geographical area identified by segmenting the transit hub region. The collection of zones form the transit hub region.
Sub-Zone	A group of bus stops which share a similar shortest path route to the destination. The collection of all the sub-zones encompass the bus stops in the zone. The techniques proposed in Chapter 5 are used to identify sub-zones and to compute routes.

Typically, it can be a city in a country or a city state. An example large geographical region is shown in Figure 6.1, which shows the map of Singapore. Although, Singapore is used as an example, all techniques proposed in this chapter can be deployed in other cities as well. The transit hubs are used to divide the large geographical region into independent transit hub regions. Figure 6.2 shows the transit hubs in Singapore marked in black circles. The transit hub regions can be identified by leveraging the feeder bus services. Two adjacent transit hubs enclosed in the red square in Figure 6.2 are used to show the transit hub regions. Figure 6.3 shows the distinct transit hub regions with minimal overlaps of the two selected transit hubs. Figure 6.3 shows the two transit hubs in blue and red ball markers and the corresponding regions shaded in the same colour. It should be noted that the techniques proposed in this thesis can distinctively identify the demand for each transit hub region, which facilitates overlapping of the transit hub regions. Zones are identified by segmenting the transit hub region. Figure 6.4 shows a sample segmentation of four zones of the transit hub region 1 of Figure 6.3. The number of zones of a transit hub region will depend on the

demand distribution and hence will be determined only at execution time. Sub-zones of a zone can be identified using the techniques proposed in Chapter 5.

Figure 6.1 Example City

Figure 6.2 Locations of Transit Hubs

Figure 6.3 Distinct Transit Hub Regions with Minimal Overlaps

Figure 6.4 Segmented Zones in a Transit Hub Region

The techniques of identifying the transit hub regions and zones are explained in detail in subsections 6.2.2 and 6.2.3 respectively. As mentioned, the transit hub regions are identified by leveraging the feeder bus services. Therefore, a classification of bus transit system is presented in subsection 6.2.1 prior to explaining the techniques on identifying transit hub regions and segmenting transit hub regions into zones.

6.2.1 Classification of Bus Transit System

In the current public transit network topology, a bus service originates at a bus interchange (transit hub) or a terminal station. As mentioned in Section 3.2, all the bus services in the bus transit system can be classified into feeder and trunk services. A feeder service provides connectivity to an area surrounding the bus interchange following a route as a loop. Feeder services facilitate short distance trips, which connect passengers to/from the transit hub. In contrast, a trunk service provides long distance point-to-point connectivity. In a trunk service, the origin and destination are different bus interchanges or terminal stations. However, since the service runs both-ways, a trunk service can be typically identified as a bi-directional single bus service, even though, a few operate as a looping service in a single direction. Figure 6.5 illustrates the routes of the different types of bus services.

Figure 6.5 Example of Bus Services

Figure 6.5 shows two transit hubs and a terminal station of bus services in blue, red ball markers and purple square marker respectively. The origin transit hub of all the bus services is shown in blue ball marker. The route of the feeder bus service is shown in blue. The service originates and terminates at the same transit hub while providing short distance connectivity to passengers surrounding the transit hub. The

route of a typical trunk bus service is shown in green and brown, which represent the two directions of the bi-directional service. The destination terminal station of the service is shown with the purple marker. It can be seen that the trunk bus service provides long distance point-to-point connectivity compared to the feeder bus service. The route shown in red of Figure 6.5, is an example of a trunk bus service that operates as a single directional service, which loops at a transit hub. Although, such services are commonly not encountered, they are also classified as trunk bus services with point-to-point connectivity in this thesis.

As the public transit network is designed based on a hub & spoke topology, feeder bus services provide connectivity to passengers surrounding transit hubs operating as a loop service. Feeder bus services are generally short distance services. Also, multiple feeder bus services can overlap in parts of a route. This is illustrated in Figure 6.6a, which shows several feeder bus services operating at Boon Lay transit hub in Singapore. Location of the transit hub is marked with the blue ball marker and coloured routes show the feeder bus services. This also shows two trunk bus services in black. Figure 6.6a confirms that feeder bus services have short distance routes and also that segments of the routes overlap each other. This is further confirmed in Figure 6.6b, which shows the Yishun transit hub in Singapore. Similar to Figure 6.6a, the purple ball marker shows the transit hub and the coloured and black paths show the feeder bus services and trunk bus services respectively. The diagram shows that the feeder bus services provides nearly 360° coverage surrounding the transit hub. Therefore, the proposed technique focuses on the feeder bus services for identification of transit hub regions. The proposed technique of identifying transit hub regions is explained in detail subsection 6.2.2. At the same time, a technique for improving the waiting time of passengers using point-to-point trunk bus services that provide long distance connectivity is proposed in Section 6.4. A method for integrating the two techniques is also presented.

(a) Boon Lay Transit Hub

(b) Yishun Transit Hub

Figure 6.6 Feeder Bus Services of Transit Hubs

6.2.2 Identifying Transit Hub Region using Feeder Bus Services

Section 6.2.1 presented the classification of bus services and also highlighted the properties of the feeder bus services. This subsection describes a technique for identifying the independent transit hub regions using the feeder bus services, which enables to systematically decompose the large geographical region.

As defined in Table 6.1, a transit hub region is an independent region containing one transit hub. Therefore, the transit hub region boundaries need to be first identified for separating the large geographical region. After the classification of bus services as mentioned in Section 6.2.1, all feeder bus services originating at the transit hub are used to generate the transit hub region boundary. Figure 6.7 shows the transit hub regions of two adjacent transit hubs with the corresponding feeder bus services. Here, the two transit hubs are shown in blue and red ball markers. The transit hub regions are shaded in blue and yellow colours respectively. As seen in Figure 6.7, the transit hub regions can contain overlapped boundaries. The techniques of identifying the transit hub region boundary and facilitation of overlapped boundaries are explained using Figure 6.8 and Figure 6.9 respectively.

Figure 6.7 Example of Transit Hub Regions with the Corresponding Feeder Bus Services

(a) Service Area of Feeder Bus 1

(b) Service Area of Feeder Bus 2

Figure 6.8 Example for Identifying Transit Hub Region Boundary

As shown in Figure 6.7, the transit hub regions enclose the routes covered by all the feeder bus services originating at the transit hub. The boundary of a single bus service is identified by the major roads that are serviced by a feeder bus. The enclosed area of the feeder bus service route is considered as the service area. Figure 6.8 shows the technique of identifying the transit hub region boundary using individual service areas of feeder bus services. Figure 6.8a shows a part of the boundary of Boon Lay transit hub region and the corresponding service area of a feeder bus service in

the region. The transit hub is marked with the blue ball marker and the feeder bus service is shown in blue colour. The transit hub region boundary is enclosed in blue shading and the service area of the feeder bus service is shown using a dotted black line. Figure 6.8b shows the service area of another feeder bus service, which shares segments of the route with the feeder bus service of Figure 6.8a. After identifying the individual service areas, they are combined to generate the transit hub region boundary such that the outermost service area of all the feeder bus services encompass the transit hub region boundary. This can be visualized by combining the service area of the two feeder bus services in Figure 6.8a and Figure 6.8b which generates a segment of the transit hub region boundary.

Figure 6.9 Overlapped Boundaries of Adjacent Transit Hub Regions

Figure 6.7 also shows that adjacent transit hub regions can have overlapping routes if the transit hubs are located in close proximity. In such a scenario, the boundaries of transit hub regions can geographically overlap. Figure 6.9 shows the overlapped boundaries of the adjacent transit hub regions shown in the dotted circle in Figure 6.7. The overlapped region is shown in the green shading. Passengers requesting for rides using a mobile phone application indicate the destination transit hub. Therefore, the demand for each transit hub region can be distinctively identified. As a result,

overlapping of boundaries of transit hub regions is facilitated in the proposed demand-aware bus transit system.

While identifying the transit hub regions, the major roads in the network are used to demarcate the boundary of a region. It should be noted that the technique of dividing a geographical area along the major roads is a common technique used by many works such as Quadrifoglio et al. [129], Shen et al. [135] and Pelzer et al. [136]. However, in this case, since the transit hub regions enclose only the routes covered by feeder bus services, several bus stops that are serviced by only the trunk bus services are not facilitated from the proposed demand-aware bus transit system. These bus stops can be within the service area of the transit hub region as well as outside the region. Although, this represents only a limited quantity of public transit users, commuters using trunk bus services can also benefit from a demand-aware bus transit system. To this end, Section 6.4 proposes a technique for demand-aware scheduling of point-to-point connections. However, in Section 6.2, focus will be only on the transit hub regions covered by the feeder bus services.

After identifying the transit hub regions, they are considered as independent entities. Transit hub regions need to be recomputed only if a new feeder bus service is introduced to the bus transit system. Therefore, identification of transit hub regions is a one-time offline process and hence does not impact the runtime for computation of routes. As a result, the complexity of route generation for large scale deployment of demand-aware bus transit system can be significantly reduced. Next, the proposed technique for segmenting a transit hub region into zones is presented.

6.2.3 Segmenting Transit Hub Region into Zones

Identifying the independent transit hub regions enables to reduce the complexity of route generation significantly. However, as shown in Section 5.3, implementing the bus transit network segmentation technique in a transit hub region will consume a considerable time and hence impacts the responsiveness of the demand-aware bus

transit system. Therefore, it is imperative that the transit hub regions are segmented into zones, which are not only small in size but also balanced in terms of workload to manage the complexity and responsiveness. This section explains the technique of segmenting a transit hub region into zones.

The identified transit hub regions are further subdivided into zones based on the location of bus stops and the demand. Dividing based on the location of bus stops help to implement parallel computations which are not inter-dependent. Considering the demand helps to balance the workloads in each zone such that runtime of the computations do not vary significantly. To this end, all the bus stops in a transit hub region are converted to a tree data structure based on the shortest paths. The technique for generating the tree is described in Section 5.2.1. Here, it is summarized for clarity. Figure 6.10 shows three bus stops (B1, B2 and B3) and the shortest path routes to the destination. The generated polyline tree is given in Figure 6.11. In this polyline tree, the destination is marked as node 1, while the bus stops B1, B2 and B3 are represented by node 6, 7 and 8 respectively. The junctions J1, J2, J3 and J4 are represented by node 5, 4, 3 and 2 respectively. Similarly, all the bus stops in the transit hub region can be converted to a tree structure. The initial tree data structure is generated offline using all the bus stops in the zone. The demand in all bus stops (workload) can then be incorporated into the leaf nodes of the tree and aggregated to the parent level at runtime. For example, if the demand at bus stops B1, B2 and B3 is 4, 2 and 5 respectively, the workload of node 4 in the tree is 6 (4 in B1 + 2 in B2).

Figure 6.10 Selected Bus Stops and the Shortest Path Routes to the Transit Hub

Figure 6.11 Generated Tree of Example in Figure 6.10

Next, the proposed partitioning algorithm generates the partitions (zones) while balancing the workloads. The proposed algorithm can dynamically compute the number of near-balanced partitions for the given tree in contrast to standard graph partitioning algorithms such as k-means, where the number of partitions needs to be specified a-priori. The algorithm is explained in detail below.

Partitioning Algorithm

The problem of finding nearly-balanced partitions of a rooted tree is an NP-hard problem [192]. Therefore, a heuristic partitioning algorithm is proposed. The objective of the partitioning algorithm is to generate partitions (zones) such that (1) road network topology is integrated and (2) the number of passengers in each partition is approximately equal. Integrating the road network topology facilitates to minimize the reduction of the solution quality. Balancing the workloads in each partition facilitates maximum utilization of parallel computing techniques. These objectives are achieved by arranging the bus stops in a tree data structure based on the shortest path route to the destination and then, partitioning the tree such that the number of passengers in each partition is approximately equal. Algorithm 5 presents the pseudo code of the proposed partitioning algorithm. The algorithm takes a tree data structure as the input. The sample tree with 200 passengers used to explain the algorithm is shown in Figure 6.12.

Algorithm 5 Pseudo Code of the Partitioning Algorithm**Input:** tree data structure with passengers in each node**Output:** partitions with approximately equal number of passengers

- 1: **Estimation of partition size** $\Rightarrow$ Sets the preferred partition size range: S_{pref}
- 2: **Identification of nearly-balanced partitions** $\Rightarrow$ Identify sub-trees with passenger counts within the bounds of S_{pref} and save the remaining sub-trees
- 3: **Merging nearby nodes to nearly-balanced partitions** $\Rightarrow$ Merging the remaining sub-trees to existing nearly-balanced partitions or create new partitions
- 4: **Demand-based EV re-allocation**

Figure 6.12 Example Generated Tree with 200 Passengers

Estimation of Partition Size

Determining the number of nearly-balanced partitions is a common problem in traditional workload balancing problems. This is a complex problem even when the number of partitions are specified prior to the execution of the algorithm. It is evident by the works of Feldmann et al. [193], Andersson et al. [194] and Andreev et al. [192]) requiring the number of partitions to be specified prior to execution. However, having a static partition size is not a feasible solution as the optimal number of partitions will depend on the location of passengers. In this problem, the size of a partition is defined by the number of passengers in the partition, which directly correlates to the workload of a partition. Therefore, in an offline setting, the partitioning algorithm can be executed multiple times with different partition sizes to find the best scenario. However, due to the limited amount of resources and runtime constraints, executing

an algorithm multiple times in real-time is not feasible. Hence, a dynamic algorithm which determines the number of partitions at runtime is preferred.

In the proposed method, the preferred size range (number of passengers) of a partition will be decided at the beginning of the algorithm based on the passenger distribution and the EV capacity. The algorithm requires the EV capacity (V_c) to be given as a predefined parameter. The preferred size of a partition will be determined as follows; starting from the root node, the tree will be traversed in a breadth-first search (BFS) until the current node has more than one child node with a passenger count greater than or equal to the EV capacity (V_c). Then, the initial size of a partition (S_{init}) will be set as given in Equation 6.1.

$$S_{init} = floor(\frac{P_{tot}}{N_c}) \quad (6.1)$$

Here, P_{tot} is the total number of passengers and N_c is the number of child nodes with a passenger count greater than or equal to the EV capacity (V_c). Assuming $V_c = 10$, referring to the example in Figure 6.12, the algorithm will traverse to the second level, where it can determine N_c as 3. Then, $S_{init} = 66$. Once the initial size of a partition is set, the selected child nodes are explored to identify nodes with a significantly large number of passengers (greater than S_{init}). If so, the number of child nodes N_c will be incremented as in Equation 6.2.

$$N_c+ = \sum_{C_i \in C} (R_i) mod(S_{init}) \quad (6.2)$$

In Equation 6.2, R_i is the total number of passengers related to the child node, C_i in C . With respect to the example in Figure 6.12, N_c will be incremented by one due to the child node with 100 passengers. Finally, the preferred size range for a partition (S_{pref}) will be determined by Equation 6.3.

$$S_{pref} \in \frac{P_{tot}}{N_c} \pm P_v\% \quad (6.3)$$

Here, P_v gives the size variation percentage (this is set at 10% based on experiments). This procedure will set an upper bound (S_u) and a lower bound (S_l) for the size of a partition. For the example shown in Figure 6.12, $S_u = 55$ and $S_l = 45$. Although, the bounds for partition size is determined in this step, as the algorithm progresses, it will allow the formation of partitions with marginally larger or smaller sizes.

Identification of Nearly-Balanced Partitions

After determining the bounds of a nearly-balanced partition, the next step is to identify them in the tree. A nearly-balanced partition is a sub-tree in the constructed tree where the current node falls within the preferred partition size range. For this, the tree is traversed using a BFS. While traversing, it is checked whether the total number of passengers (N_i) of the current node falls within the preferred partition size. Then, one of the following cases can occur;

Case 1: $S_l \leq N_i \leq S_u$, The sub-tree which has the current node as its root node, will be marked as visited and the passengers in this sub-tree will be identified as a nearly-balanced partition.

Case 2: $N_i < S_l$, The sub-tree which has the current node as its root node, will be marked as visited and the sub-tree will be added to a dictionary (henceforth referred to as *skippedNodes*). The dictionary contains a list of sub-trees per depth of the tree (only the root node of the sub-tree will be saved).

Case 3: $S_u < N_i$, Since the number of passengers exceed the upper bound S_u , the BFS continues to search the child nodes of the current node.

At the end of the search, a set of nearly-balanced partitions of ideal size and a set of sub-trees saved in *skippedNodes* are obtained. The next step in the algorithm strives to merge the sub-trees to existing nearly-balanced partitions or creates new partitions.

Merging Nearby Nodes with Nearly-Balanced Partitions

A strict rule is followed for merging sub-trees, the root node of the nearly-balanced partition selected as an eligible partition and the root node of the sub-tree should be the same (henceforth referred to as *same-parent* constraint). Although this constraint significantly reduces the possibility of merging, by observing the tree, the rule was enforced to prevent undesired combinations, which can result in poor quality routes. Geographically this implies that only two regions which are nearby are merged to form a zone. Also, the number of passengers in an eligible partition should be less than upper bound (S_u) of the nearly-balanced partition size range. The following two scenarios can occur depending on the availability of eligible partitions for merging.

Case 1: Eligible nearly-balanced partitions are available. A similarity score is calculated to identify the best nearly-balanced partition out of the eligible partitions to assign the sub-tree. Here, the similarity score is calculated based on the Euclidean distance of the GPS coordinates of the bus stops in the sub-tree rather than processing the real road network distance to reduce the computational complexity. Given an eligible partition and a sub-tree, the similarity score is calculated as the reciprocal of the Euclidean distance between the average GPS coordinate of all the bus stop locations contained in the eligible partition (henceforth referred to as GPS_{part}) and the average GPS coordinate of all the bus stop locations contained in the sub-tree. Since there can be multiple sub-trees in *skippedNodes* competing for multiple eligible partitions, the similarity score between each sub-tree and eligible partition is calculated and merging is done for the pair with the highest similarity score. This ensures that the most nearby sub-tree and nearly-balanced partition is merged. This process is iterated until merging stops. However, in each iteration, the similarity scores are re-calculated for each sub-tree since adding a sub-tree to an existing nearly-balanced partition, changes GPS_{part} of the partition.

At the end of case 1, there can be sub-trees remaining in *skippedNodes* since eligible partitions are not available (due to the *same-parent* constraint) for some sub-

trees or the eligible partitions have exceeded capacity. Then, new partitions must be created in order to assign the sub-trees in *skippedNodes*. This is done using case 2.

Case 2: Eligible nearly-balanced partitions are not available. A new partition will be created with the sub-tree containing the highest number of passengers out of the sub-trees in *skippedNodes*. Then, the remaining sub-trees in *skippedNodes* is merged with the new partition (if eligible) based on the similarity score until the new partition has exceeded its' capacity or there are no more sub-trees which can be combined with it. This process is repeated until *skippedNodes* list is exhausted.

As a result of the *same-parent* constraint and size of nearly-balanced partitions, sub-trees that represent a set of bus stops with very low demand can be created at the end of merging. Here, a sub-tree that contains passengers less than the capacity of a EV (V_c) is defined as a small partition. Next, the small partitions are merged with the existing partitions (nearly balanced or new partitions) based on similarity scores, subject to the same-parent restriction without considering the partition size. However, it should be noted that even after merging small partitions can still exist as a result of the *same-parent* constraint. The remaining small partitions are not combined with the existing partitions since it can reduce the quality of the routes as it is more likely to introduce detours in the EV routing phase. Also, it should be noted that only small partitions are considered for merging at this stage since the impact to workload balancing is minimal from merging them.

Demand-based EV Re-allocation

As mentioned in Section 3.3, EVs are assumed to be located in nearby car parks closer to the bus stops. Hence, at the end of partitioning, the algorithm can compute the number of EVs in each partition. However, it is possible to form partitions where the supply of EVs is less than the demand of passengers. Executing the directionality-centric clustering strategy for such partitions is not useful because EVs are not available in the partition to allocate all the passengers. Therefore, after finalizing the partitions,

excess EVs are removed from partitions and allocated to other partitions which require additional EVs. During the selection of EVs for re-allocation, it is ensured that EVs most beneficial to the native partition are not removed to prevent degradation in quality of routes. Therefore, EVs which are farthest away from the destination in each partition are kept intact (EVs closest to the destination are chosen for re-allocation). The motivation of selecting the EVs farthest away is to enable such EVs to pick up a higher number of passengers from other bus stops along the way to the destination, which contributes to reduce the VMT. Here, similar to case 1 of merging nearby nodes with nearly-balanced partitions, a similarity score is calculated for all the excess EVs and the partitions which require additional EVs. EVs are then assigned based on the similarity score. The partitioning algorithm is terminated after re-allocation of EVs. The algorithm segments the transit hub region into workload balanced zones, which contain bus stops in close proximity and the allocated EVs.

6.2.4 Generating Sub-Zones

The techniques proposed in Section 6.2.3 segments a transit hub region into zones. Next, the techniques proposed in Chapter 5 are used to generate sub-zones. This enables to further subdivide the problem of generating routes of the large geographical region into small manageable problems. The algorithm proposed in Chapter 4 is implemented in each sub-zone to generate the routes. The output of this technique generates the flexible routes for each zone. All the routes from the zones form the set of routes for the transit hub region. Thus, all the proposed techniques can be combined for adopting demand-aware bus transit system for large scale deployment. The methodology is explained in Section 6.2.5.

6.2.5 Citywide Adoption

Figure 6.13 Methodology for Citywide Adoption

All the proposed techniques can be used for citywide adoption of demand-aware bus transit system. Section 6.2.2 presented a technique for identifying independent transit hub regions from a given large geographical region. Section 6.2.3 showed a technique for segmenting a transit hub region into zones. Section 6.2.4 presented a method of using the techniques in Chapter 5 for generating sub-zones. Figure 6.13 presents the methodology for citywide adoption. The input to the methodology consists of the large geographical region such as city and the bus transit network. Next, the transit hubs in the large geographical region are identified. Thereafter, the transit hub regions are identified by leveraging the feeder bus services as explained in Section 6.2.2. The independent transit hub regions are further segmented into zones

by the proposed partitioning algorithm in Section 6.2.3. Thereafter, the bus transit network segmentation technique proposed in Chapter 5 is implemented in the zones to generate sub-zones, where the routes are computed for the fleet of EVs. At the end of execution of the bus transit network segmentation technique, the set of flexible routes for each zone is obtained. Finally, the routes of all the zones in a transit hub region are combined to generate the flexible transit hub region routes. The final output is the flexible routes for each transit hub region.

6.3 Performance Evaluation

This section presents the experimental results of the proposed partitioning algorithm that is used to segment a transit hub region into balanced zones. Since a transit hub region functions independently, the experiments conducted in this section are based on a single transit hub region. Boon Lay transit hub region, outlined in blue in Figure 6.9 is used in the experiments. All the assumptions in Chapter 3 regarding the location of EVs, random distribution of passenger requests, driving range and capacity constraints of EVs, time and distance matrix generation using OSRM are used in the experiments. The main purpose of the experiments is to show that the proposed partitioning algorithm can generate balanced partitions and that the quality of solutions and runtime is significantly better compared to existing partitioning algorithms. Therefore, experiments are conducted in different scenarios by varying the demand. The proposed partitioning technique is implemented in JAVA. Runtime is measured on a PC with 32 GB RAM, running Windows 10 Pro on an Intel Xeon E5-1630 V4 CPU at 3.70 GHz.

6.3.1 Experimental Setup

Experimental Parameters

Table 6.2 shows the values of the parameters used in the experiments. EV capacity, driving range and tolerance factor are set to 16, 30 km and 5 respectively. In the GA, population and number of iterations are set to 50 and 100 respectively.

Table 6.2 Values of the Parameters

Parameter	Value
EV Capacity	16
EV Driving Range	30 km
Tolerance Factor	5
Population Size	50
Iterations	100

Test Cases

Experiments are conducted by generating multiple test cases by varying the demand and supply. Table 6.3 shows the test cases used in the experiments. Here, the demand shows the total number of ride requests in the transit hub region considered in the partitioning algorithm and the number of empty EV are given as the supply. Compared to previous chapters, these test cases use significantly large demands that represent the transit hub region.

Table 6.3 Test Cases

Test Case	Demand (Requests)	Supply (EV)
1	600	120
2	700	120
3	800	120
4	900	120

6.3.2 Experiments

In order to compare the performance of the proposed methodology, the partitioning technique is replaced using the widely-used k-means algorithm [190]. However, k-means algorithm requires the number of partitions to be specified prior to the execution of the algorithm. Hence, k-means algorithm is used with $k = 5$ (henceforth referred to as KM_5) and $k = 10$ (henceforth referred to as KM_{10}). The workload balancing capability of the algorithm is compared by analysing passenger distribution among the partitions. Next, the solution quality is analysed by comparing the VMT of the fleet of EVs for the given test cases. Then, the runtime of the three algorithms are compared.

Workload Balancing

Workload balancing capability of the proposed partitioning technique is compared by the distribution of passengers in each zone. For all scenarios given in Table 6.2, the proposed partitioning algorithm identifies 5 zones. KM_5 and KM_{10} algorithms have 5 and 10 zones respectively. Passenger distribution among zones for the three algorithms is shown in the clustered bar chart given in Figure 6.14. Here, the horizontal axis shows the number of passengers and the vertical axis shows the zone number. Distribution of passengers for the proposed, KM_5 and KM_{10} algorithms are shown in orange, black and grey bars respectively. The results in Figure 6.14 indicate that passengers are evenly distributed in the proposed algorithm compared to KM_5 , which has the same number of zones as the proposed algorithm. KM_5 shows large variations in the number of passengers distributed between the zones. An analysis of the spread of the three passenger distributions shown in Table 6.4 confirms this. Table 6.4 shows the average values of the maximum & minimum number of passengers in the zones, the range and the variance of the distributions. Distribution of passengers for the case of KM_{10} is relatively better compared to the proposed algorithm. However, workload balancing alone does not improve the runtime of the algorithm. For example, the imbalance in zone sizes of the proposed method can be justified since the proposed

(a) Test Case 1

(b) Test Case 2

(c) Test Case 3

(d) Test Case 4

Figure 6.14 Passenger Distribution in Zones

algorithm considers the similarity of routes to the destination during the partitioning process, which infers additional constraints. Violating these constraints may generate balanced zones but leads to low quality routes. This is shown in the results of the solution quality and runtime.

Table 6.4 Analysis of the Passenger Spread in the Zones

Method	Parameter			
	Max.	Min.	Range	Variance
Proposed	210	46.25	163.75	64.04
KM_5	278.75	44.75	234	77.91
KM_{10}	183.25	23	160.25	41.65

Solution Quality

Table 6.5 Comparison of VMT

Test Case	VMT (km)		
	Proposed	KM_5	KM_{10}
1	193.8	207.9	189.3
2	143.2	159.6	144.8
3	148.9	158.4	153.2
4	170.6	172.3	168.9

Solution quality is measured by the VMT of the fleet of EVs. This is obtained by combining all the routes generated in all the zones. Routes for the zones are generated using the directionality-centric bus transit network segmentation technique as given in Section 6.2.4. The single-objective GA is used in all algorithms to generate the flexible routes for each zone with a population size of 50 and 100 iterations. A lower VMT for the fleet suggests that the solution generated is superior which in turn results in higher profits for the operator. The VMT of the fleet for the 4 test cases is presented in Table 6.5. Results indicate that the proposed method outperforms KM_5 by a significant margin in all test cases. This is due to the superiority of the generated zones in the proposed method, which leverages road network data embedded within the tree structure. In the case of KM_{10} , the VMT of the fleet is marginally similar in two test cases. However, the runtime of the proposed technique is significantly better

which underlines the benefits of the proposed technique. Further, specifying the value of “k” in k-means algorithm does not generate consistently superior solutions as shown for the case of $k = 5$ and $k = 10$.

Runtime

Runtime results are presented in Table 6.6. As explained in Chapter 4, the GA which is used to generate the routes in each zone will terminate the route generation prior to reaching 100 iterations if the solution quality is not improved after 5 consecutive iterations. This is exemplified in the runtime results. Due to the quality of the zones created from the proposed partitioning algorithm, the routes generated can converge much quicker. Therefore, it does not require all 100 iterations. Similar to the results for solution quality, runtime of the proposed technique is significantly better compared to the KM_5 algorithm. This is due to the poor workload balancing of the KM_5 algorithm, which results in several sub-zones consuming a significant time to generate the routes. The proposed method outperforms the other approaches by a significant margin except for test case 1, where KM_{10} produces better routes with almost similar runtime. In contrast, the proposed method consistently generates higher quality solutions with significantly lower runtimes.

Table 6.6 Comparison of Runtime

Test Case	Runtime (s)		
	Proposed	KM_5	KM_{10}
1	25	133	26
2	28	162	69
3	34	211	77
4	46	315	128

6.4 Demand-aware Scheduling of Point-to-Point Connections

The methodology proposed in Section 6.2.5 can be used for citywide adoption of demand-aware bus transit system, which can be used to improve the user experience and operator profitability of the existing feeder bus services. However, passengers serviced by trunk bus services, which provide long distance point-to-point connectivity can also benefit from a demand-aware bus transit system. Therefore, in this subsection, a technique that can be used for enhancing user experience and operator profitability for long distance point-to-point connectivity is proposed. The proposed technique can be integrated with the method presented in Section 6.2.5 to provide a demand-aware bus transit system for both feeder and trunk services.

Trunk bus services provide point-to-point connections between bus interchanges (transit hubs) or terminal stations. Even though, most of the trunk bus service routes are changed to align with the hub & spoke model to avoid the duplication of routes with the high speed rapid transit links [15], several trunk bus services still provide connections between towns. The main purpose of long distance trunk bus services is to reduce the inconvenience of passengers by the bus transfers and waiting times. Therefore, trunk bus services often depict common properties. This is explained using Figure 6.15a and Figure 6.15b, which shows the routes of a few trunk bus services originating at Boon Lay and Jurong East transit hubs respectively. Figure 6.15 confirms that unlike the feeder bus services shown in Figure 6.6a and Figure 6.6b, which service a small geographical area of coverage, trunk bus services cover a large geographical area. Therefore, the end to end travel time of these bus services are relatively high. Also, overlapping of routes for trunk bus services is relatively low compared to the feeder bus services.

At the same time, a trunk bus service operates as two independent routes (directions) and may even have different schedules for the route between the two bus interchanges/terminal stations. Figure 6.16 shows the different scheduling frequencies

(a) Trunk Buses Originating at Boon Lay Transit Hub

(b) Trunk Buses Originating at Jurong East Transit Hub

Figure 6.15 Trunk Bus Service Routes

of the bi-directional trunk bus service number 30 in Singapore. However, compared to a feeder bus service frequency given in Figure 3.6, the scheduling frequency of a trunk bus service is relatively low. Thus, waiting time of passengers for this service is high. Further, similar to the case of the feeder bus services, the published operating schedules are seldom adjusted based on the demand. However, if the demand can be obtained or predicted, the techniques proposed in Section 4.4 can be used for demand-aware scheduling of trunk bus services.

30	SBS Transit	Weekdays		Saturdays		Suns / P.Hs	
		1st bus	Last bus	1st bus	Last bus	1st bus	Last bus
	Bedok Int	0515	2335	0515	2335	0530	2335
	Boon Lay Int	0530	0020	0530	0020	0545	0020

Period	0630 - 0830	0831 - 1659	1700 - 1900	After 1900
Freq. Range	09 - 09 minutes	06 - 18 minutes	14 - 15 minutes	11 - 16 minutes
Direction 1	08 - 11 minutes	08 - 16 minutes	08 - 15 minutes	11 - 18 minutes
Direction 2				

Figure 6.16 Different Scheduling Frequencies of a Trunk Bus Service

Compared to feeder bus services that have a short round trip, trunk bus services have a long trip time. In order to schedule buses based on demand, an operator needs to predict the future demand for a long duration in order to make scheduling decisions. Therefore, as an alternative, it is proposed to use the historical demand data. To this end, Section 7.3 proposes a technique for aggregating historical demand data. Therefore, by leveraging the historical demand data, trunk bus services can be scheduled based on demand. However, since the trunk bus service will have to visit most of the bus stops along the route, incorporating flexible routing is not considered as an option in this case. Further, since each trunk bus service operates independently, scheduling decisions can be performed in parallel without impacting the scalability. Demand-aware scheduling can reduce the waiting time of passengers and thus improve the user experience. Also, it can be used to reduce the number of buses in operation, since the buses do not have to use a fixed schedule when the demand is low, which enhances the operator profitability. However, it is noteworthy that demand-aware schedules will have to be updated periodically in order to facilitate the evolving travel patterns.

The proposed technique for demand-aware scheduling of trunk bus services can be integrated into the methodology given in Section 6.2.5 for citywide demand-aware bus transit system. Figure 6.17 shows the method for leveraging the road network and existing bus transit routes for citywide adoption of the demand-aware bus transit system and the proposed demand-aware scheduling technique for point-to-point connections.

Figure 6.17 Proposed Methodology for Integrating Demand-aware Scheduling of Point-to-Point Connections

The input to the methodology consists of the large geographical region and the bus transit network. Next, the transit hubs in the large geographical region are identified. Thereafter, the bus services of each transit hub are classified into feeder and trunk services. The feeder bus system is used to implement the techniques proposed in Sections 6.2.2, 6.2.3 and 6.2.4. At the same time, demand-aware scheduling of the trunk bus services is performed using the techniques proposed in Section 6.4. Therefore, the proposed demand-aware scheduling technique is scalable and can be effortlessly integrated with the methodology for citywide adoption of demand-aware bus transit system.

6.5 Summary

This chapter proposed a methodology for adopting demand-aware bus transit system for large scale deployment. The proposed methodology systematically identifies independent transit hub regions from a large geographical region such as a city by leveraging the hub & spoke topology of the public transit network. The independent transit hub regions, which are generated offline enable to significantly reduce the complexity of computations by facilitating parallelism. In order to increase the responsiveness, the transit hub regions are further segmented into workload balanced zones by using the proposed partitioning algorithm. The partitioning algorithm generates nearly-balanced zones while taking into account the real-time demand. Also, the algorithm incorporates the road network topology that improves convergence. The strategy proposed in Chapter 5 is implemented in each zone to generate flexible routes.

The partitioning algorithm has been evaluated using a real road network of a transit hub region. Experiments conducted using openly available public transit demand data show that the partitioning algorithm can generate nearly-balanced zones compared with a widely-used unsupervised learning algorithm. Experimental results confirm the notable 83% improvement in runtime obtained without compromising the solution quality when the technique proposed in Chapter 5 is used to generate the routes. This improvement is achieved as a result of balancing the workload in each zone and the integration of the road network topology in the partitioning algorithm. Balancing the workload reduces the complexity of the routing algorithm and integrating road network topology helps to converge much faster due to superior solutions. This confirms that the partitioning algorithm can be reliably used to generate workload balanced zones for a transit hub region.

Noting that trunk bus services that provide long distance point-to-point connectivity can also benefit from a demand-aware transit system, a technique that can improve user experience and operator profitability is proposed. The technique leverages historical demand data to generate demand-aware schedules for the trunk bus services. Thus, the

proposed methodology can be used for adopting a citywide demand-aware bus transit system.

Chapter 7

Citywide Deployment of Demand-aware Bus Transit System

7.1 Introduction

Chapter 6 proposed a methodology for adopting demand-aware bus transit system for large scale deployment. The methodology systematically identified independent transit hub regions from a large geographical region such as a city, which are further segmented into workload balanced zones using the proposed partitioning algorithm. This enabled to leverage the bus transit network segmentation methodology in Chapter 5 to generate flexible routes for the fleet of EV in each sub-zone. At the same time, a demand-aware scheduling technique was proposed to improve the point-to-point connections. However, deploying the demand-aware bus transit system requires techniques for accurately estimating real-time and near-future demand that helps to make intelligent scheduling and routing decisions that improve user experience and operator profitability. In addition, techniques need to be devised to integrate demand estimation techniques with the routing techniques proposed in this thesis. To this end, this chapter proposes a framework for citywide deployment of demand-aware bus transit system that unifies the techniques presented in this thesis.

The proposed framework enables to leverage demand estimation and route computation techniques for citywide deployment of demand-aware bus transit system. In addition, the framework facilitates demand-aware scheduling of long distance point-to-point connections. The intelligent techniques deployed in the framework not only make it scalable but also highly responsive. The rest of the chapter is organized as follows. Section 7.2 discuss techniques for real-time demand generation and estimation. Section 7.3 proposes a technique for near-future demand estimation. Section 7.4 presents the proposed framework that unifies all the techniques for citywide deployment of demand-aware bus transit system. Section 7.5 presents the conclusions.¹

7.2 Real-Time Demand Generation & Estimation

The techniques proposed in the previous chapters can be used to rapidly find near-optimal routes for a fleet of EVs providing demand-aware bus transit in a large geographical region such as a city. However, unlike existing bus transit system that operate with fixed schedules and routes, the demand-aware bus transit system operates with flexible schedules and routes by relying on the demand. Therefore, obtaining the real-time demand is crucial for the functionality of the system.

Dedicated mobile phone-based applications are the most widely-used method for active real-time demand generation. For example, existing ride-hailing services such as Uber, Grab and Lyft use a dedicated mobile phone application, which enables the user to indicate the ride-request. In general, the dedicated application allows the user to indicate the origin, destination, number of required seats and in some cases the type of vehicle that is required. Then, the ride-hailing system will aggregate the demand of all passengers that provided the information to match the ride requests with the available vehicles. Dedicated mobile phone applications have the added advantage of generating the exact real-time demand unlike passive techniques, which only provide approximations. However, passive techniques are required for real-time demand

¹The work in this chapter has been published in [195]

generation in instances where passengers do not actively provide the ride requests using the dedicated mobile phone-based applications. Further, for bus transit systems, passengers may not expect to actively provide the demand. Therefore, intelligent passive sensing techniques are needed to generate or estimate the real-time demand.

To this end, this section explains passive techniques for real-time demand generation and estimation. Section 7.2.1 shows a technique for leveraging mobile phone-based sensors. Section 7.2.2 presents camera-based crowd estimation techniques. Section 7.2.3 shows a technique for real-time demand estimation using public transit tap-in and tap-out data.

7.2.1 Mobile Phone-based Sensors

The penetration of mobile phones has proliferated around the globe [196]. As highlighted in the study in [197], 67% of world population have been active unique mobile phone users in 2019. Thus, the mobile phone can be used as a reliable universal tool for sensing of crowds. At the same time, features of mobile phones have also been enhanced with novel low-energy and efficient communication technologies such as Bluetooth and WiFi. This enables mobile phones to be leveraged as sensors for passive demand generation. This subsection explains a technique for real-time demand generation using mobile phones.

Passive sensing techniques can use mobile phone applications in conjunction with Bluetooth technology for real-time demand generation. Bluetooth technology relies on the short range communication between a mobile device and bluetooth low energy (BLE) beacon. A BLE-beacon broadcasts packets of data at regular intervals of time. These data packets are detected by an application or installed services of nearby mobile phones. The application would then communicate with a server where a database is maintained with anonymised asset IDs of mobile phones and temporal data. A typical scenario of using BLE-beacons for device tracking is shown in Figure 7.1. This shows a bus stop, which has been installed with 3 BLE-beacons. Assuming that passengers

Figure 7.1 Example of BLE-beacon based Real-Time Demand Generation at a Bus Stop. Passengers that have Installed the Mobile Application and Enabled Bluetooth Communication will Connect to the 3 Placed BLE-Beacons to Share Anonymised Data. This Enables to Generate Real-Time Demand at the Bus Stop

in the bus stop have installed the application and Bluetooth service is enabled, such a setup will allow to track the anonymised asset IDs of mobile phones. This data can be used to generate the real-time demand at the bus stop. However, as opposed to the active technique, the operator needs to install BLE-beacons in all bus stops for demand generation. Further, Bluetooth communication needs to be enabled in the mobile device, which can create issues such as privacy and battery draining. However, it can be used as a reliable source for real-time demand generation.

It should also be noted that the mobile phone-based passive technique that generates real-time passenger demand, provides an approximation of the number of passengers at a bus stop. For example, since the technique counts the number of devices, it can count multiple phones that are being used by a single user resulting in a marginally higher approximation. Also, if a user does not possess a mobile phone, the approximation will be lower than the actual value. However, since the anonymised asset IDs of mobile phones are maintained in the database, this technique can be used to distinctively identify a passenger in a bus stop. This can facilitate the prediction model proposed in Section 7.3 to provide robust estimations of near-future demand.

7.2.2 Camera-based Crowd Estimation

Section 7.2.1 proposed a technique that leverages mobile phones for generating the real-time demand by using dedicated applications in conjunction with Bluetooth technology to count devices near BLE-beacons. This section presents a robust camera-based crowd estimation technique that leverages a live video footage instead of the presence of a mobile phone. At the same time, compared to mobile phone-based sensors, the camera-based crowd estimation technique does not create issues such as battery draining and requirement to install applications.

Surveillance cameras can be leveraged as low-cost devices to generate the real-time demand at bus stops. For example, the output of a surveillance camera directed at a bus stop can provide a live video footage of the bus stop. This can be used alongside a computer-vision software that can automatically count the number of passengers in the bus stop. Similar studies have been conducted in literature such as the work by the sensing and management for agile transport (SMAT) group at TUM-CREATE [198] and Wang et al. [199]. The SMAT group has devised fast and accurate computer-vision based techniques for automatic real-time scene understanding. This encompasses techniques for detecting passengers for estimating crowds using deep learning methods [200] [201].

A sample setup of a surveillance camera is shown in Figure 7.2. Here, the camera shown in the top left corner is mounted opposite the road and directed at a bus stop. The live video footage can be used to count the number of passengers in the bus stop in real-time. Two sample snapshots of a video footage that has been processed by the computer vision software are shown in Figure 7.3. In both snapshots, passengers that have been identified at the bus stop are marked in green dots. The SMAT group highlights that computation time of the proposed robust crowd-estimation technique is in the order of seconds. Therefore, the technique can be used to sample the live footage at a high frequency (every minute), which can provide the required real-time estimates.

Figure 7.2 Example of using Surveillance Camera for Real-Time Demand Generation. The Surveillance Camera is Mounted at a Pole Opposite the Bus Stop. It can be used Alongside a Computer Vision Software to Count Real-Time Passenger Demand

Figure 7.3 Output of Computer Vision Software for Counting Passengers. Identified Passengers in the Bus Stop has been Marked using The Green Dots

The proposed setup can be used to generate the overall real-time demand in a bus stop. Although, generating the overall real-time demand in a bus stop is sufficient for the functionality in most areas, identifying the distinctive demand for each transit hub region is required for the functionality in overlapped transit hub regions. As mentioned in Section 6.2.2, different bus services originating at multiple transit hub regions, service the overlapped transit hub region. Therefore, generating the total

demand at a bus stop, which belongs to an overlapped transit hub region is insufficient for identifying the unique demand. To this end, the generated real-time demand can be used in conjunction with the real-time bus arrival data. The difference between the number of passengers prior to the arrival of a bus and after the arrival can provide an approximation of the number of passengers using a bus service. Since feeder bus services originate at different transit hub regions, this will enable to identify the distinctive demand for a transit hub region. Therefore, real-time bus arrival information can be used as a parameter for estimating the distinctive demand in the technique proposed in Section 7.3.

7.2.3 Public Transit Tap-In & Tap-Out Data

The techniques proposed in Sections 7.2.1 and 7.2.2 can be used for real-time demand generation by detecting the presence of a mobile phone or using surveillance camera footage alongside a computer-vision software for counting passengers. The proposed techniques generate approximations of real-time demand at bus stops. However, in some instances, the exact demand for bus services can be obtained using openly available public transit fare card tap-in and tap-out data [202] [203] [204]. This section presents a technique to obtain real-time demand estimates using the fare card tap-in and tap-out data.

In the proposed demand-aware bus transit system, the number of rides originating from bus stops need to be obtained to estimate the demand at any given time. To this end, an operator can rely on existing openly available public transit data published by transit operators and regulators [202] [203] [204]. For example, the number of passengers boarding at bus stops can provide the operator the typical historical demand for the service. In the case of Singapore, the Data Mall by the LTA [202] is used to obtain the demand data of bus services. Anonymised fare card data from the "Passenger Volume by Bus Stops" and "Passenger Volume by Origin Destination Bus Stops" APIs of the Data Mall can be used for this. The first data set provides the tap-in and tap-out passenger volume for individual bus stops. However, this data consists of passengers

using multiple bus services. In such a scenario, the operator can rely on the second data set, which provides the number of trips from the origin to destination bus stop. Although, fare card tap-in and tap-out data provide the exact demand for a bus service, one drawback of this data is that it can only be obtained with a significant delay. Therefore, generating the real-time demand using fare card tap-in and tap-out data can only be performed using a prediction algorithm. Similar works have been explored in the past by Zhang et al [205], which implements a system utilizing tap-in and tap-out data and GPS data to forecast the passenger flow in real-time. An alternative approach of using fare card tap-in and tap-out data is to update a historical demand database, which can be used for validating the real-time demand that was generated using the methods proposed in Sections 7.2.1 and 7.2.2. Another drawback of the historical fare card tap-in and tap-out data is the resolution of available data. For example, the LTA, Singapore publishes data with a resolution of 1 hour. Therefore, determining the exact passenger count of a single bus requires additional processing such as obtaining the number of buses that are in operation and building sample profiles of passenger flow.

7.3 Near-Future Demand Estimation

Section 7.2 presented techniques for real-time demand generation and estimation at a bus stop. Although, the real-time demand can be used to determine the schedules and routes of the proposed demand-aware bus transit system, obtaining the near-future demand at a bus stop is significantly beneficial for both passengers and the operator. Passengers can get a better user experience and at the same time, the operator can increase the utilization of the fleet which results in higher profits. Therefore, this section proposes a technique for near-future demand estimation by leveraging the techniques for real-time demand generation and estimation.

Figure 7.4 shows the proposed technique for estimating the near-future demand at a bus stop. The proposed technique is based on a prediction model that uses a standard Long Short Term Memory (LSTM) network [206]. Prediction models have been used

Figure 7.4 Proposed Technique for Near-Future Demand Estimation. Technique uses Demand Data from Mobile Phone-based Sensors, Camera-based Crowd Estimation and Fare Card Tap-In & Tap-Out Data to Estimate the Near-Future Demand

in literature for estimating the passenger flow in different domains. Tang et al. [207] strive to facilitate proactive dynamic traffic control by monitoring the present traffic and foreseeing its immediate future. Yong et al. [208] proposes a hybrid, optimized LSTM network to model bus passenger flow data to facilitate dynamic bus scheduling and regional coordination scheduling. Therefore, it is assumed that LSTM networks can be used as a robust and effective technique for estimating demand. The model consists of an offline training phase and an online estimation phase. The input to the prediction model consists of the real-time demand estimates of the bus stop and the temporal information such as day, type of day, time and bus arrival. Data from the historical demand database are used to train the LSTM network. During execution, once the input data are provided to the prediction model, the estimated near-future demand for a bus stop for the next n minutes (timeframe for near-future demand estimation) is output from the model. The proposed technique is described below.

The proposed technique relies on the real-time demand estimates obtained from the methods presented in Section 7.2. The real-time demand estimate from all 3 sources are stored in the historical demand database, which is used for training the LSTM network. However, during the online estimation phase, only camera-based crowd

estimation data and mobile phone-based sensor data are considered for near-future demand estimation. As shown in Figure 7.4, the real-time estimation from a selected source is provided as input to the prediction model. The criterion for source selection is explained in Section 7.4.1. On the other hand, fare card tap-in and tap-out data can only be obtained with a significant delay. Therefore, it is not used as a source for near-future demand estimation in the prediction model. However, this data is significantly beneficial since it can be used as a validation source during the training phase.

Training the prediction model is performed offline using the historical demand database. During the training phase, the historical demand data is provided to the demand estimation system with the corresponding real-time inputs. Also, the prediction model is provided with the temporal information such as the day, type of day, time, bus arrival. These features will be significantly important to predict the near-future demand at a bus stop. For example, the demand observed in a bus stop at a given time of two different days reflecting a weekday and weekend will be vastly different. Also, in special cases such as a holiday, the demand during a weekday will also be less. Bus arrival information is required for bus stops located in overlapped transit hub regions.

After the training phase, the model can be used to predict the near-future demand for a bus stop for the next n minutes by providing the real-time demand and temporal information. However, it is assumed that the timeframe for near-future demand estimation (n) will significantly impact the prediction accuracy of the model. For example, predicting the near-future demand for the next 15 minutes is expected to have a higher accuracy compared to predicting the demand for the next 30 minutes. Therefore, at runtime, the proposed framework should intelligently leverage the timeframe for near-future demand estimation.

7.4 Proposed Framework for Citywide Deployment

Techniques have been presented in this thesis for estimating near-future demand, decomposing citywide large geographical regions into sub-zones for demand-aware flexible routing, and demand-aware scheduling of point-to-point connections. However, deploying the proposed techniques require selecting between alternative input methods to meet robustness and intelligent division of workloads to manage runtime performance. In this section, techniques for deploying the proposed demand-aware bus transit system are discussed followed by the proposed framework for citywide deployment.

7.4.1 Demand Estimation Process

Section 7.3 proposed a technique for near-future demand estimation by leveraging real-time demand generation and estimation techniques. The proposed technique estimates the demand for the near-future demand estimation timeframe for each sampling instance of input data. Figure 7.5 shows the technique of realizing the proposed demand estimation process at runtime. Compared to the technique proposed in Section 7.3, two additional functions need to be performed while deploying the demand estimation process. This subsection explains these two functions, namely *source selection* and the feedback path from the *scheduling decision block*.

As shown in Figure 7.4, a real-time estimation source needs to be selected for runtime operation of the proposed model. Therefore, it is assumed that the proposed technique relies on the camera-based crowd estimation technique as the primary source. This is shown in the *source selection* module in Figure 7.5 that gives priority to camera-based estimation output. Here, priority is given to the camera-based crowd estimation technique since it can accurately estimate the number of passengers in a bus stop compared to the mobile phone-based technique. Although, mobile phone-based sensors can provide the distinctive demand of a bus stop by tracking the anonymized asset IDs, the estimates are significantly dependent on the usage of mobile phones by

Figure 7.5 Technique for Realizing the Near-Future Demand Estimation Process

the passengers. However, estimates from mobile phone-based sensors can also be used as an alternative input especially in developed cities where mobile phone penetration is significantly high.

The near-future demand estimates of all the bus stops encompassed by a transit hub region are input to a decision block that is executed to evaluate if EVs should be scheduled. The *scheduling decision block* performs this function. The outcome of the *scheduling decision block* is communicated through the feedback path to update the prediction module. The feedback is used in the next sampling instance of the prediction model to appropriately adjust the near-future demand estimates.

7.4.2 Demand-aware Online Scheduling & Routing Process

Section 6.2.5 presented a methodology for citywide adoption of the proposed demand-aware bus transit system by leveraging the hub & spoke topology of public transit

Figure 7.6 Technique for Realizing the Demand-aware Online Scheduling & Routing Process. The Process is Divided into Offline and Online Modules that Expedites Real-Time Flexible Route Generation

network and the bus transit system. The technique enabled to generate demand-aware routes for each independent transit hub region by leveraging the techniques proposed in previous chapters. This subsection shows the technique of deploying the proposed methodology given in Figure 7.6. The different colours in Figure 7.6 highlight the geographical areas, namely grey for city, green for transit hub region, yellow for zone and blue for sub-zones.

The demand-aware online scheduling & routing process generates the citywide flexible routes for the demand-aware bus transit system. Therefore, the main factor that needs to be considered when realizing the proposed process is to minimize the computation time that improves the responsiveness of the demand-aware bus transit system. As mentioned in Chapter 6, the technique of decomposing the city into independent transit hub regions, zones and sub-zones enables to parallelize the computations to manage scalability. However, the computation time can be significantly reduced by processing some of the tasks offline. Thus, as shown in Figure 7.6 the process consists of an offline module that is executed once and an online module that is executed in real-time. The two modules are shown in yellow and purple shading in Figure 7.6 respectively.

The input to the offline module is the citywide geographical region and the bus transit system. The locations of the transit hubs are identified by the former input, while the latter is used to classify the bus services as explained in Section 6.2.1. Next, as explained in Section 6.2.2, the transit hub regions are identified by using the feeder bus services. The method in Section 6.2.3 segments the transit hub regions into balanced zones. Although, generating balanced zones require the demand of the system, initial zones are created by utilizing the bus stops as shown in *generation of zones* in Figure 7.6. This significantly reduces the runtime of the online module.

The online module generates the flexible routes for the transit hub regions by intelligently leveraging the demand and supply. The input to the module consists of the near-future demand, supply of vehicles and initial zones. Initially, the near-future demand obtained from the demand estimation process is used in the *formation of*

balanced zones step. Techniques proposed in Section 6.2.3 are used for balancing the zones. The demand of the balanced zones is then used as the input in the *vehicle allocation of zones* step. The demand of the sub-zones is identified by using the bus transit network segmentation technique proposed in Section 5.2. This is performed in the *generation of sub-zones* step. Next, the demand of each sub-zone and supply of zones are used in the *vehicle allocation of sub-zones* step. This allocates the minimum number of EVs required for each sub-zone by matching the input demand and EV capacity. This technique enables to minimize the number of EVs such that complexity is reduced, which aids to lower the runtime. After identifying the demand and supply of each sub-zone, the next step performs the *demand-aware routing*. This generates the routes in each sub-zone. Thereafter, the obtained routes of each sub-zone are passed to the *route generation of zones* step. The sub-zone routes will be combined to generate the flexible routes of each zone using the *route generation of zones* step. Finally, all the routes of the zones are processed in the *generation of transit hub region routes* step. All the routes of the transit hub regions are combined to generate the citywide demand-aware flexible routes.

7.4.3 Demand-aware Offline Scheduling Process

Section 6.4 proposed a methodology for demand-aware scheduling of point-to-point connections provided by trunk bus services. The technique enabled the offline generation of demand-aware schedules for trunk bus services. This subsection shows the technique of realizing the proposed methodology given in Figure 7.7.

Similar to Section 7.4.2, the input to the process consists of the citywide geographical region and the bus transit system. The locations of the transit hubs are identified by the former input, while the latter is used to classify the bus services as explained in Section 6.2.1. Initially, the trunk bus services are grouped by using the locations of the origin transit hub. Although, the computations of this process are performed offline, grouping the bus services will enable to systematically partition the trunk services. Therefore, the remaining functionality can be performed in parallel based on

Figure 7.7 Technique for Realizing the Demand-aware Offline Scheduling Process. The Process Generates Demand-aware Offline Schedules for Trunk Bus Services

the transit hub. The travel time of trunk bus services is significantly high. Therefore, reacting to the near-future demand at runtime is not feasible. As a result, the proposed technique uses historical travel demands to generate demand-aware schedules for trunk bus services. The demand patterns of each bus service are identified by utilizing the *historical demand database*. The database is updated as shown in Section 7.4.1. It should be noted that after updating the schedules of the trunk bus services, the travel patterns of passengers can change according to the new schedules. Therefore, it may be required to periodically update the schedules. The brown dotted line shows the task of periodic execution of schedule generation. However, determining the schedule update frequency requires further exploration, which is beyond the scope of this thesis.

7.4.4 Proposed Framework

This section presents the proposed framework for citywide deployment of demand-aware bus transit system as shown in Figure 7.8. The framework unifies demand

Figure 7.8 Proposed Framework for Citywide Deployment

estimation (Section 7.4.1), demand-aware online scheduling & routing (Section 7.4.2) and demand-aware offline scheduling (Section 7.4.3) processes. The demand estimation process estimates the demand at each bus stop for the timeframe for near-future demand estimation. The demand-aware online scheduling & routing process computes the citywide flexible routes in real-time by leveraging the decomposed regions, near-future demand and supply of vehicles while considering the system requirements. The demand-aware offline scheduling process computes the schedules for the trunk bus services. This section explains the details of the framework and highlights the intelligent techniques used for citywide deployment.

The demand estimation process (shown in red outline in Figure 7.8) outputs the near-future demand of a bus stop for each sampling period. This is based on the technique explained in Section 7.4.1. The *historical demand database* is maintained by storing all three real-time demand estimation sources. The database is used for two periodic offline tasks, namely as training data for the prediction model and for the scheduling of trunk bus services. At runtime, the prediction model estimates the near-future demand for the next n (timeframe for near-future demand estimation) minutes for each sampling period. The estimates of all the bus stops encompassed by the transit hub region are fed into a decision module that is executed to evaluate if EVs should be scheduled. If the near-future demand exceeds the threshold, it triggers the online module of the demand-aware scheduling & routing process. The feedback path updates the prediction module of the outcome of the scheduling decision. The input from the feedback path is used in the next instance of the prediction model to appropriately adjust the near-future demand estimation.

The demand-aware online scheduling & routing process (shown in mauve outline in Figure 7.8) consists of two modules. The two modules are shown in yellow and purple shading in Figure 7.8 respectively. The process is realized by using the technique described in Section 7.4.2. The demand-aware offline scheduling process (shown in green outline in Figure 7.8) performs the trunk bus scheduling. This process is realized as explained in Section 7.4.3. However, in the proposed framework, several intelligent

interconnections optimize and expedite the computations. These are described in detail below.

The supply of vehicles is provided to a *vehicle database* instead of directly providing to the *vehicle allocation of zones* step. The database consists of the typical demand observed in the transit hub region, which will determine the type of EV that is most suitable for deployment. As shown in Section 4.4, the most profitable type of EV (mini vans/mini buses) will be dependent on the demand in the region. Therefore, offline analysis of such scenarios will be updated in the database, which will output the type of EV that is required for the *vehicle allocation of zones* step.

In the *generation of sub-zones* step, the timeframe for near-future demand estimation is intelligently used for generating sub-zones. Therefore, as mentioned in Chapter 5 when combining the elementary clusters to create a sub-zone, the timeframe for near-future demand estimation, n (for example 15 minutes) is also provided as an input. This ensures that the travel time between bus stops in a single sub-zone is always less than n . Therefore, when computing routes, the near-future demand of the entire sub-zones can be optimally utilized.

The proposed technique also uses a two-step EV allocation method to perform the *demand-aware routing* step. As mentioned in Section 7.4.2, routing is initially performed with the minimum number of EVs. The achieved user experience of the routes of this step is compared with the expected user experience. If the validation passes, the routes are passed to the *route optimization of zones* step. However, if the validation fails, the system checks if additional EVs can be allocated to sub-zones, which failed to meet the user experience criteria. The additional EVs are proportionately allocated to the sub-zones that require EVs. This is performed in the *allocate additional vehicles for sub-zones* step. Then, the *demand-aware routing* step is performed for the selected sub-zones. However, if the user experience is not met at this step, the framework outputs a request for additional supply or recommends increasing the supply.

The different colours used to highlight the geographical areas such as grey for city, green for transit hub region in Figure 7.8, exemplifies the level of parallelism that can be achieved by the proposed framework. It can be seen that a city (grey) comprises of multiple transit hub regions (green) and that each transit hub region is segmented into multiple zones (yellow) and sub-zones (blue). This is combined with intelligent workload balancing and separation of tasks for offline and online processing for citywide deployment of highly responsive demand-aware bus transit system.

7.4.5 Computing Requirements of Sub-Zones

Figure 7.9 Runtime of Generating Flexible Routes for a Zone

The responsiveness of the proposed framework for citywide deployment depends on the computation time of the online module of the demand-aware online scheduling and routing process. The experiments conducted in this thesis are performed on a PC equipped with a Xeon E5-1630 V4 CPU @ 3.70 GHz processor with 32 GB RAM. The processor has 4 cores and 8 threads [209]. Therefore, at any given time the processor can perform 8 flexible route computations of sub-zones. The proposed techniques in this thesis strives to balance the workloads in each sub-zone. Given this scenario, the experimental results revealed that for a maximum workload of 50 - 65 passengers

in a sub-zone, the computation time of flexible route generation is 20 - 24 seconds. Theoretically, this implies that the given processor can support 400 - 500 passengers in a zone, which finally gets decomposed into 8 sub-zones. This shows that the proposed framework has a high response time. The given data can be used as a reference for deployment. However, if the operator wants to achieve a higher responsiveness the configuration of the processor can be upgraded. Figure 7.9 illustrates this scenario. This shows that if the total number of sub-zones is less than 8, the workload in a sub-zone will determine the responsiveness of the system. However, if the number of sub-zones that needs to be computed is greater than 8, context switching and other overheads will impact the responsiveness of the system.

7.5 Summary

The various methods proposed in this thesis have been integrated to a framework for citywide deployment of demand-aware bus transit system. The proposed framework consisting of three processes, namely demand estimation, demand-aware online scheduling & routing and demand-aware offline scheduling can be used to generate flexible schedules and routes for simultaneously enhancing user experience and operator profitability.

Techniques have been proposed to leverage mobile phone-based and camera-based crowd estimation sensors to generate the real-time demand at bus stops. The proposed demand estimation process leverages the real-time demand estimates and public transit tap-in & tap-out data to predict the demand at bus stops for the near-future demand estimation timeframe. The near-future demand estimates enable to facilitate intelligent scheduling and routing decisions. The techniques proposed in the previous chapters have been realized using the demand-aware online scheduling & routing process and the demand-aware offline scheduling process. The former can generate the citywide flexible routes for the proposed demand-aware bus transit system while the latter generates schedules for trunk bus services.

The demand-aware online scheduling & routing process has been realized using a combination of offline and online techniques. The responsiveness of the process is improved by processing a significant component of the workload offline. Runtime of the online module has been reduced by using a two-step route computation technique that minimizes the number of vehicles used in the computations, which in-turn reduces complexity and hence runtime. A vehicle database, which recommends the most suitable type of vehicle to be used based on the demand is also incorporated to the framework. The demand-aware offline schedule generation process leverages the historical demand database that is updated in real-time. Experimental results have confirmed that for typical demands, the proposed framework can generate flexible routes significantly faster (20 - 24 sec.) depicting the high responsiveness.

Chapter 8

Conclusions and Future Directions

8.1 Conclusions

In this thesis, several techniques have been proposed to realize a responsive demand-aware bus transit system in urban cities to replace traditional fixed routes and schedules. The proposed techniques have been tailored to facilitate in the trade-off analysis between user experience and operator profitability. In addition, these techniques leverage on existing road network and hub & spoke topology of the public transit network to support large-scale deployments without adversely impacting on the overall responsiveness of the solution.

In Chapter 3, a demand-aware bus transit system using EVs has been proposed to bridge the connectivity gap of the existing feeder bus services due to fixed routes and schedules. The flexible routes are generated using the proposed three-index mathematical model based on the vehicle flow formulation. As opposed to the set partitioning method, which necessitates the need to solve the much tighter linear programming relaxations, the proposed vehicle flow formulation offers compactness, allowing general purpose optimization packages to solve them. In addition, branch and bound/cut methods based on specialized valid inequalities can help to expedite solution generation in the vehicle flow formulation. The three-index model has been shown to

scale well in terms of execution time compared to the two-index model, which helps to generate the optimal routes relatively faster. The Pareto front for simultaneously optimizing for user experience and operator profitability has been derived using the normalized weighted technique, which enables the operator to analyse solutions with varying weights for both objectives according to specific requirements. The proposed model can be relied upon to generate routes that enhance the passenger travel time (user experience) as well as VMT by the fleet (operator profitability). Experimental evaluations using openly available historical demand data on a real road network show that a fewer number of higher capacity vehicles are required to enhance operator profitability with minimal impact on user experience. However, due to the maximum travel time constraint of each passenger, the number of EVs used cannot be reduced below a threshold, thereby enforcing a lower limit for the VMT.

In Chapter 4, a heuristic routing technique has been proposed to generate near-optimal routes in real-time in order to lower compute complexity in the presence of a large number of vehicles and bus stops. The technique combines a GA and the ϵ -constraint method. Compared to existing heuristics that provide a single solution, GAs can provide multiple feasible solutions while avoiding a local minima, while the ϵ -constraint method converts an objective into a constraint during optimization, which helps to significantly reduce runtime. A local search technique has been incorporated into the GA to generate a superior initial population and genetic operators have been chosen for fast convergence. A hybrid parent selection algorithm has been proposed for balancing solution quality with diversity in the population. The runtime of the GA is governed by the selection of population size and number of iterations. It is observed that a low population size improves the runtime of the algorithm at the expense of reasonable selection of the population. Noting that a large population consumes a few hours to converge on a Xeon E5-1630 V4 CPU running at 3.70 GHz (e.g. 60 min for a population size of 200), a population size of 50 was selected for the experiments to ensure a realistic turnaround time. Tuning the number of iterations of the GA revealed that the convergence rate reduces significantly with the number of iterations. Therefore, 100 iterations were selected to perform the experiments. Experimental evaluation with

openly available real data showed that the proposed GA achieves over 97% accuracy compared to the optimal result. Most importantly, the higher accuracy of the proposed GA was achieved while lowering the runtime, which was superior to the widely-used baseline algorithm [183].

Even though, the heuristic routing technique proposed in Chapter 4 can be used for fast generation of near-optimal routes, the compute complexity of the heuristic becomes unacceptable due to the total workload of a zone. This motivated the development of the proposed directionality-centric bus transit network segmentation technique for the systematic segmentation of a zone into sub-zones to lower compute complexity in Chapter 5. Moreover, the proposed directionality & shortest-path centric clustering of bus stops into sub-zones lends well for highly responsive parallel computations as a direct result of size restricted sub-zones that are predominantly independent of each other. The strategy proposed to leverage on flexible routing to accommodate the outlier bus stops to nearest sub-zones has been shown to minimize detours. The sub-zone generation process endeavours to balance the workload of each sub-zone to ensure that flexible routing for all sub-zones are completed without notable deviations. The outlier bus stops have been taken into consideration with this awareness to determine the optimal number of EVs for each sub-zone. Experimental evaluations with openly available real-life data confirm that compared to existing widely-used unsupervised learning algorithms, which require a-priori information, the proposed technique can consistently produce routes that lower VMT by nearly 10%. It is noteworthy that the improvement in route quality is achieved while considerably improving the runtime when compared to a global heuristic routing technique. Evaluations confirm that the proposed technique is also capable of generating superior routes when time constrained. In addition, superiority of the proposed technique, both in terms of responsiveness and efficiency, is clearly evident for sporadic demands that are typical in real-life scenarios, especially during off-peak periods.

A methodology for adopting demand-aware bus transit systems for city-scale deployment is presented in Chapter 6. It was evident that the bus transit system of an

urban city is classified into feeder and trunk bus services. Feeder bus services operate with short routes surrounding the transit hub and multiple feeder bus services share overlapping route segments. In contrast, trunk bus services have long point-to-point routes, which connect two transit hubs or terminal stations. It was also established that trunk bus services tend not to overlap with each other. Systematic identification of independent transit hub regions was carried out using the properties of feeder bus services. Noting that the workload of a zone determines the responsiveness of the system, transit hub regions are further segmented into balanced and mutually exclusive zones. Each zone is then further partitioned into smaller sub-zones using directionality centric segmentation to significantly lower the compute complexity for route computation. Experimental evaluation with openly available real data showed that even though, workload balancing can improve responsiveness, zones must be generated without adversely affecting the route quality. The proposed partitioning algorithm for zone generation has been shown to recommend reasonably well-balanced zones without affecting the route quality. A demand-aware offline scheduling technique based on the properties of the trunk bus services has also been incorporated into the proposed methodology for improving long distance point-to-point connectivity.

A unified framework for citywide deployment of demand-aware bus transit system is presented in Chapter 7. The framework takes the road network of the city and existing bus transit system as inputs and leverage the near-future demand estimates obtained from mobile phone-based and camera-based crowd estimation sensors to generate the citywide demand-aware flexible routes. The separation of workloads for offline and online processing has been introduced to improve the overall responsiveness of the flexible routing system. The framework describes the systematic identification of the required number of vehicles to meet a desired level of user experience. Offline processing of demand-aware schedules for the trunk bus services has been incorporated to provide a holistic solution. Schedules for trunk bus services are generated by utilizing the demand database that is updated with the historical tap-in and tap-out data in addition to the real-time demand generation sources. Analysis confirm that the execution time for a sub-zone is about 24 seconds when implemented on a 4-core

Xeon E5-1630 V4 CPU with hyper-threading to support up to 8 sub-zones. Finally, the contributions made in this thesis have made it possible for the large scale deployment of demand-aware bus transit system that is capable of highly responsive flexible routing for enhancing user experience and operator profitability in urban cities.

8.2 Future Research Directions

1. Enhancing EV Driving Range Estimation

Accurate estimation of an EV's driving range is critical to minimize disruptions to demand-aware transit services. In this thesis, the driving range of an EV is considered based only on the travel distance. It would be of interest to refine the model to consider other factors such as air-conditioning, passenger capacity, traffic conditions and age of battery. to facilitate in the realistic estimation of an EV's driving range. Since the number of passengers assigned to an EV is known prior to route computations, an improved model for estimating EV driving range should maximize EV utilization, especially during peak hours.

2. Selection of Mixed Fleet for Optimal Scheduling

It is desirable to ensure that the number of vehicles is appropriate for an anticipated demand. It is proposed that the right mix of vehicle capacities are determined by analysing the demand-supply patterns to minimize over supply. Provision to mix high-capacity and low-capacity vehicles to maximise effective utilization should enhance user experience. A mixed fleet containing vehicles of lower seating capacity can respond to lack of demand during off-peak periods. However, implications of deploying low-capacity vehicles in terms of operating costs due to increased number of vehicles, VMT of the fleet and the requirement for additional drivers must be considered in order to examine the trade-off between user experience and operating costs. Historical demand data of the existing public transit system can be relied upon to identify the mixed fleet for optimal scheduling.

3. Scheduling Mixed Fleet for Flexible Routing

While a heterogeneous fleet of EV provides for flexibility without increasing VMT, it requires the development of a suitable method for determining the appropriate combinations of heterogeneous fleet for the flexible routes of a sub-zone in real-time. Grouping decisions should be governed by the demand distribution in each sub-zone, to facilitate in the optimal utilization of available fleet, especially when demand varies notably among bus stops. For example, a bus stop with many passengers can benefit from a large capacity EV, whereas other bus stops in the sub-zone can be served by low-capacity EVs. Thus, it would be interesting to explore how an optimal scheduling of the pre-assigned heterogeneous fleet of EVs can be realized on-the-fly.

4. Traffic Incidents-aware Flexible Routing

Traffic congestions due to accidents and vehicle breakdowns are a common phenomenon in urban road networks. Noting that it will be impractical to respond to congestions once after scheduling for flexible routing, recurring congestion patterns of historical data can be considered during scheduling to minimise disruptions. On-the-fly identification of alternate routes, when possible, should also be supported to overcome exigencies due to traffic incidents. In addition, alternate flexible routes can be supported by additional EVs, when appropriate, to minimize passenger travel time by addressing only those affected bus stops.

5. Combining Neighbouring Sub-Zones During Off-Peak Periods

In Chapter 5, sub-zones are generated by directionality-centric segmentation of the bus transit network. Even though, generating the routes in each sub-zone improves responsiveness, quality of the solutions may marginally deteriorate due to the segmentation of the network. Noting that not all the bus stops of a sub-zone are likely to be considered during off-peak periods, combining two adjacent sub-zones may yield optimal routes while lowering operating costs. This should also avail more EVs for charging and/or maintenance.

6. Automating Sub-Zone Generation & Demand-aware EV Allocation

The framework proposed in this thesis has paved the way for large-scale deployment of demand-aware bus transit system. The proposed framework is based on the systematic identification of independent transit hub regions in urban cities that are further decomposed into zones to balance the workload. Zones are further segmented into sub-zones by directionality-centric shortest-path based clustering, followed by demand-aware EV allocation. It should be possible to automate the sub-zone generation and demand aware EV allocation processes. In addition, routes of a sub zone can be generated using the fully automated proposed heuristic routing technique. This should facilitate in the detailed evaluation of city-scale adoption of demand aware flexible routing for bus transit systems prior to real-life deployment.

7. Restricted Flexible Routing for Neighbouring Trunk Bus Services

The techniques proposed in Chapter 6 for improving long distance point-to-point connectivity proposes demand-aware offline schedules for trunk bus services. The methods proposed for demand aggregation can be integrated to allow for the periodic adjustment of bus arrival times. In addition, flexible routing for trunk bus services can be explored by combining nearby trunk bus services of similar range. The techniques proposed in Chapter 5 can be used to identify neighbouring bus services with significant overlaps in routes. This should pave the way for overall reduction in the supply of buses and operating costs without affecting user experience during off-peak hours.

References

- [1] Stefan Knupfer and Vadim Pokotilo and Jonathan Woetzel. Elements of success: Urban transportation systems of 24 global cities. https://www.mckinsey.com/~media/McKinsey/Business%20Functions/Sustainability/Our%20Insights/Elements%20of%20success%20Urban%20transportation%20systems%20of%2024%20global%20cities/Urban-transportation-systems_e-versions.ashx, 2018.
- [2] Land Transport Authority, Singapore. Public Transport Utilisation - Average Daily Public Transport Ridership. <https://data.gov.sg/dataset/public-transport-utilisation-average-public-transport-ridership>, 2020.
- [3] SBS Transit, Singapore. Average Daily Ridership in Rail Transit. <https://www.sbstransit.com.sg/ridership>, 2020.
- [4] Ankita Bhutani and Prasenjit Saha. Global Mobility on-Demand Market worth \$250bn by 2026. <https://www.gminsights.com/pressrelease/mobility-on-demand-mod-market>, 2019.
- [5] Ministry of Transport, Singapore. Update on Bus Routes Planning. <https://bit.ly/2X0gTzr>, 2010.
- [6] Gopinath Menon and Loh Chow Kuang. Lessons from Bus Operations. <https://www.ptc.gov.sg/docs/default-source/publications-and-papers/lessonsfrombusoperationsrev5.pdf>, 2006.
- [7] Alex Gibson. Eliminating Public Transit’s First-Mile/Last-Mile Problem. <http://transloc.com/eliminating-public-transits-first-milelast-mile-problem/>, 2016.
- [8] Jennifer Toole. Update of the AASHTO Guide for the Planning, Design, and Operation of Pedestrian Facilities. [http://onlinepubs.trb.org/onlinepubs/nchrp/docs/NCHRP20-07\(263\)_FR.pdf](http://onlinepubs.trb.org/onlinepubs/nchrp/docs/NCHRP20-07(263)_FR.pdf), 2010.
- [9] Roy Ka-Wei Lee and Tin Seong Kam. Time-Series Data Mining in Transportation: A Case Study on Singapore Public Train Commuter Travel Patterns. *International Journal of Engineering and Technology*, 6(5):431–438, 2014.
- [10] Lilah M. Besser and Andrew L. Dannenberg. Walking to Public Transit. *American Journal of Preventive Medicine*, 29(4):273–280, 2005.

- [11] Transit Oriented Development Institute. Transit Oriented Development. <http://www.tod.org/>, 2017.
- [12] Urban Redevelopment Authority Singapore. Walk2Ride Scheme. <https://www.ur.gov.sg/ms/walkandcycle/about/walk2ride>, 2015.
- [13] Nabilah Naharudin, Mohd Sansui S. Ahamad, and Ahmad Farhan Mohd Sadullah. Optimizing Pedestrian-Friendly Walking Path for the First and Last Mile Transit Journey by Using the Analytical Network Process (anp) Decision Model and GIS Network Analysis. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLII-4/W5:137–144, 2017.
- [14] Federal Highway Administration U.S. Department of Transportation. Summary of Travel Trends. 2017 National Household Travel Survey. https://nhts.ornl.gov/assets/2017_nhts_summary_travel_trends.pdf, 2017.
- [15] Menon Gopinath and Loh Chow Kuang. Lessons from Bus Operations. <https://www.ptc.gov.sg/docs/default-source/publications-and-papers/lessonsfrombusoperationsrev5.pdf>, 2006.
- [16] Paul DeMaio. Bike-sharing: history, impacts, models of provision, and future. *Journal of Public Transportation*, 12(4):41–56, 2009.
- [17] Susan Shaheen and Stacey Guzman. Worldwide Bikesharing. <http://tsrc.berkeley.edu/worldwide%20bikesharing>, 2011.
- [18] Elliot Fishman. Bikeshare: A Review of Recent Literature. *Transport Reviews*, 36(1):92–113, 2016.
- [19] Matthew Curtis Lesh. Innovative Concepts in First-Last Mile Connections to Public Transportation. In *Third International Conference on Urban Public Transportation Systems*, 2013.
- [20] Ashwani Kumar, Viet Anh Nguyen, and Kwong Meng Teo. Commuter cycling policy in Singapore: a farecard data analytics based approach. *Annals of Operations Research*, 236(1):57–73, 2016.
- [21] Dowling Julia Robyn, D. Irwin, J. Faulksb Ian, and Howitt Richie. Use of personal mobility devices for first-and-last mile travel: The Macquarie Ryde trial. In *Proceedings of the 2015 Australasian Road Safety Conference*, ARSC’16, 2015.
- [22] Shuo Ma and Ouri Wolfson. Analysis and Evaluation of the Slugging Form of Ridesharing. In *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, SIGSPATIAL’13, pages 64–73, New York, NY, USA, 2013. ACM.
- [23] David LeBlanc. About Slugging. http://www.slug-lines.com/Slugging/About_slugging.asp, 2017.

- [24] European Union. Demand Responsive Transportation: A Policy Brief from the Policy Learning Platform on Low-carbon economy. https://www.interregeurope.eu/fileadmin/user_upload/plp_uploads/policy_briefs/2018-06-27_Policy_Brief_Demand_Responsive_Transport.pdf, 2020.
- [25] Uber. UberPool: Together, we save. <https://www.uber.com/us/en/ride/uberpool/>.
- [26] Spencer Woodman. WELCOME TO UBERVILLE. <https://www.theverge.com/2016/9/1/12735666/uber-altamonte-springs-fl-public-transportation-taxi-system>, 2016.
- [27] Sand Logan, Samantha Beckerman, and Catherine Cox Blair. First and Last Mile Connections: New Mobility. <http://on.nrdc.org/2utPxUd>, 2016.
- [28] Bin Cao, Louai Alarabi, Mohamed F. Mokbel, and Anas Basalamah. SHAREK: A Scalable Dynamic Ride Sharing System. In *16th IEEE International Conference on Mobile Data Management*, volume 1, pages 4–13, 2015.
- [29] P. Larrañaga, C.M.H. Kuijpers, R.H. Murga, I. Inza, and S. Dizdarevic. Genetic Algorithms for the Travelling Salesman Problem: A Review of Representations and Operators. *Artificial Intelligence Review*, 13(2):129–170, 1999.
- [30] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by Simulated Annealing. *Science*, 220(4598):671–680, 1983.
- [31] R. M. Brady. Optimization strategies gleaned from biological evolution. *Nature*, 317:804–806, 1985.
- [32] J. J. Hopfield and D. W. Tank. "Neural" Computation of Decisions in Optimization Problems. *Biological Cybernetics*, 52(3):141–152, 1985.
- [33] G. B. Dantzig and J. H. Ramser. The Truck Dispatching Problem. *Management Science*, 6(1):80–91, 1959.
- [34] G. Clarke and J. W. Wright. Scheduling of Vehicles from a Central Depot to a Number of Delivery Points. *Operations Research*, 12(4):568–581, 1964.
- [35] Amos Levin. Scheduling and Fleet Routing Models for Transportation Systems. *Transportation Science*, 5(3):232–255, 1971.
- [36] David H Marks and Robert Stricker. Routing for Public Service Vehicles. *Journal of the Urban Planning and Development Division*, 97(2):165–178, 1971.
- [37] S. Eilon, C. D. T. Watson-Gandy, N. Christofides, and R. de Neufville. Distribution Management-Mathematical Modelling and Practical Analysis. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-4(6):589–589, 1974.

- [38] B. L. Golden, T. L. Magnanti, and H. Q. Nguyen. Implementing vehicle routing algorithms. *Networks*, 7(2):113–148, 1977.
- [39] Jairo R. Montoya-Torres, Julián López Franco, Santiago Nieto Isaza, Heriberto Felizzola Jiménez, and Nilson Herazo-Padilla. A literature review on the vehicle routing problem with multiple depots. *Computers & Industrial Engineering*, 79(Supplement C):115 – 129, 2015.
- [40] Thomas M. Cook and Robert A. Russell. A Simulation and Statistical Analysis of Stochastic Vehicle Routing with Timing Constraints. *Decision Sciences*, 9(4):673–687, 1978.
- [41] Bruce L. Golden and James R. Yee. A Framework For Probabilistic Vehicle Routing. *IIE Transactions*, 11(2):109–112, 1979.
- [42] Marius M. Solomon. Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints. *Operations Research*, 35(2):254–265, 1987.
- [43] Warren B. Powell. A Stochastic Model of the Dynamic Vehicle Allocation Problem. *Transportation Science*, 20(2):117–129, 1986.
- [44] A. Larsen, O. Madsen, and M. Solomon. Partially dynamic vehicle routing—models and algorithms. *Journal of the Operational Research Society*, 53(6):637–646, 2002.
- [45] Bernhard Fleischmann, Stefan Gnutzmann, and Elke Sandvoß. Dynamic Vehicle Routing Based on Online Traffic Information. *Transportation Science*, 38(4):420–433, 2004.
- [46] A. Serdar Tasan and Mitsuo Gen. A genetic algorithm based approach to vehicle routing problem with simultaneous pick-up and deliveries. *Computers & Industrial Engineering*, 62(3):755 – 761, 2012.
- [47] Ann Melissa Campbell and Jill Hardin Wilson. Forty years of periodic vehicle routing. *Networks*, 63(1):2–15, 2014.
- [48] Jorrit J. Bakker. *Contesting range anxiety: The role of electric vehicle charging infrastructure in the transportation transition*. PhD thesis, Eindhoven University of Technology, 2011.
- [49] Sevgi Erdoğan and Elise Miller-Hooks. A Green Vehicle Routing Problem. *Transportation Research Part E: Logistics and Transportation Review*, 48(1):100 – 114, 2012.
- [50] Ángel Felipe, M. Teresa Ortuño, Giovanni Righini, and Gregorio Tirado. A heuristic approach for the green vehicle routing problem with multiple technologies and partial recharges. *Transportation Research Part E: Logistics and Transportation Review*, 71:111 – 128, 2014.

- [51] Michael Schneider, Andreas Stenger, and Dominik Goeke. The Electric Vehicle-Routing Problem with Time Windows and Recharging Stations. *Transportation Science*, 48(4):500–520, 2014.
- [52] Gerhard Hiermann, Jakob Puchinger, Stefan Ropke, and Richard F. Hartl. The Electric Fleet Size and Mix Vehicle Routing Problem with Time Windows and Recharging Stations. *European Journal of Operational Research*, 252(3):995 – 1018, 2016.
- [53] Tomislav Erdelić and Tonci Caric. A Survey on the Electric Vehicle Routing Problem: Variants and Solution Approaches. *Journal of Advanced Transportation*, 2019:1–48, 2019.
- [54] Busra Gulnihan Dascioglu and Gulfem Tuzkaya. A Literature Review for Hybrid Vehicle Routing Problem. In Fethi Calisir, Emre Cevikcan, and Hatice Camgoz Akdag, editors, *Industrial Engineering in the Big Data Era*, pages 249–257, Cham, 2019. Springer International Publishing.
- [55] J. K. Lenstra and A. H. G. Rinnooy Kan. Complexity of vehicle routing and scheduling problems. *Networks*, 11(2):221–227, 1981.
- [56] Gilbert Laporte and Yves Nobert. Exact Algorithms for the Vehicle Routing Problem. In *Surveys in Combinatorial Optimization*, volume 132 of *North-Holland Mathematics Studies*, pages 147 – 184. North-Holland, 1987.
- [57] Gilbert Laporte. The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, 59(3):345 – 358, 1992.
- [58] P. D. Wasserman. *Neural Computing: Theory and Practice*. Van Nostrand Reinhold Co., New York, NY, USA, 1989.
- [59] M. Dorigo, V. Maniezzo, and A. Colorni. Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 26(1):29–41, 1996.
- [60] Emile Aarts and Jan K. Lenstra, editors. *Local Search in Combinatorial Optimization*. John Wiley & Sons, Inc., New York, NY, USA, 1997.
- [61] Fred Glover and Manuel Laguna. *Tabu Search*. Kluwer Academic Publishers, Norwell, MA, USA, 1997.
- [62] Michel Gendreau, Jean-Yves Potvin, Olli Bräumlaysy, Geir Hasle, and Arne Løkketangen. *Metaheuristics for the Vehicle Routing Problem and Its Extensions: A Categorized Bibliography*, pages 143–169. Springer US, Boston, MA, USA, 2008.
- [63] Gilbert Laporte. Scheduling issues in vehicle routing. *Annals of Operations Research*, 236(2):463–474, 2016.

- [64] Harilaos N. Psaraftis, Min Wen, and Christos A. Kontovas. Dynamic vehicle routing problems: Three decades and counting. *Networks*, 67(1):3–31, 2016.
- [65] Sin C. Ho, W.Y. Szeto, Yong-Hong Kuo, Janny M.Y. Leung, Matthew Petering, and Terence W.H. Tou. A survey of dial-a-ride problems: Literature review and recent developments. *Transportation Research Part B: Methodological*, 111:395 – 421, 2018.
- [66] Jean-François Cordeau and Gilbert Laporte. The dial-a-ride problem: models and algorithms. *Annals of Operations Research*, 153(1):29–46, 2007.
- [67] Sophie N. Parragh. Introducing heterogeneous users and vehicles into models and algorithms for the dial-a-ride problem. *Transportation Research Part C: Emerging Technologies*, 19(5):912 – 930, 2011.
- [68] Renaud Masson, Fabien Lehuédé, and Olivier Péton. An Adaptive Large Neighborhood Search for the Pickup and Delivery Problem with Transfers. *Transportation Science*, 47(3):344–355, 2013.
- [69] Jean-François Cordeau and Gilbert Laporte. A tabu search heuristic for the static multi-vehicle dial-a-ride problem. *Transportation Research Part B: Methodological*, 37(6):579 – 594, 2003.
- [70] Harilaos N. Psaraftis. A Dynamic Programming Solution to the Single Vehicle Many-to-Many Immediate Request Dial-a-Ride Problem. *Transportation Science*, 14(2):130–154, 1980.
- [71] Harilaos N. Psaraftis. An Exact Algorithm for the Single Vehicle Many-to-Many Dial-A-Ride Problem with Time Windows. *Transportation Science*, 17(3):351–357, 1983.
- [72] Thomas R. Sexton and Lawrence D. Bodin. Optimizing Single Vehicle Many-to-Many Operations with Desired Delivery Times: I. Scheduling. *Transportation Science*, 19(4):378–410, 1985.
- [73] Thomas R. Sexton and Lawrence D. Bodin. Optimizing Single Vehicle Many-to-Many Operations with Desired Delivery Times: II. Routing. *Transportation Science*, 19(4):411–435, 1985.
- [74] Jacques Desrosiers, Yvan Dumas, and François Soumis. A Dynamic Programming Solution of the Large-Scale Single-Vehicle Dial-A-Ride Problem with Time Windows. *American Journal of Mathematical and Management Sciences*, 6(3-4):301–325, 1986.
- [75] Jang-Jei Jaw, Amedeo R. Odoni, Harilaos N. Psaraftis, and Nigel H.M. Wilson. A heuristic algorithm for the multi-vehicle advance request dial-a-ride problem with time windows. *Transportation Research Part B: Methodological*, 20(3):243 – 257, 1986.

- [76] Jean-François Cordeau. A Branch-and-Cut Algorithm for the Dial-a-Ride Problem. *Operations Research*, 54(3):573–586, 2006.
- [77] Stefan Ropke, Jean-François Cordeau, and Gilbert Laporte. Models and branch-and-cut algorithms for pickup and delivery problems with time windows. *Networks*, 49(4):258–272, 2007.
- [78] Zhihai Xiang, Chengbin Chu, and Haoxun Chen. A fast heuristic for solving a large-scale static dial-a-ride problem under complex constraints. *European Journal of Operational Research*, 174(2):1117 – 1139, 2006.
- [79] Michel Gendreau, Gilbert Laporte, and Frédéric Semet. A dynamic model and parallel tabu search heuristic for real-time ambulance relocation. *Parallel Computing*, 27(12):1641 – 1653, 2001.
- [80] Michel Gendreau, François Guertin, Jean-Yves Potvin, and René Séguin. Neighborhood search heuristics for a dynamic vehicle dispatching problem with pickups and deliveries. *Transportation Research Part C: Emerging Technologies*, 14(3):157 – 174, 2006.
- [81] Dusan Teodorovic and Gordana Radivojevic. A fuzzy logic approach to dynamic dial-a-ride problem. *Fuzzy Sets and Systems*, 116(1):23 – 33, 2000.
- [82] Luca Coslovich, Raffaele Pesenti, and Walter Ukovich. A two-phase insertion technique of unexpected customers for a dynamic dial-a-ride problem. *European Journal of Operational Research*, 175(3):1605 – 1615, 2006.
- [83] Thomas Hanne, Teresa Melo, and Stefan Nickel. Bringing Robustness to Patient Flow Management Through Optimized Patient Transports in Hospitals. *INFORMS Journal on Applied Analytics*, 39(3):241–255, 2009.
- [84] Alexandre Beaudry, Gilbert Laporte, Teresa Melo, and Stefan Nickel. Dynamic transportation of patients in hospitals. *OR Spectrum*, 32(1):77–107, 2010.
- [85] Carl H. Häll and Anders Peterson. Improving paratransit scheduling using ruin and recreate methods. *Transportation Planning and Technology*, 36(4):377–393, 2013.
- [86] K.I. Wong, A.F. Han, and C.W. Yuen. On dynamic demand responsive transport services with degree of dynamism. *Transportmetrica A: Transport Science*, 10(1):55–73, 2014.
- [87] Douglas O. Santos and Eduardo C. Xavier. Taxi and Ride Sharing: A Dynamic Dial-a-Ride Problem with Money as an Incentive. *Expert Systems with Applications*, 42(19):6728 – 6737, 2015.
- [88] Oli B. G. Madsen, Hans F. Ravn, and Jens Moberg Rygaard. A heuristic algorithm for a dial-a-ride problem with time windows, multiple capacities, and multiple objectives. *Annals of Operations Research*, 60(1):193–208, 1995.

- [89] Paolo Toth and Daniele Vigo. Heuristic Algorithms for the Handicapped Persons Transportation Problem. *Transportation Science*, 31(1):60–71, 1997.
- [90] K. I. Wong and Michael G. H. Bell. Solution of the Dial-a-Ride Problem with multi-dimensional capacity constraints. *International Transactions in Operational Research*, 13(3):195–208, 2006.
- [91] Cristián E. Cortés, Martín Matamala, and Claudio Contardo. The pickup and delivery problem with transfers: Formulation and a branch-and-cut solution method. *European Journal of Operational Research*, 200(3):711 – 724, 2010.
- [92] Line Blander Reinhardt, Tommy Clausen, and David Pisinger. Synchronized dial-a-ride transportation of disabled passengers at airports. *European Journal of Operational Research*, 225(1):106 – 117, 2013.
- [93] Paolo Detti, Francesco Papalini, and Garazi Zabalo Manrique de Lara. A multi-depot dial-a-ride problem with heterogeneous vehicles and compatibility constraints in healthcare. *Omega*, 70:1 – 14, 2017.
- [94] Yuan Qu and Jonathan F. Bard. The heterogeneous pickup and delivery problem with configurable vehicle capacity. *Transportation Research Part C: Emerging Technologies*, 32:1 – 20, 2013.
- [95] Alberto Colorni and Giovanni Righini. Modeling and optimizing dynamic dial-a-ride problems. *International Transactions in Operational Research*, 8(2):155–166, 2001.
- [96] Majid Aldaihani and Maged M. Dessouky. Hybrid scheduling methods for paratransit operations. *Computers & Industrial Engineering*, 45(1):75 – 96, 2003.
- [97] Brahim Rekiek, Alain Delchambre, and Hussain Aziz Saleh. Handicapped Person Transportation: An application of the Grouping Genetic Algorithm. *Engineering Applications of Artificial Intelligence*, 19(5):511 – 520, 2006.
- [98] Rémy Chevrier, Arnaud Liefoghe, Laetitia Jourdan, and Clarisse Dhaenens. On optimizing a demand responsive transport with an evolutionary multi-objective approach. In *13th International IEEE Conference on Intelligent Transportation Systems*, pages 575–580, 2010.
- [99] Rémy Chevrier, Arnaud Liefoghe, Laetitia Jourdan, and Clarisse Dhaenens. Solving a dial-a-ride problem with a hybrid evolutionary multi-objective approach: Application to demand responsive transport. *Applied Soft Computing*, 12(4):1247 – 1258, 2012.
- [100] R. S. Mendes, E. F. Wanner, J. F. M. Sarubbi, and F. V. C. Martins. Optimization of the vehicle routing problem with demand responsive transport using the NSGA-II algorithm. In *19th IEEE International Conference on Intelligent Transportation Systems*, pages 2657–2662, 2016.

- [101] Renan J. S. Viana, André G. Santos, Flávio V. C. Martins, and Elizabeth F. Wanner. Optimization of a Demand Responsive Transport Service Using Multi-objective Evolutionary Algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 2064–2067, 2019.
- [102] Arthur Scheltes and Gonçalo Homem de Almeida Correia. Exploring the use of automated vehicles as last mile connection of train trips through an agent-based simulation model: An application to Delft, Netherlands. *International Journal of Transportation Science and Technology*, 6(1):28 – 41, 2017.
- [103] Mohamed Amine Masmoudi, Manar Hosny, Emrah Demir, Konstantinos N. Genikomsakis, and Naoufel Cheikhrouhou. The dial-a-ride problem with electric vehicles and battery swapping stations. *Transportation Research Part E: Logistics and Transportation Review*, 118:392 – 420, 2018.
- [104] M. Zhu, X. Liu, and X. Wang. An Online Ride-Sharing Path-Planning Strategy for Public Vehicle Systems. *IEEE Transactions on Intelligent Transportation Systems*, 20(2):616–627, 2019.
- [105] Merve Keskin, Gilbert Laporte, and Bülent Çatay. Electric Vehicle Routing Problem with Time-Dependent Waiting Times at Recharging Stations. *Computers & Operations Research*, 107:77 – 94, 2019.
- [106] Claudia Bongiovanni, Mor Kaspi, and Nikolas Geroliminis. The electric autonomous dial-a-ride problem", journal = "Transportation Research Part B: Methodological. 122:436 – 456, 2019.
- [107] Zhihai Xiang, Chengbin Chu, and Haoxun Chen. The study of a dynamic dial-a-ride problem under time-dependent and stochastic environments. *European Journal of Operational Research*, 185(2):534 – 551, 2008.
- [108] Esa Hyytiä, Aleksi Penttinen, and Reijo Sulonen. Non-myopic vehicle and route selection in dynamic DARP with travel time and workload objectives. *Computers & Operations Research*, 39(12):3021 – 3030, 2012.
- [109] M. Schilde, K.F. Doerner, and R.F. Hartl. Integrating stochastic time-dependent travel speed in solution methods for the dynamic dial-a-ride problem. *European Journal of Operational Research*, 238(1):18 – 30, 2014.
- [110] Baoxiang Li, Dmitry Krushinsky, Tom Van Woensel, and Hajo A. Reijers. The Share-a-Ride problem with stochastic travel times and stochastic delivery locations. *Transportation Research Part C: Emerging Technologies*, 67:95 – 108, 2016.
- [111] S. C. Nagavarapu et al. Development of a simulation platform to implement vehicle routing algorithms for large scale fleet management systems. In *ITSC*, 2017.
- [112] Matti Pursula. Simulation of traffic systems - an overview. *Journal of Geographic Information and Decision Analysis*, 1999.

- [113] Tom V. Mathew. Microscopic Traffic Simulation. <https://www.civil.iitb.ac.in/tvm/nptel/tse1nw37.pdf>, 2017.
- [114] Gordon D. B. Cameron and Gordon I. D. Duncan. Paramics—parallel microscopic simulation of road traffic. *The Journal of Supercomputing*, 1996.
- [115] B. Liu et al. Vgsim: An integrated networking and microscopic vehicular mobility simulation platform. *IEEE Comm. Magazine*, 2009.
- [116] Daniel Krajzewicz et al. Recent development and applications of SUMO - Simulation of Urban MObility. *International Journal On Advances in Systems and Measurements*, 2012.
- [117] Aimsun. Move Brilliantly. <https://www.aimsun.com/>, 2018.
- [118] J. Macedo et al. A hla-based multi-resolution approach to simulating electric vehicles in simulink and sumo. In *ITSC*, 2013.
- [119] T. Mayer et al. An open-source discrete event simulator for rich vehicle routing problems. In *ITSC*, 2016.
- [120] J. Bischoff et al. City-wide shared taxis: A simulation study in berlin. In *ITSC*, 2017.
- [121] D. Fiedler, M. Čertický, J. Alonso-Mora, and M. Čáp. The impact of ridesharing in mobility-on-demand systems: Simulation case study in prague. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 1173–1178, 2018.
- [122] Grab Singapore. GrabShare: Share the ride, and the fare. <https://www.grab.com/sg/share/>.
- [123] RYDE Singapore. RYDE: Connecting People. <https://www.rydesharing.com/sg/home/>.
- [124] SWAT Singapore. The smart transport experience for your employees. <https://swatmobile.io/>.
- [125] Shotl. Full Seats, better streets. <https://shotl.com/>.
- [126] Bridj. Out of the box shared transportation. <http://www.bridj.com/>.
- [127] Via. Welcome to Via. We ride together. <https://ridewithvia.com/>.
- [128] Irina Ioachim, Jacques Desrosiers, Yvan Dumas, Marius Solomon, and Daniel Villeneuve. A Request Clustering Algorithm in Door-to-Door Transportation. *Transportation Science*, 29:63–78, 1995.
- [129] Luca Quadrifoglio, Maged Dessouky, and Fernando Ordóñez. A simulation study of demand responsive transit system design. *Transportation Research Part A: Policy and Practice*, 42:718–737, 2008.

- [130] Han Zheng, Junhua Chen, Xingchen Zhang, and Zixian Yang. Designing a New Shuttle Service to Meet Large-Scale Instantaneous Peak Demands for Passenger Transportation in a Metropolitan Context: A Green, Low-Cost Mass Transport Option. *Sustainability*, 11(18), 2019.
- [131] L.D Bodin and T. Sexton. The multi-vehicle subscriber dial-a-ride problem. *TIMS Studies in Management Science*, 26:73–86, 1986.
- [132] Jacques Desrosiers, Y Dumas, F Soumis, S Taillefer, and Daniel Villeneuve. An algorithm for mini-clustering in handicapped transport. 1991.
- [133] R M Jorgensen, J. Larsen, and K B Bergvinsdottir. Solving the Dial-a-Ride problem using genetic algorithms. *Journal of the Operational Research Society*, 58(10):1321–1331, 2007.
- [134] Doris Saez, Cristián Cortés, and Alfredo Núñez. Hybrid adaptive predictive control for the multi-vehicle dynamic pick-up and delivery problem based on genetic algorithms and fuzzy clustering. *Computers & Operations Research*, 35:3412–3438, 2008.
- [135] Chung-Wei Shen and Luca Quadrifoglio. Evaluating Centralized versus Decentralized Zoning Strategies for Metropolitan ADA Paratransit Services. *Journal of Transportation Engineering*, 139:524–532, 2013.
- [136] Dominik Pelzer, Jiajian Xiao, Daniel Zehe, Michael Lees, Alois Knoll, and Heiko Aydt. A Partition-Based Match Making Algorithm for Dynamic Ridesharing. *IEEE Transactions on Intelligent Transportation Systems*, 16:1–12, 2015.
- [137] Shukai Chen, Hua Wang, and Qiang Meng. Solving the first-mile ridesharing problem using autonomous vehicles. *Computer-Aided Civil and Infrastructure Engineering*, 35(1):45–60, 2020.
- [138] S. E. Comert, H. R. Yazgan, S. Kır, and F. Yener. A cluster first-route second approach for a capacitated vehicle routing problem: a case study. *International Journal of Procurement Management (IJPM)*, 11:399, 2018.
- [139] Meghna Lowalekar, Pradeep Varakantham, and Patrick Jaillet. ZAC: A Zone Path Construction Approach for Effective Real-Time Ridesharing. In *International Conference on Automated Planning and Scheduling*, 2019.
- [140] K. Uchimura, H. Takahashi, and T. Saitoh. Demand responsive services in hierarchical public transportation system. *IEEE Transactions on Vehicular Technology*, 51(4):760–766, 2002.
- [141] K. Tsubouchi, K. Hiekata, and H. Yamato. Scheduling Algorithm for On-Demand Bus System. In *6th International Conference on Information Technology: New Generations*, pages 189–194, 2009.
- [142] K. Tsubouchi, H. Yamato, and K. Hiekata. Innovative on-demand bus system in Japan. *IET Intelligent Transport Systems*, 4(4):270–279, 2010.

- [143] Kousuke Kawamura and Naoto Mukai. Optimization of Transport Plan for On-Demand Bus System Using Electrical Vehicles. In Juan D. Velásquez, Sebastián A. Ríos, Robert J. Howlett, and Lakhmi C. Jain, editors, *Knowledge-Based and Intelligent Information and Engineering Systems*, pages 656–663, Berlin, Heidelberg, 2009. Springer.
- [144] K. Uehara, Y. Akamine, N. Toma, M. Nerome, and S. Endo. A proposal of a transport system connecting demand responsive bus with mass transit. In *2014 IEEE International Conference on Consumer Electronics*, pages 143–144, 2014.
- [145] K. Uehara, Y. Akamine, N. Toma, M. Nerome, and S. Endo. Evaluation of a Hierarchical Cooperative Transport System Using Demand Responsive Bus on a Dynamic Simulation. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E99.A(1):310–318, 2016.
- [146] L. Miguel Martínez, José Manuel Viegas, and Tomás Eiró. Formulating a New Express Minibus Service Design Problem as a Clustering Problem. *Transportation Science*, 49(1):85–98, 2015.
- [147] Nikola Marković, Rahul Nair, Paul Schonfeld, Elise Miller-Hooks, and Matthew Mohebbi. Optimizing dial-a-ride services in Maryland: Benefits of computerized routing and scheduling. *Transportation Research Part C: Emerging Technologies*, 55:156 – 165, 2015.
- [148] Tao Liu, Avishai Ceder, Romain Bologna, and Benjamin Cabantous. Commuting by Customized Bus: A Comparative Analysis with Private Car and Conventional Public Transport in Two Cities. *Journal of Public Transportation*, 19:55–74, 2016.
- [149] Lu Tong, Leishan Zhou, Jiangtao Liu, and Xuesong Zhou. Customized bus service design for jointly optimizing passenger-to-vehicle assignment and vehicle routing. *Transportation Research Part C: Emerging Technologies*, 85:451 – 475, 2017.
- [150] Claudia Archetti, M. Grazia Speranza, and Dennis Weyland. A simulation study of an on-demand transportation system. *International Transactions in Operational Research*, 25(4):1137–1161, 2018.
- [151] Hai Wang. Routing and Scheduling for a Last-Mile Transportation System. *Transportation Science*, 53(1):131–147, 2019.
- [152] Arvind Raghunathan, David Bergman, John Hooker, Thiago Serra, and Shingo Kobori. The Integrated Last-Mile Transportation Problem (ILMTP). In *International Conference on Automated Planning and Scheduling*, 2018.
- [153] Olli Sulopuiso. Why helsinki’s on-demand bus service failed. <https://www.citylab.com/transportation/2016/03/helsinki-on-demand-bus-service-kutsuplus/472545/>.

- [154] Kari Rissanen. Kutsuplus – final report. Technical report, Helsinki Regional Transport Authority (HSL), 2016.
- [155] University of Duke. Duke Vans. <https://parking.duke.edu/buses/duke-vans>, 2017.
- [156] Georgia Tech University. Stingerette, Safe Ride Home & Paratransit . <http://pts.gatech.edu/stingerette-safe-ride-home-paratransit>.
- [157] Thilina Perera, Alok Prakash, and Thambipillai Srikanthan. A Scalable Heuristic Algorithm for Demand Responsive Transportation for First Mile Transit. In *IEEE 21st International Conference on Intelligent Engineering Systems (INES)*, pages 157–162, 2017.
- [158] Thilina Perera, Alok Prakash, and Thambipillai Srikanthan. A Hybrid Methodology for Optimal Fleet Management in an Electric Vehicle Based Flexible Bus Service. In *15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, pages 331–336, 2018.
- [159] Lin Zhaowei. Fingerprint of a bus route. <https://blog.data.gov.sg/fingerprint-of-a-bus-route-73e5be53dcf0>, 2015.
- [160] United States Environmental Protection Agency. Sources of Greenhouse Gas Emissions. <https://www.epa.gov/ghgemissions/sources-greenhouse-gas-emissions>, 2014.
- [161] European Environmental Agency. Greenhouse gas emissions from transport. <https://www.eea.europa.eu/data-and-maps/indicators/transport-emissions-of-greenhouse-gases>, 2018.
- [162] Möhner Matthias. Driving ban for diesel-powered vehicles in major cities: an appropriate penalty for exceeding the limit value for nitrogen dioxide? *International Archives of Occupational and Environmental Health*, 91(4):373–376, 2018.
- [163] The Guardian. France to ban sales of petrol and diesel cars by 2040. <https://www.theguardian.com/business/2017/jul/06/france-ban-petrol-diesel-cars-2040-emmanuel-macron-volvo>, 2017.
- [164] Land Transport Guru, Singapore. Electric Buses in Singapore. <https://landtransportguru.net/electric-buses-in-singapore/>, 2018.
- [165] Nichola Groom, Reuters. U.S. transit agencies cautious on electric buses despite bold forecasts. <https://reut.rs/2nSSZq5>, 2017.
- [166] Ralf Sulzbach and Jennifer Bilatscheck. EV charging anxiety is the new range anxiety. <https://autovistagroup.com/news-and-insights/ev-charging-anxiety-new-range-anxiety/>, 2020.

- [167] Paolo Toth and Daniele Vigo. Models, relaxations and exact approaches for the capacitated vehicle routing problem. *Discrete Applied Mathematics*, 123(1):487 – 512, 2002.
- [168] Pedro Munari, Twan Dollevoet, and Remy Spliet. A generalized formulation for vehicle routing problems, 2016.
- [169] Quan Lu and Maged Dessouky. An Exact Algorithm for the Multiple Vehicle Pickup and Delivery Problem. *Transportation Science*, 38(4):503–514, 2004.
- [170] R. Chevrier, P. Canalda, P. Chatonnay, and D. Josselin. Comparison of three algorithms for solving the convergent demand responsive transportation problem. In *IEEE Intelligent Transportation Systems Conference*, pages 1096–1101, 2006.
- [171] Julie Paquette, Jean-François Cordeau, Gilbert Laporte, and Marta M.B. Pascoal. Combining multicriteria analysis and tabu search for dial-a-ride problems. *Transportation Research Part B: Methodological*, 52:1 – 16, 2013.
- [172] Marco Diana, Luca Quadrioglio, and Cristina Pronello. A methodology for comparing distances traveled by performance-equivalent fixed-route and demand responsive transit services. *Transportation Planning and Technology*, 32(4):377–399, 2009.
- [173] Open Source Routing Machine. OSRM API Documentation. <https://bit.ly/2XCChvM>, 2019.
- [174] Kuang-Hua Chang. Chapter 17 - Design Optimization. In Kuang-Hua Chang., editor, *e-Design*, pages 907 – 1000, Boston, 2015. Academic Press.
- [175] Jean-François Bérubé, Michel Gendreau, and Jean-Yves Potvin. ϵ -constraint method for bi-objective combinatorial optimization problems: Application to the Traveling Salesman Problem with Profits. *European Journal of Operational Research*, 194(1):39 – 50, 2009.
- [176] IBM ILOG CPLEX Optimization Studio. <https://ibm.co/2vXgzRC>, 2020.
- [177] J.-F. Cordeau, G. Desaulniers, J. Desrosiers, M. M. Solomon, and F. Soumis. *VRP with Time Windows*, page 157–193. Society for Industrial and Applied Mathematics, USA, 2001.
- [178] R. Timothy Marler and Jasbir S. Arora. The weighted sum method for multi-objective optimization: new insights. *Structural and Multidisciplinary Optimization*, 41(6):853–862, 2010.
- [179] Thilina Perera, Alok Prakash, Chathura Nagoda Gamage, and Thambipillai Srikanthan. Hybrid Genetic Algorithm for an On-Demand First Mile Transit System Using Electric Vehicles. In Yong Shi, Haohuan Fu, Yingjie Tian, Valeria V. Krzhizhanovskaya, Michael Harold Lees, Jack Dongarra, and Peter M. A. Sloot, editors, *Computational Science – ICCS 2018*, pages 98–113, Cham, 2018. Springer International Publishing.

- [180] Thilina Perera, Alok Prakash, and Thambipillai Srikanthan. Genetic Algorithm based EV Scheduling for On-Demand Public Transit System. In João M. F. Rodrigues, Pedro J. S. Cardoso, Jânio Monteiro, Roberto Lam, Valeria V. Krzhizhanovskaya, Michael H. Lees, Jack J. Dongarra, and Peter M.A. Slood, editors, *Computational Science – ICCS 2019*, pages 595–603, Cham, 2019. Springer International Publishing.
- [181] Zhihai Xiang, Chengbin Chu, and Haoxun Chen. A fast heuristic for solving a large-scale static dial-a-ride problem under complex constraints. *European Journal of Operational Research*, 174(2):1117 – 1139, 2006.
- [182] G. G. Yen and Haiming Lu. Dynamic multiobjective evolutionary algorithm: adaptive cell-based rank and density estimation. *IEEE Transactions on Evolutionary Computation*, 7(3):253–274, 2003.
- [183] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.
- [184] Land Transport Guru. Bus Service Singapore. <https://landtransportguru.net/bus/bus-services/>, 2019.
- [185] Boon Lay Feeders. Feeder Bus Services - Boon Lay. https://sgwiki.com/wiki/Boon_Lay_Feeders, 2019.
- [186] Bureau of Transportation Statistics, United States Department of Transportation. Bus Fuel Consumption and Travel. <https://www.bts.gov/content/bus-fuel-consumption-and-travel>, 2020.
- [187] Thilina Perera, Alok Prakash, Deshya Wijesundera, Thambipillai Srikanthan, and Chathura Nagoda Gamage. Cluster-first, route-second heuristic for ev scheduling in on-demand public transit. In *IEEE International Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM)*, pages 176–181, 2019.
- [188] Thilina Perera, Deshya Wijesundera, Lahiru Wijerathna, and Thambipillai Srikanthan. Directionality-centric Bus Transit Network Segmentation for On-Demand Public Transit. *IET Intelligent Transport Systems*.
- [189] Google Maps Platform. Encoded polyline algorithm format. <https://developers.google.com/maps/documentation/utilities/polylinealgorithm>.
- [190] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, pages 281–297, Berkeley, Calif., 1967. University of California Press.
- [191] L. Kaufman and P.J Rousseeuw. *Partitioning Around Medoids (Program PAM)*, chapter 2, pages 68–125. John Wiley & Sons, Ltd, 2008.

- [192] Konstantin Andreev and Harald Räcke. Balanced Graph Partitioning. In *Proceedings of the Sixteenth Annual ACM Symposium on Parallelism in Algorithms and Architectures*, SPAA '04, page 120–124, New York, NY, USA, 2004. Association for Computing Machinery.
- [193] Andreas Emil Feldmann and Luca Foschini. Balanced Partitions of Trees and Applications. *Algorithmica*, 71(2):354–376, 2015.
- [194] Mattias Andersson, Joachim Gudmundsson, Christos Levkopoulos, and Giri Narasimhan. Balanced Partition of Minimum Spanning Trees. *International Journal of Computational Geometry and Applications*, 13(4):303–316, 2003.
- [195] Thilina Perera, Chathura Nagoda Gamage, Alok Prakash, and Thambipillai Srikanthan. A Simulation Framework for a Real-Time Demand Responsive Public Transit System. In *21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 608–613, 2018.
- [196] Laura Silver. Smartphone Ownership Is Growing Rapidly Around the World, but Not Always Equally. <https://www.pewresearch.org/global/2019/02/05/smartphone-ownership-is-growing-rapidly-around-the-world-but-not-always-equally/>, 2019.
- [197] We Are Social Inc. Digital 2020. <https://wearesocial.com/digital-2020>, 2020.
- [198] TUMCREATE. Sensing and Management for Agile Transport. <https://www.tum-create.edu.sg/research/sensing-and-management-agile-transport>, 2020.
- [199] Qi Wang, Junyu Gao, Wei Lin, and Yuan Yuan. Learning from synthetic data for crowd counting in the wild. *CoRR*, abs/1903.03303, 2019.
- [200] Chengju Zhou, Meiqing Wu, and Siew-Kei Lam. Fast and accurate pedestrian detection using dual-stage group cost-sensitive realboost with vector form filters. In *Proceedings of the 25th ACM International Conference on Multimedia*, page 735–743, New York, NY, USA, 2017. Association for Computing Machinery.
- [201] M. Wu, S. Lam, and T. Srikanthan. A framework for fast and robust visual odometry. *IEEE Transactions on Intelligent Transportation Systems*, 18(12):3433–3448, 2017.
- [202] Land Transport Authority, Singapore. Land Transport Data Mall: Dynamic Datasets. <https://www.mytransport.sg/content/mytransport/home/dataMall/dynamic-data.html#Public%20Transport>, 2019.
- [203] Association for Open Data of Public Transportation Secretariat Office, Japan. 3rd Open Data Challenge for Public Transportation in Tokyo. <https://tokyochallenge.odpt.org/en/>, 2019.
- [204] Roads and Transport Authority (RTA), Dubai. Dubai Pulse, Your Window to Dubai. <https://www.dubaipulse.gov.ae/organisation/rta?organisation=rta&service=rta-bus>, 2019.

- [205] J. Zhang, D. Shen, L. Tu, F. Zhang, C. Xu, Y. Wang, C. Tian, X. Li, B. Huang, and Z. Li. A Real-Time Passenger Flow Estimation and Prediction Method for Urban Bus Transit Systems. *IEEE Transactions on Intelligent Transportation Systems*, 18(11):3168–3178, 2017.
- [206] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997.
- [207] Liyang Tang, Yang Zhao, Javier Cabrera, Jian Ma, and Kwok Leung Tsui. Forecasting Short-Term Passenger Flow: An Empirical Study on Shenzhen Metro. *IEEE Transactions on Intelligent Transportation Systems*, 20(10):3613–3622, October 2019.
- [208] Yong Han, Cheng Wang, Yibin Ren, Shukang Wang, Huangcheng Zheng, and Ge Chen. Short-Term Prediction of Bus Passenger Flow Based on a Hybrid Optimized LSTM Network. *ISPRS International Journal of Geo-Information*, 8(9), 2019.
- [209] Intel. Intel® Xeon® Processor E5-1630 v4 : Product Specification. <https://ark.intel.com/content/www/us/en/ark/products/92987/intel-xeon-processor-e5-1630-v4-10m-cache-3-70-ghz.html>, 2020.